J Autom Reasoning
DOI 10.1007/s10817-016-9364-6

Labelled Interpolation Systems for Hyper-Resolution, Clausal, and Local Proofs

Matthias Schlaipfer · Georg Weissenbacher

Received: 1 October 2014 / Accepted: 4 February 2016

Abstract Craig's interpolation theorem has numerous applications in model checking, automated reasoning, and synthesis. There is a variety of interpolation systems which derive interpolants from refutation proofs; these systems are ad-hoc and rigid in the sense that they provide exactly one interpolant for a given proof. In previous work, we introduced a parametrised interpolation system which subsumes existing interpolation methods for propositional resolution proofs and enables the systematic variation of the logical strength and the elimination of non-essential variables in interpolants. In this paper, we generalise this system to propositional hyper-resolution proofs as well as clausal proofs. The latter are generated by contemporary SAT solvers. Finally, we show that, when applied to local (or split) proofs, our extension generalises two existing interpolation systems for first-order logic and relates them in logical strength.

1 Introduction

Craig interpolation [14] has proven to be an effective heuristic in applications such as model checking, where it is used as an approximate method for computing invariants of transition systems [39,54], and synthesis, where interpolants represent deterministic implementations of specifications given as relations [31]. The intrinsic properties of interpolants enable concise abstractions in verification and smaller circuits in synthesis. Intuitively, stronger interpolants provide more precision [29, 46], and interpolants with fewer variables lead to smaller designs [7,31]. However, interpolation is mostly treated as a black box, leaving no room for a systematic exploration of the solution space. In addition, the use of different interpolation systems complicates a comparison of their interpolants. We present a novel framework which generalises a number of existing interpolation techniques and supports a systematic variation and comparison of the generated interpolants.

Supported by the Austrian National Research Network S11403-N23 (RiSE), the LogiCS doctoral program W1255-N23 of the Austrian Science Fund (FWF), and by the Vienna Science and Technology Fund (WWTF) through grant VRG11-005.

TU Wien, Austria

Contributions. We present a novel *parametrised* interpolation system which extends our previous work on propositional interpolation [16].

- The extended system supports hyper-resolution (see § 3) and allows for systematic variation of the logical strength (with an additional degree of freedom over [16]) and the elimination of non-essential literals [15] in interpolants.
- We generalise (in § 4) our interpolation system for hyper-resolution steps to clausal refutations generated by contemporary SAT solvers such as PicoSAT [5], allowing us to avoid the generation of intermediate interpolants.
- When applied to local (or split) proofs [30], the extended interpolation system generalises the existing interpolation systems for first-order logic presented in [32] and [55] and relates them in logical strength (§ 5).

This paper is an extended version of [56], and includes novel results on interpolation for clausal proofs and empirical results (see § 4).

2 Background

This section introduces our notation (§ 2.1) and restates the main results of our previous paper on labelled interpolation systems [16] in § 2.2.

2.1 Formulae and Proofs

In our setting, the term *formula* refers to either a propositional logic formula or a formula in standard first-order logic.

Propositional Formulae. We work in the standard setting of propositional logic over a set X of propositional variables, the logical constants T and F (denoting true and false, respectively), and the standard logical connectives $\wedge$, $\vee$, $\Rightarrow$, and $\neg$ (denoting conjunction, disjunction, implication, and negation, respectively).

Moreover, let $\mathtt{Lit}_X = \{x, \overline{x} \,|\, x \in X\}$ be the set of literals over X, where $\overline{x}$ is short for $\neg x$. We write $\mathrm{var}(t)$ for the variable occurring in the literal $t \in \mathtt{Lit}_X$. A clause C is a set of literals. The empty clause $\square$ contains no literals and is used interchangeably with F. The disjunction of two clauses C and D is their union, denoted $C \vee D$, which is further simplified to $C \vee t$ if D is the singleton $\{t\}$. In clauses, we sometimes omit the disjunction $\vee$ to save space. A propositional formula in Conjunctive Normal Form (CNF) is a conjunction of clauses, also represented as a set of clauses.

First-Order Logic. The logical connectives from propositional logic carry over into first-order logic. We fix an enumerable set of variables, function and predicate symbols over which formulae are built in the usual manner. The *vocabulary* of a formula A is the set of its function and predicate symbols. $\mathcal{L}(A)$ refers to the set of well-formed formulae which can be built over the vocabulary of A.

Variables may be universally ($\forall$) or existentially ($\exists$) quantified. A formula is *closed* if all its variables are quantified and *ground* if it contains no variables. As previously, conjunctions of formulae are also represented as sets.

Given a formula A in either first-order or propositional logic, we use $\mathrm{Var}(A)$ to denote the set of free (unquantified) variables in A.

Inference Rules and Proofs. We write $A_1, \cdots, A_n \models A$ to denote that the formula A holds in all models of $A_1, \ldots, A_n$ (where $n \geq 0$). An inference rule

$$\frac{A_1 \quad \cdots \quad A_n}{A} \tag{1}$$

associates zero or more *premises* (or *antecedents*) $A_1, \ldots, A_n$ with a *conclusion* A. The inference rule (1) is *sound* if $A_1, \ldots, A_n \models A$ holds. A (sound) inference system $\mathcal{I}$ is a set of (sound) inference rules.

The propositional *resolution* rule (Res), for example, is a sound inference rule stating that an assignment satisfying the clauses $C \vee x$ and $D \vee \overline{x}$ also satisfies $C \vee D$. The clauses $C \vee x$ and $D \vee \overline{x}$ are the *antecedents*, x is the *pivot*, and the conclusion $C \vee D$ is called the *resolvent*. $\mathrm{Res}(C, D, x)$ denotes the resolvent of C and D with the pivot x.

Definition 1 (Proof) A *proof* (or derivation) P in an inference system $\mathcal{I}_P$ is a directed acyclic graph $(V_P, E_P, \ell_P, \mathtt{s}_P)$, where V_P is a set of vertices, E_P is a set of edges, ℓ_P is a function mapping vertices to formulae, and $\mathtt{s}_P \in V_P$ is the sink vertex. An *initial vertex* has in-degree 0. All other vertices are *internal* and have in-degree ≥ 1. The sink has out-degree 0. Each internal vertex v with edges $(v_1, v), \ldots, (v_m, v) \in E_P$ is associated with an inference rule $\mathsf{Inf} \in \mathcal{I}_P$ with antecedents $\ell_P(v_1), \ldots, \ell_P(v_m)$ and conclusion $\ell_P(v)$.

The subscripts above are dropped if clear. A vertex v_i in P is a *parent* of v_j if $(v_i, v_j) \in E_P$. A proof P is a *refutation* if $\ell_P(\mathtt{s}_P) = \mathsf{F}$. Let A and B be conjunctive formulae. A refutation P of an unsatisfiable formula $A \wedge B$ is an (A, B)-*refutation* (i.e., for each initial vertex $v \in V_P$, $\ell_P(v)$ is a conjunct of A or a conjunct of B). A proof is *closed* (*ground*, respectively) if $\ell_P(v)$ is closed (ground) for all $v \in V_P$.

In the following, we use the propositional resolution calculus to instantiate Definition 1.

Definition 2 (Resolution Proof) A *resolution proof* R is a proof in the inference system comprising only the resolution rule Res. Consequently, ℓ_R maps each vertex $v \in V_R$ to a clause, and all internal vertices have in-degree 2. Let piv_R be the function mapping internal vertices to pivot variables. For an internal vertex v and $(v_1, v), (v_2, v) \in E_R$, $\ell_R(v) = \mathrm{Res}(\ell_R(v_1), \ell_R(v_2), piv_R(v))$.

Note that the value of ℓ_R at internal vertices is determined by that of ℓ_R at initial vertices and the pivot function piv_R. We write v^+ for the parent of v with $piv(v)$ in $\ell(v^+)$ and v^- for the parent with $\neg piv(v)$ in $\ell(v^-)$. A resolution proof R is a *resolution refutation* if $\ell_R(\mathtt{s}_R) = \square$.

2.2 Interpolation Systems and Labelling Functions

There are numerous variants and definitions of Craig's interpolation theorem [14]. We use the definition of a Craig interpolant introduced by McMillan [39]:

Definition 3 (Interpolant) A Craig *interpolant* for a pair of formulae (A, B), where $A \wedge B$ is unsatisfiable, is a formula I whose free variables, function and predicate symbols occur in both A and B, such that $A \Rightarrow I$ and $B \Rightarrow \neg I$ hold.

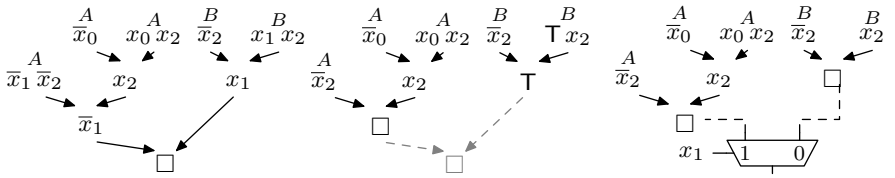


Fig. 1 The interpolant $\overline{x}_1$ acts as a "separator" for the resolution refutation

Craig's interpolation theorem guarantees the existence of such an interpolant for unsatisfiable pairs of formulae (A, B) in first order logic. Consequently, it also holds in the propositional setting, where the conditions of Definition 3 reduce to $A \Rightarrow I$, $B \Rightarrow \neg I$, and $\mathrm{Var}(I) \subseteq \mathrm{Var}(A) \cap \mathrm{Var}(B)$.

Example 1 Let $A = (\overline{x}_0) \wedge (x_0 \vee x_2) \wedge (\overline{x}_1 \vee \overline{x}_2)$ and $B = (\overline{x}_2) \wedge (x_1 \vee x_2)$. Then $I = \overline{x}_1$ is an interpolant for (A, B). Intuitively, $\overline{x}_1$ interpolant acts as a "separator" for the underlying refutation proof (the leftmost proof in Figure 1). By setting $\overline{x}_1$ to F we obtain a refutation of the A-partition, as illustrated in Figure 1. Similarly, setting $\overline{x}_1$ to T yields a refutation for B — the interpolant can be understood as a multiplexer. Equivalently, I is T if A is T, and $\neg I$ is T if B is T. ◁

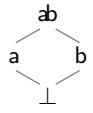
Numerous techniques to construct interpolants have been proposed (c.f. § 6). In particular, there is a class of algorithms that derive interpolants from proofs; the first such algorithm for the sequent calculus is presented in Maehara's constructive proof [37] of Craig's theorem. In this paper, we focus on *interpolation systems* that construct an interpolant from an (A, B)-refutation by mapping the vertices of a resolution proof to a formula called the *partial interpolant*.

Formally, an interpolation system Itp is a function that given an (A, B)-refutation R yields a function, denoted $\mathsf{Itp}(R, A, B)$, from vertices in R to formulae over $\mathrm{Var}(A) \cap \mathrm{Var}(B)$. An interpolation system is *correct* if for every (A, B)-refutation R with sink $\mathtt{s}$, it holds that $\mathsf{Itp}(R, A, B)(\mathtt{s})$ is an interpolant for (A, B). We write $\mathsf{Itp}(R)$ for $\mathsf{Itp}(R, A, B)(\mathtt{s})$ when A and B are clear. Let v be a vertex in an (A, B)-refutation R. The pair $(\ell(v), \mathsf{Itp}(R, A, B)(v))$ is an *annotated clause* and is written $\ell(v)\ [\mathsf{Itp}(R, A, B)(v)]$ in accordance with [40].

In the following, we review the labelled interpolation systems we introduced in [16]. Labelled interpolation generalises several existing propositional interpolation systems presented by Huang [28], Krajíček [33] and Pudlák [42], and McMillan [39]. A distinguishing feature of a labelled interpolation system is that it assigns an individual label $\mathsf{c} \in \{\bot, \mathsf{a}, \mathsf{b}, \mathsf{ab}\}$ to *each literal* in the resolution refutation.

Definition 4 (Labelling Function) Let $(\mathcal{S}, \sqsubseteq, \sqcap, \sqcup)$ be the lattice below, where $\mathcal{S} = \{\bot, \mathsf{a}, \mathsf{b}, \mathsf{ab}\}$ is a set of symbols and $\sqsubseteq$, $\sqcap$ and $\sqcup$ are defined by the Hasse diagram to the right. A *labelling function* $L_R : V_R \times \mathtt{Lit} \to \mathcal{S}$ for a refutation R over a set of literals $\mathtt{Lit}$ satisfies that for all $v \in V_R$ and $t \in \mathtt{Lit}$:

1. $L_R(v, t) = \bot$ iff $t \notin \ell_R(v)$
2. $L_R(v, t) = L_R(v_1, t) \sqcup \cdots \sqcup L_R(v_m, t)$ for an internal vertex v, its parents $\{v_1, \cdots, v_m\}$, and literal $t \in \ell_R(v)$.



Due to condition (2) above, the labels of literals at initial vertices completely determine the labelling function for literals at internal vertices. The following

For an initial vertex v with $\ell(v) = C$

(A-clause) $\dfrac{}{C \quad [C\restriction_{\mathsf{b}}]}$ if $C \in A$ (B-clause) $\dfrac{}{C \quad [\neg(C\restriction_{\mathsf{a}})]}$ if $C \in B$

For an internal vertex v with $piv(v) = x$, $\ell(v^+) = C_1 \vee x$ and $\ell(v^-) = C_2 \vee \overline{x}$

$$\frac{C_1 \vee x \quad [I_1] \qquad C_2 \vee \overline{x} \quad [I_2]}{C_1 \vee C_2 \quad [I_3]}$$

(A-Res)	if $L(v^+, x) \sqcup L(v^-, \overline{x}) = \mathsf{a}$,	$I_3 \stackrel{\text{def}}{=}$	$I_1 \vee I_2$
(AB-Res)	if $L(v^+, x) \sqcup L(v^-, \overline{x}) = \mathsf{ab}$,	$I_3 \stackrel{\text{def}}{=}$	$(x \vee I_1) \wedge (\overline{x} \vee I_2)$
(B-Res)	if $L(v^+, x) \sqcup L(v^-, \overline{x}) = \mathsf{b}$,	$I_3 \stackrel{\text{def}}{=}$	$I_1 \wedge I_2$

Fig. 2 Labelled interpolation system for resolution proofs

condition ensures that a labelling function respects the *locality* of a literal t in accordance with (A, B). A literal t is A-*local* and therefore labelled a if $\mathrm{var}(t) \in \mathrm{Var}(A) \setminus \mathrm{Var}(B)$. Conversely, t is B-*local* and therefore labelled b if $\mathrm{var}(t) \in \mathrm{Var}(B) \setminus \mathrm{Var}(A)$. Literals t for which $\mathrm{var}(t) \in \mathrm{Var}(A) \cap \mathrm{Var}(B)$ are *shared* and can be labelled a, b, or ab (which generalises existing interpolation systems).

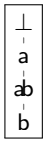
Definition 5 (Locality) A labelling function L for an (A, B)-refutation R *preserves locality* if for any initial vertex v and literal t in R
1. $\mathsf{a} \sqsubseteq L(v, t)$ implies that $\mathrm{var}(t) \in \mathrm{Var}(A)$, and
2. $\mathsf{b} \sqsubseteq L(v, t)$ implies that $\mathrm{var}(t) \in \mathrm{Var}(B)$.

For a given labelling function L, we define the downward *projection* of a clause at a vertex v with respect to $\mathsf{c} \in \mathcal{S}$ as $\ell(v)\restriction_{\mathsf{c},L} \stackrel{\text{def}}{=} \{t \in \ell(v) \mid L(v, t) \sqsubseteq \mathsf{c}\}$ and the upward projection $\ell(v)\restriction^{\mathsf{c},L}$ as $\ell(v)\restriction^{\mathsf{c},L} \stackrel{\text{def}}{=} \{t \in \ell(v) \mid \mathsf{c} \sqsubseteq L(v, t)\}$. The subscript L is omitted if clear from the context.

Definition 6 (Labelled Interpolation System for Resolution) Let L be a locality preserving labelling function for an (A, B)-refutation R. The labelled interpolation system $\mathsf{Itp}(L)$ maps vertices in R to partial interpolants as defined in Figure 2.

Labelling functions provide control over the interpolants constructed from a resolution proof. Firstly, labelled interpolation systems support the elimination of *non-essential* (*peripheral* [50], respectively) variables from interpolants [15]. Secondly, labelled interpolation systems – and their respective interpolants – are ordered by logical strength. A labelled interpolation system $\mathsf{Itp}(L)$ is *stronger than* $\mathsf{Itp}(L')$ if for all refutations R (for which L and L' are locality preserving labelling functions), $\mathsf{Itp}(L, R) \Rightarrow \mathsf{Itp}(L', R)$. The partial order $\preceq$ on labelling functions (first introduced in [16]) guarantees an ordering in strength:

Definition 7 (Strength Order) We define the total order $\preceq$ on the lattice $\mathcal{S} = \{\bot, \mathsf{a}, \mathsf{b}, \mathsf{ab}\}$ as $\mathsf{b} \preceq \mathsf{ab} \preceq \mathsf{a} \preceq \bot$ (c.f. the Hasse diagram to the right). Let L and L' be labelling functions for an (A, B)-refutation R. The function L is stronger than L', denoted $L \preceq L'$, if for all $v \in V_R$ and $t \in \ell(v)$, $L(v, t) \preceq L'(v, t)$.



Theorem 2 in [16] shows that if L is a stronger labelling function than L', the interpolant obtained from $\mathsf{Itp}(L)$ logically implies the one obtained from $\mathsf{Itp}(L')$.

3 Interpolation for Hyper-Resolution

In this section, we extend labelled interpolation systems to a richer inference system, in particular, the inference system comprising (propositional) *hyper-resolution* [43]. Hyper-resolution is a condensation of a derivation consisting of several resolutions and avoids the construction of intermediate clauses. Hyper-resolution has several applications in propositional satisfiability checking, such as pre-processing [21] of formulae or as an integral part of the solver (e.g., [2]).

Positive hyper-resolution combines a single clause (called the *nucleus*) containing n negative literals $\overline{x}_1, \ldots, \overline{x}_n$ and n *satellite* clauses each of which contains one of the corresponding non-negated literals x_i (where $1 \leq i \leq n$):

$$\frac{\overbrace{(C_1 \vee x_1) \quad \cdots \quad (C_n \vee x_n)}^{\text{satellites}} \qquad \overbrace{(\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee D)}^{\text{nucleus}}}{\bigvee_{i=1}^{n} C_i \vee D} \;\; [\mathsf{HyRes}]$$

In *negative* hyper-resolution the roles of x_i and $\overline{x}_i$ are exchanged.

Definition 8 (Hyper-Resolution Proof) A *hyper-resolution proof* R is a proof using only the inference rule HyRes. Accordingly, ℓ_R maps each vertex $v \in V_R$ to a clause, and all internal vertices have in-degree ≥ 2. Each internal vertex v has $n \geq 1$ parents $v_1^+, \ldots, v_n^+$ such that $\ell_R(v_i^+) = C_i \vee x_i$ and one parent v^- with $\ell_R(v^-) = \overline{x}_1 \vee \cdots \vee \overline{x}_n \vee D$, and consequently, $\ell_R(v) = \bigvee_{i=1}^{n} C_i \vee D$.

The definition of labelling functions (Definition 4) readily applies to hyper-resolution proofs. Note that $\preceq$ is not a total order on labelling functions. Lemma 1 (a generalisation of Lemma 3 in [16] to hyper-resolution proofs) enables a comparison of labelling functions based solely on the values at the *initial* vertices.

Lemma 1 *Let L and L' be labelling functions for an (A, B)-refutation R. If $L(v, t) \preceq L'(v, t)$ for all initial vertices v and literals $t \in \ell(v)$, then $L \preceq L'$.*

A proof of Lemma 1 is given in Appendix A. In the following, we generalise labelled interpolation systems to hyper-resolution. The underlying intuition is to replace the multiplexer in the case AB-Res in Definition 6 with a general multiplexer controlled by the pivot literals of the hyper-resolution step. This idea is illustrated in Figure 3 for the proof in Example 1 and formalised in the following definition:

Definition 9 (Labelled Interpolation System for Hyper-Resolution) Let L be a locality preserving labelling function for an (A, B)-refutation R, where R is a hyper-resolution proof. The labelled interpolation system $\mathsf{Itp}(L)$ maps vertices in R to partial interpolants as defined in Figure 4.

The interpolation system leaves us a choice for internal nodes AB-HyRes. We will use Itp_1 (Itp_2, respectively) to refer to the interpolation system that always chooses case 1 (case 2, respectively). Note furthermore that Definition 6 and Definition 9 are equivalent in the special case where $n = 1$.

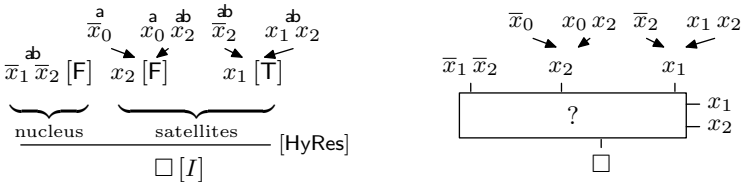


Fig. 3 Generalising labelled interpolation to hyper-resolution

For an initial vertex v with $\ell(v) = C$

(A-clause) $\dfrac{}{C \quad [C\restriction_{\mathsf{b}}]}$ if $C \in A$ (B-clause) $\dfrac{}{C \quad [\neg(C\restriction_{\mathsf{a}})]}$ if $C \in B$

For an internal vertex v with predecessors $\{v_1^+, \ldots, v_n^+, v^-\}$ (where $n \geq 1$) with $\ell(v_i^+) = (C_i \vee x_i)$, for $1 \leq i \leq n$, and $\ell(v^-) = (D \vee \overline{x}_1 \vee \cdots \vee \overline{x}_n)$

$$\frac{C_1 \vee x_1 \quad [I_1] \qquad \cdots \qquad C_n \vee x_n \quad [I_n] \qquad \overline{x}_1 \vee \cdots \vee \overline{x}_n \vee D \quad [I_{n+1}]}{\bigvee_{i=1}^n C_i \vee D \quad [I]}$$

(A-HyRes) if $\forall i \in \{1..n\} \,.\, L(v_i^+, x_i) \sqcup L(v^-, \overline{x}_i) = \mathsf{a}$, $\quad I \stackrel{\text{def}}{=} \bigvee_{i=1}^{n+1} I_i$

(AB-HyRes) if $\forall i \in \{1..n\} \,.\, L(v_i^+, x_i) \sqcup L(v^-, \overline{x}_i) = \mathsf{ab}$,

1.) $I \stackrel{\text{def}}{=} \bigwedge_{i=1}^n (x_i \vee I_i) \wedge \big(I_{n+1} \vee \bigvee_{i=1}^n \overline{x}_i\big)$, or

2.) $I \stackrel{\text{def}}{=} \bigvee_{i=1}^n (\overline{x}_i \wedge I_i) \vee \big(I_{n+1} \wedge \bigwedge_{i=1}^n x_i\big)$

(B-HyRes) if $\forall i \in \{1..n\} \,.\, L(v_i^+, x_i) \sqcup L(v^-, \overline{x}_i) = \mathsf{b}$, $\quad I \stackrel{\text{def}}{=} \bigwedge_{i=1}^{n+1} I_i$

Fig. 4 Labelled interpolation system for hyper-resolution proofs

Remark 1 Note that unlike the interpolation system for ordinary resolution proofs presented in Definition 6, Itp is not total for hyper-resolution proofs: the case split requires the pivots of the hyper-resolution step to be uniformly labelled, i.e., the rules A-HyRes, AB-HyRes, and B-HyRes require $L(v_i^+, x_i) \sqcup L(v^-, \overline{x}_i)$ to be a, ab, or b, respectively, for all $i \in \{1..n\}$. This limitation is addressed in § 4.1.

In the following we present a conditional correctness result:

Theorem 1 (Correctness) *For any (A, B)-refutation R (where R is a hyper-resolution proof) and locality preserving labelling function L, $\mathsf{Itp}(L, R)$ (if defined) is an interpolant for (A, B).*

The proof of Theorem 1 (given in Appendix A) establishes that for each vertex $v \in V_R$ with $\ell_R(v) = C$ and $I = \mathsf{Itp}(L, R)(v)$, the following conditions hold:

- $A \wedge \neg(C\restriction_{\mathsf{a},L}) \Rightarrow I$,
- $B \wedge \neg(C\restriction_{\mathsf{b},L}) \Rightarrow \neg I$, and
- $\mathrm{Var}(I) \subseteq \mathrm{Var}(A) \cap \mathrm{Var}(B)$.

For $\ell_R(\mathsf{s}) = \square$, this establishes the correctness of the system.

We emphasise that Theorem 1 does not constrain the choice for the case AB-HyRes. Since both $\mathsf{Itp}_1(L, R)$ and $\mathsf{Itp}_2(L, R)$ satisfy the conditions above, this choice does not affect the correctness of the interpolation system. In fact, it is valid to *mix* both systems by defining a choice function $\chi : V_R \to \{1, 2\}$ which determines

which interpolation system is chosen at each internal node. We use $\mathsf{Itp}_\chi(L, R)$ to denote the resulting interpolation system. This modification, however, may have an impact on the logical strength of the resulting interpolant.

Theorem 2 *Let the hyper-resolution proof R be an (A, B)-refutation and L be a locality preserving labelling function. Moreover, let $\mathsf{Itp}_\chi(L, R)$ and $\mathsf{Itp}_{\chi'}(L, R)$ be labelled interpolation systems (defined for L, R) with the choice functions χ and χ', respectively. Then $\mathsf{Itp}_\chi(L, R) \Rightarrow \mathsf{Itp}_{\chi'}(L, R)$ if $\chi(v) \leq \chi'(v)$ for all internal vertices $v \in V_R$.*

Proof sketch: This follows (by structural induction over R) from

$$\left(\bigwedge_{i=1}^{n}(x_i \vee I_i) \wedge \left(I_{n+1} \vee \bigvee_{i=1}^{n} \overline{x}_i\right)\right) \Rightarrow \left(\bigvee_{i=1}^{n}(\overline{x}_i \wedge I_i) \vee \left(I_{n+1} \wedge \bigwedge_{i=1}^{n} x_i\right)\right). \qquad \blacksquare$$

Note that the converse implication does not hold; a simple counterexample for an internal vertex with $n = 2$ is the assignment $x_1 = x_2 = \mathsf{F}$, $I_1 = \mathsf{T}$, and $I_2 = I_3 = \mathsf{F}$.

The final theorem in this section extends the result of Theorem 2 in [16] to hyper-resolution proofs:

Theorem 3 *If L and L' are labelling functions for an (A, B)-refutation R (R being a hyper-resolution proof) and $L \preceq L'$ such that $\mathsf{Itp}_i(L, R)$ as well as $\mathsf{Itp}_i(L', R)$ are defined, then $\mathsf{Itp}_i(L, R) \Rightarrow \mathsf{Itp}_i(L', R)$ (for a fixed $i \in \{1, 2\}$).*

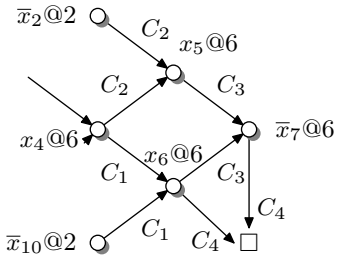
The proof of Theorem 3, provided in Appendix A, is led by structural induction over R. For any vertex v in R, let I_v and I'_v be the partial interpolants due to $\mathsf{Itp}_i(L, R)$ and $\mathsf{Itp}_i(L', R)$, respectively. We show that $I_v \Rightarrow I'_v \vee \{t \in \ell_R(v) \mid L(v, t) \sqcup L'(v, t) = \mathsf{ab}\}$ for all vertices v, establishing $I_v \Rightarrow I'_v$ for the sink to show that $\mathsf{Itp}_i(L, R) \Rightarrow \mathsf{Itp}_i(L', R)$.

Theorems 2 and 3 enable us to fine-tune the strength of interpolants, since the sets of all labelling and choice functions ordered by $\preceq$ and $\leq$, respectively, form complete lattices (c.f. [16, Theorem 3]). Finally, we remark that the Theorems 2 and 3 are orthogonal. The former fixes the labelling function L, whereas the latter fixes the choice function χ.

4 Interpolation for Clausal Proofs

Contemporary SAT solvers such as MiniSAT [17] and PicoSAT [5] are based on conflict-driven clause learning (CDCL) [49]. The CDCL algorithm avoids the repeated exploration of conflicting variable assignments by caching the causes of failures in the form of learned clauses. To this end, the solver stores assignments (decisions) and their implications in an *implication graph*, from which it derives learned clauses in case of a conflict. We refrain from providing a description of CDCL, since numerous excellent expositions are available (e.g. [6, 34]). The following example, borrowed from [38], illustrates the construction of resolution proofs in CDCL solvers.

Example 2 Figure 5 shows a partial implication graph for the clauses $(\overline{x}_4\, x_{10}\, x_6)$, $(\overline{x}_4\, x_2\, x_5)$, $(\overline{x}_5\, \overline{x}_6\, \overline{x}_7)$, and $(\overline{x}_6\, x_7)$. Nodes represent assignments (annotated with the corresponding decision level, e.g., $\overline{x}_{10}@2$ indicates that x_{10} was assigned F at level 2) and each edge represents an implication deriving from a clause in which all but one literal is assigned under the current assignment. The final node $\square$ indicates



$C_1 \equiv \{\overline{x}_4,\, x_{10},\, x_6\}$
$C_2 \equiv \{\overline{x}_4,\, x_2,\, x_5\}$
$C_3 \equiv \{\overline{x}_5,\, \overline{x}_6,\, \overline{x}_7\}$
$C_4 \equiv \{\overline{x}_6,\, x_7\}$

Fig. 5 Implication graph and conflict analysis

a conflict under the current assignment, and its incoming edges are annotated with the conflicting clause C_4. This conflict stems from the fact that C_4 disagrees with C_1 and C_3 on the implied literals $\overline{x}_6$ and x_7, respectively. By subsequently resolving on the conflicting literals, we obtain

$$C_5 = \mathrm{Res}(C_4, C_3, x_7) = \{\overline{x}_5,\, \overline{x}_6\} \qquad \text{and} \qquad C_6 = \mathrm{Res}(C_5, C_1, x_6) = \{\overline{x}_4,\, \overline{x}_5,\, x_{10}\}\,.$$

The clause C_6 disagrees with C_2 on the implied literal x_5. The resolvent of these clauses is $C_7 = \mathrm{Res}(C_6, C_2, x_5) = \{x_2,\, \overline{x}_4,\, x_{10}\}$. C_7 contains a single literal (x_4) assigned at decision level 6 while still conflicting with the current partial assignment. Accordingly, reverting the decision x_4 at level 6 and adding C_7 as learned clause prevents the solver from revisiting this part of the search space. ◁

The learned clause in Example 2 is a consequence of clauses of the original instance and previously learned clauses. Each learned clause is the conclusion of a chain of resolution steps.

Definition 10 (Chain) A (resolution) *chain* of length n is a tuple consisting of an input clause D_0 and an ordered sequence of clause-pivot pairs $\langle C_i, x_i\rangle$ (where $1 \leq i \leq n$). The final resolvent D_n of a resolution chain is defined inductively as $D_i = \mathrm{Res}(D_{i-1}, C_i, x_i)$.

A resolution chain generated by a CDCL solver has the following properties [4]:

- Regularity: each pivot variable is resolved upon at most once in the chain.
- Linearity: each intermediate clause D_i $(1 \leq i \leq n)$ in a chain is obtained by deriving D_{i-1} with an initial clause C_j $(2 \leq j \leq n)$ or with a previously derived clause D_k $(k < i - 1)$.
- Tree-likeness: Each derived clause is used exactly once in the chain.

A resolution derivation with these properties is called trivial [4]. For reasons of performance, proof-logging solvers discard all intermediate resolvents generated during the construction of a conflict clause and retain only resolution chains. Clausal proofs [22,25] and proofs stored in the TraceCheck-format[1] moreover omit the pivot literals as well as the order of the resolution steps, recording only the unordered set of clauses $D_0, C_1, \ldots, C_n$ for each resolution chain.

If D_0 is a nucleus and $C_1, \ldots, C_n$ are suitable satellites, the chain can be replaced by a hyper-resolution step assuming its conclusion D_n satisfies the HyRes rule. In general, this may not be the case: $D_0 = \{x_1,\, x_2\}, C_1 = \{\overline{x}_2,\, x_3\}, C_2 =$

[1] `http://fmv.jku.at/tracecheck/README.tracecheck`

$\{\overline{x}_3, x_4\}$ is a valid resolution chain (with conclusion $\{x_1, x_4\}$) that does not match the antecedents HyRes rule.

To address this problem, we introduce a more general inference rule which requires the existence of a resolution chain matching its premises and conclusion as a side condition. Each of the n premises contains a non-empty (sub-)set of pivot literals P_i which occur in opposite phase in the other clauses of the premise. The clause learning algorithm illustrated in Example 2 results in resolution chains that satisfy the following properties:

- The pivot literals $\bigcup_{i=1}^{n} P_i$ do not occur in the conclusion of the chain.

  Remark 2 The algorithm resolves upon pivot literals that are implied but not yet assigned at the respective node in the implication graph. Accordingly, the clauses preceding the node in the implication graph cannot contain the implied literal, since they would otherwise not be unit. Therefore, a pivot literal, once resolved, is never re-introduced in a resolution chain.

- The conjunction $\bigwedge_{i=1}^{n} P_i$ is unsatisfiable (guaranteed by the existence of a resolution chain).

These properties are reflected in the following inference rule:

Definition 11 (TraceCheck Resolution) Let $D_1, \dots D_n$ be an (unordered) set of clauses. Let $P_i \stackrel{\text{def}}{=} \{t \in D_i \mid \exists j \,.\, 1 \leq j \leq n \wedge j \neq i \wedge \bar{t} \in D_j\}$ and $C_i \stackrel{\text{def}}{=} D_i \setminus P_i$. If there exists a resolution chain $D_1, \langle D_2, x_2\rangle, \dots, \langle D_n, x_n\rangle$ with conclusion $\bigvee_{i=1}^{n} C_i$ then

$$\frac{(C_1 \vee P_1) \quad \cdots \quad (C_n \vee P_n)}{\bigvee_{i=1}^{n} C_i} \quad [\mathsf{TCRes}]$$

Analogously to Definition 8, we introduce the notion of a clausal proof.

Definition 12 (Clausal Proof) A clausal proof R is a proof using only the inference rule TCRes. Accordingly, ℓ_R maps each vertex $v \in V_R$ to a clause and every internal vertex v has $n \geq 2$ parents $v_1, \dots, v_n$ such that $\ell_R(v_i) = C_i \vee P_i$ (as in Definition 11). Consequently, $\ell_R(v) = \bigvee_{i=1}^{n} C_i$.

The following definition extends the interpolation system for hyper-resolution proofs presented in § 3 to clausal proofs.

Definition 13 (Labelled Interpolation System for Clausal Proofs) Let L be a locality preserving labelling function for an (A, B)-refutation R, where R is a clausal proof. The labelled interpolation system $\mathsf{Itp}(L)$ maps vertices in R to partial interpolants as defined in Figure 6.

Note that the interpolation system in Definition 13 is a generalisation of the interpolation system for hyper-resolution (Definition 9). Its correctness is established using a similar argument as used for Theorem 1. The proof of the following theorem is provided in Appendix A.

Theorem 4 (Correctness) *For any (A, B)-refutation R (where R is a clausal proof) and locality preserving labelling function L, $\mathsf{Itp}(L, R)$ (if defined) is an interpolant for (A, B).*

The results of Theorems 2 and 3 can be generalised to clausal proofs in a straight-forward manner. We omit the discussion of the details.

For an initial vertex v with $\ell(v) = C$

(A-clause) $\dfrac{}{C \quad [C\restriction_{\mathsf{b}}]}$ if $C \in A$ (B-clause) $\dfrac{}{C \quad [\neg(C\restriction_{\mathsf{a}})]}$ if $C \in B$

For an internal vertex v with predecessors $\{v_1, \ldots, v_n\}$ (where $n \geq 1$) with $\ell(v_i) = (C_i \vee P_i)$, for $1 \leq i \leq n$

$$\frac{C_1 \vee P_1 \quad [I_1] \quad \cdots \quad C_n \vee P_n \quad [I_n]}{\bigvee_{i=1}^{n} C_i \quad [I]}$$

(A-TCRes) if $\forall i \in \{1..n\} \,.\, t \in P_i \Rightarrow L(v_i, t) = \mathsf{a}$, $I \stackrel{\text{def}}{=} \bigvee_{i=1}^{n} I_i$

(AB-TCRes) if $\forall i, j \in \{1..n\}$, $i \neq j \,.\, x \in P_i \wedge \overline{x} \in P_j \Rightarrow L(v_i, x) \sqcup L(v_j, \overline{x}) = \mathsf{ab}$,

1.) $I \stackrel{\text{def}}{=} \bigwedge_{i=1}^{n} I_i \vee P_i$ or

2.) $I \stackrel{\text{def}}{=} \bigvee_{i=1}^{n} I_i \wedge \neg P_i$

(B-TCRes) if $\forall i \in \{1..n\} \,.\, t \in P_i \Rightarrow L(v_i, t) = \mathsf{b}$, $I \stackrel{\text{def}}{=} \bigwedge_{i=1}^{n} I_i$

Fig. 6 Labelled interpolation system for clausal proofs

$$\dfrac{\dfrac{(\overset{\mathsf{a}}{x_1} \vee C_1)\ (\overset{\mathsf{ab}}{x_2} \vee C_2)\ (\overset{\mathsf{a}}{x_3} \vee C_3)\ (\overset{\mathsf{a}}{x_4} \vee C_4) \quad (\overset{\mathsf{a}}{\overline{x}_1} \vee \overset{\mathsf{a}}{\overline{x}_2} \vee \overset{\mathsf{a}}{\overline{x}_3} \vee \overset{\mathsf{b}}{\overline{x}_4} \vee D)}{C_1 \vee C_2 \vee C_3 \vee C_4 \vee D}}{\vdots}$$

$$\dfrac{\dfrac{(\overset{\mathsf{a}}{x_1} \vee C_1)\ (\overset{\mathsf{a}}{x_3} \vee C_3) \quad (\overset{\mathsf{a}}{\overline{x}_1} \vee \overset{\mathsf{a}}{\overline{x}_2} \vee \overset{\mathsf{a}}{\overline{x}_3} \vee \overset{\mathsf{b}}{\overline{x}_4} \vee D)}{\vdots}\ [A\text{-HyRes}] \qquad (\overset{\mathsf{ab}}{x_2} \vee C_2)\ (\overset{\mathsf{a}}{x_4} \vee C_4)}{C_1 \vee C_2 \vee C_3 \vee C_4 \vee D}\ [AB\text{-HyRes}]$$

Fig. 7 Splitting hyper-resolution steps

4.1 Splitting and Reordering Resolution Chains

Just like the interpolation system for hyper-resolution proofs, the interpolation system in Definition 13 has the deficiency that the function $\mathsf{Itp}(L)$ is not total: there are labelling functions L for which the result of $\mathsf{Itp}(L)$ is undefined. This problem arises whenever the pivots in a TraceCheck resolution step are not uniformly labelled, and therefore none of the rules in Figure 6 is applicable.

Instead of adapting the interpolation system, we address the problem by splitting the corresponding resolution chains. A single chain can be split into two consecutive chains, with the final resolvent of the first acting as the input clause of the second, without affecting the final result. By splitting resolution steps whose pivots are not uniformly labelled we can *always* generate a labelled refutation for which Itp is a total function. The example in Figure 7 illustrates this transformation for a single hyper-resolution step.

Each hyper-resolution or TraceCheck resolution step may need to be rewritten into several subsequent uniformly labelled steps, thus changing the proof structure. Note that the results on the relative strength of interpolants in § 3 naturally only apply if both proofs have the same structure. The effect of the order of resolution steps on the strength of interpolants is discussed in [16, § 5.2] and exceeds the scope of this paper.

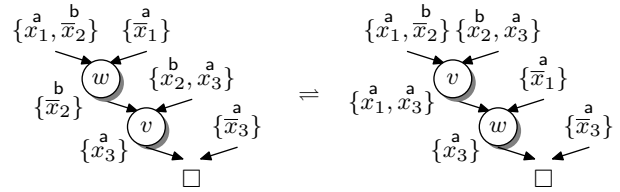


Fig. 8 Reordering resolution chains

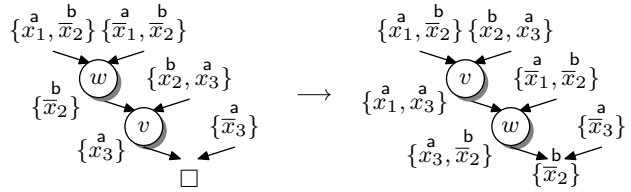


Fig. 9 Reordering in the presence of merge literals may invalidate the resolution chain

The number of resolution steps resulting from splitting depends on the order of the pivots in the given resolution chain, as demonstrated in the following example.

Example 3 Figure 8 shows two resolution chains (presented as trivial resolution proofs). In the left proof, the order of the pivots is x_1^{a}, x_2^{b}, x_3^{a}, necessitating *two* splits to obtain a uniform labelling of the pivots. The proof to the right corresponds to a similar resolution chain in which the first two resolution steps have been swapped. The resulting split yields the following two TraceCheck resolution steps:

$$\frac{(x_1^{\mathsf{a}} \vee \overline{x}_2^{\mathsf{b}}) \quad (x_2^{\mathsf{b}} \vee x_3^{\mathsf{a}})}{(x_1^{\mathsf{a}} \vee x_3^{\mathsf{a}})}\ [B\text{-TCRes}] \qquad \frac{(x_1^{\mathsf{a}} \vee x_3^{\mathsf{a}}) \quad (\overline{x}_1^{\mathsf{a}}) \quad (\overline{x}_3^{\mathsf{a}})}{\square}\ [A\text{-TCRes}]$$

Accordingly, the interpolation system $\mathsf{Itp}(L)$ is applicable to the corresponding clausal proof. ◁

Example 3 shows that reordering the resolution steps in a chain can result in fewer uniformly labelled TraceCheck resolution steps. A *swap* ($\rightleftharpoons$) of two subsequent resolution steps, formally defined in [16, Def. 10] and illustrated in Figure 8, is allowed whenever it does not change the conclusion of the resolution chain. In the presence of merge literals [1] (i.e., literals $t \in \ell(v)$ such that $t \in \ell(v^+)$ and $t \in \ell(v^-)$) this is not guaranteed [16], as illustrated in Figure 9.

The final resolvent of a chain may depend on the order of the ordinary resolution steps: literal $\overline{x}_2$ is re-introduced after being eliminated in the modified chain, while it is *merged* and eliminated once and for all in the original chain.

In the absence of merge literals, this issue does not arise. For this reason, [56] prohibits merge literals in resolution chains (in addition to requiring that the premises match the HyRes rule). While this guarantees that a any permutation of the clause-pivot sequence still represents a valid resolution chain and leaves the final resolvent unaffected (an immediate consequence of [16, Lemma 4]), the requirement is overly restrictive. In the following, we discuss conditions under which reordering does not invalidate the proof even in the presence of merge literals.

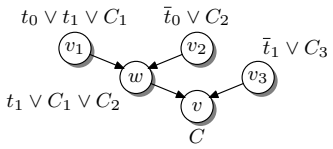


Fig. 10 Proof R

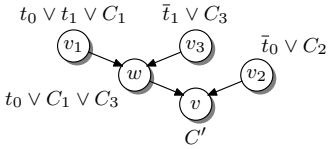


Fig. 11 Graph $R' \stackrel{\text{def}}{=} R[w \rightleftharpoons v]$

Let R and $R' \stackrel{\text{def}}{=} R[w \rightleftharpoons v]$ be as in Figure 10 and Figure 11. According to [16], the clause label $C' = \ell'(v) = \mathrm{Res}(\ell'(w), \ell'(v_2), t_0)$ in Figure 11 differs from $C = \ell(v)$ in Figure 10 in the following two cases:

1. If $t_0 \in C_3$ then $t_0 \in C$, but $t_0 \notin C'$.
2. If $t_1 \in C_2$ then $t_1 \notin C$, but $t_1 \in C'$.

As explained in Remark 2, the former case does not occur in resolution chains generated by CDCL, since resolved literals are never reintroduced. In the second case, however, the swap *introduces* a literal into an (intermediate) resolvent. Since the resolution chain is regular, this literal propagates to the final resolvent of the chain, potentially invalidating the clausal proof.

Instead of prohibiting the transformation in general, however, it is possible to analyse the underlying resolution proof R to determine whether the literal introduced by the transformation is eliminated along all paths to the sink of the proof [3,19,9]. The set of literals eliminated along all paths from $v \in V_R$ to $\mathtt{s}_R$ can be defined as the *meet-over-all-paths* in the terminology of data-flow analysis:

Definition 14 (Safe Literals) Let $R = (V_R, E_R, \ell_R, \mathtt{s}_R)$ be a resolution refutation. The *safe literals* $\sigma(v)$ of a vertex $v \in V_R$ are defined inductively as follows:

$$\mathrm{rlit}(v,w) = t \text{ s.t. } t \in \ell(v), \mathrm{var}(t) = piv(w), \exists u \neq w \,.\, (u,w) \in E_R \wedge \mathrm{rlit}(u,w) = \bar{t}$$

$$\sigma(v) = \begin{cases} \emptyset & \text{if } v = \mathtt{s}_R \\ \bigcap_{(v,w)\in E_R} (\sigma(w) \cup \{\mathrm{rlit}(v,w)\}) & \text{otherwise} \end{cases}$$

A solution to the data-flow equation in Definition 14 can be computed in linear time since the graph R is acyclic. For the proof to the left of Figure 9 we obtain $\sigma(v) = \{x_3\}$ and $\sigma(w) = \{\overline{x}_2, x_3\}$, for instance.

Let v be the final vertex of the trivial resolution derivation that corresponds to a given resolution chain. A swap of two vertices of the chain that introduces a literal t in $\ell(v)$ is admissible iff $t \in \sigma(v)$. Accordingly, the literal t is introduced in the conclusion (final resolvent, respectively) of the chain. The proof remains valid since t is subsequently eliminated.

Example 4 Figure 12 shows a refutation with two chains generated by a CDCL solver, where the vertex p marks the end of the first chain. As in Example 3, the pivot order $\overset{\mathsf{a}}{x_1}, \overset{\mathsf{b}}{x_2}, \overset{\mathsf{a}}{x_3}$ of the first chain enforces a split resulting in the TraceCheck resolution steps on the right side in Figure 12. Similarly to the example in Figure 9, reordering of the vertices w and v results in the introduction of the literal $\overset{\mathsf{b}}{\overline{x}_2}$ in $\ell(p)$. The transformation is safe, however, since $\overline{x}_2 \in \sigma(p)$. The transformation

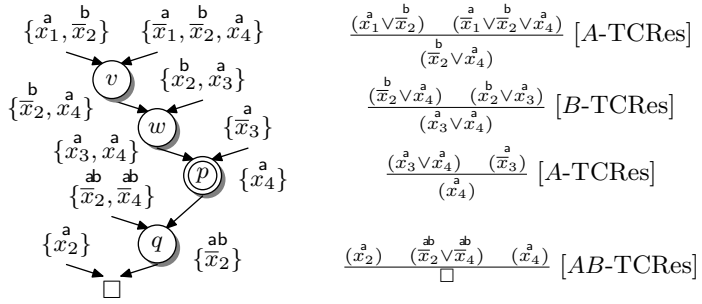


Fig. 12 Two resolution chains and a corresponding clausal proof (after splitting)

yields the following uniformly labelled TraceCheck resolution steps:

$$\frac{(x_1^{\mathsf{a}} \vee \overline{x}_2^{\mathsf{b}}) \quad (x_2^{\mathsf{b}} \vee x_3^{\mathsf{a}})}{(x_1^{\mathsf{a}} \vee x_3^{\mathsf{a}})}\ [B\text{-TCRes}] \qquad \frac{(x_1^{\mathsf{a}} \vee x_3^{\mathsf{a}}) \quad (\overline{x}_1^{\mathsf{a}} \vee \overline{x}_2^{\mathsf{b}} \vee x_4^{\mathsf{a}}) \quad (\overline{x}_3^{\mathsf{a}})}{(\overline{x}_2^{\mathsf{b}} \vee x_4^{\mathsf{a}})}\ [A\text{-TCRes}]$$

$$\frac{(\overline{x}_2^{\mathsf{b}} \vee x_4^{\mathsf{a}}) \quad (\overline{x}_2^{\mathsf{ab}} \vee \overline{x}_4^{\mathsf{ab}}) \quad (x_2^{\mathsf{a}})}{\square}\ [AB\text{-TCRes}]$$

◁

The interpolation system in Definition 13 remains applicable to the transformed clausal proof, since conclusions of TraceCheck resolution steps may always be weakened. The transformation may, however, affect the labelling of the pivots of the subsequent resolution steps. This might be undesirable, if it forces us to split subsequent chains. It is possible to avoid a change of the labelling by computing *safe labels* for the literals in a proof.

Definition 15 (Safe Labels) Given a refutation $R = (V_R, E_R, \ell_R, \mathtt{s}_R)$, the mapping $\varsigma : V_R \times \mathtt{Lit} \to \mathcal{S}$ (where $\mathcal{S} = \{\bot, \mathsf{a}, \mathsf{b}, \mathsf{ab}\}$ as in Definition 4) is defined inductively as follows:

$$\begin{aligned} \operatorname{litlab}(u, v, t) &= \begin{cases} L(v^+, \operatorname{var}(t)) \sqcup L(v^-, \overline{\operatorname{var}(t)}) & \text{if } t = \operatorname{rlit}(u, v) \\ \varsigma(v, t) & \text{otherwise} \end{cases} \\ \varsigma(v, t) &= \begin{cases} \bot & \text{if } v = \mathtt{s}_R \\ \bigsqcap_{(v,w) \in E} \operatorname{litlab}(v, w, t) & \text{otherwise} \end{cases} \end{aligned} \tag{2}$$

Given a vertex $v \in V_R$ and a literal $t \in \ell(v)$, we call $\varsigma(v, t)$ the safe label of t.

The safe labels ς are computed in lockstep with σ (Definition 14). Whenever a literal $t \in \sigma(v)$ introduced into $\ell(v)$ is labelled such that $L(v, t) \sqsubseteq \varsigma(v, t)$, then the labelling of the pivots in the subsequent resolution steps remains unchanged [9].

Example 5 For the resolution refutation in Figure 12 we obtain $\varsigma(q, \overline{x}_2) = \varsigma(p, \overline{x}_2) = \mathsf{ab}$. Swapping the vertices v and w introduces $\overline{x}_2$ in $\ell(p)$ with $L(p, \overline{x}_2) = \mathsf{a}$. Consequently, the labelling of the pivot in the final resolution step is preserved. ◁

The empirical evaluation in the following section motivates the use of interpolation systems for clausal proofs.

4.2 Empirical Results

We implemented the labelled interpolation system for clausal proofs as an extension to the TraceCheck-tool.[2] TraceCheck's original purpose is the verification of the output of SAT solvers, based on proof certificates stored in the TraceCheck-format.

Our interpolation system can be easily incorporated into TraceCheck. The only significant change arises from splitting the resolution chains to establish that $\mathsf{Itp}(L)$ is defined for a given labelling function L, as described in § 4.1. Our implementation currently does not try to reduce the number of splits by means of reordering.

For the experimental evaluation of our implementation, we use benchmarks from reactive synthesis [8] obtained via the interpolation-based relation determinisation technique presented in [31]. We use PicoSAT 957 [5] to obtain clausal proofs in the TraceCheck-format. We limit the proofs to those with a file size between 100kB and 10MB, resulting in 133 benchmarks. We label the literals in A-clauses a and the literals in B-clauses b, which provably results in the introduction of fewer literals than other labellings [15, 9]. All experiments were executed on an Intel Core i5 M560 at 2.67GHz and with 8GB of RAM.

To measure the impact of transforming a clausal proof for labelled interpolation, we look at proofs before (INITIAL) and after (SPLIT) splitting (Figure 13). Using TraceCheck's `-b` option (BINARY), we also compare the clausal interpolation system to the conventional interpolation system for binary resolution proofs (presented in § 2.2).[3] Figure 14 shows the average length of chains before and after splitting. On average, 44.86% of the chains generated by TraceCheck need to be split to enable interpolation (Figure 15).

Figure 16 compares the number of Boolean operations in the interpolants generated by clausal interpolation and binary interpolation. The difference is negligible, since n-ary conjunctions are encoded by binary gates. Figure 17 shows the memory consumption of our interpolation systems (in megabytes). The plot for run-time has a similar shape. The average run-time for SPLIT proofs is 0.9s and 5.49s for BINARY proofs. The quantiles are as follows:

	0%	25%	50%	75%	100%
SPLIT (s)	0.01	0.17	0.52	1.40	4.85
BINARY (s)	0.06	0.63	1.79	5.55	54.09

We use the And-Inverter-Graph (AIG) library AIGER[4] to store interpolants. The library performs trivial simplifications and structural hashing to keep the circuit size small. The graph on the left of Figure 18 shows that the interpolants extracted from clausal proofs are consistently smaller than interpolants generated by the conventional interpolation technique.

Finally, we use ABC [12] to gather statistics about the interpolant and to reduce the circuit size further with the following commands: `strash; balance; fraig; refactor -z; rewrite -z; fraig;`. After reduction, the sizes of the in-

[2] `http://fmv.jku.at/booleforce/`

[3] Note that this transformation affects the whole proof, resulting in high memory usage. TCRes offers a natural way to compute interpolants for resolution chains without intermediate clauses. Alternatively, one could apply ordinary resolution iteratively on resolution chains and retain only partial interpolants at the end of a chain. We did not experimentally evaluate the latter approach.

[4] `http://fmv.jku.at/aiger/`

terpolants extracted from clausal proofs and from binary proofs are similar. We emphasise that interpolation based on clausal proofs is superior with respect to memory consumption and the intermediate size of interpolants.

5 Local Refutations and Hyper-Resolution

Jhala and McMillan demonstrate in [30, Theorem 3] that the applicability of propositional interpolation systems is not restricted to propositional logic. If a first-order refutation R has a certain structure, namely if for each inference step in R the antecedents as well as the conclusion are either entirely in $\mathcal{L}(A)$ or in $\mathcal{L}(B)$, then one can use a propositional interpolation system (such as the ones in § 2.2 and § 3) to construct an interpolant that is a Boolean combination of the formulae in R. Kovács and Voronkov subsequently arrived at a similar result [32].

We recapitulate the results from [30,32] before we proceed to show that our interpolation system from Definition 9 generalises the system of [32] as well as a variation of [32] presented in [55].

Definition 16 (Local Refutation) An (A, B)-refutation R in a given inference system for first-order logic is *local* if there exists a total *partitioning* function $\pi_R : V_R \to \{A, B\}$ such that for all edges $(v_1, v_2) \in E_R$ we have $\ell_R(v_1), \ell_R(v_2) \in \mathcal{L}(\pi_R(v_2))$.

While proofs in general do *not* have this property, there is a variety of decision procedures that yield local (ground) refutations. The construction of local proofs is addressed in [30,41,20,32], to name only a few.

The following operation, which resembles the constructions in [32, Lemma 8], [30, Theorem 3], and [20, Section 5.5]), extracts a premise in $\mathcal{L}(A)$ ($\mathcal{L}(B)$, respectively) for a vertex $v \in V_R$ with $\pi(v) = A$ ($\pi(v) = B$, respectively) from a local refutation R.

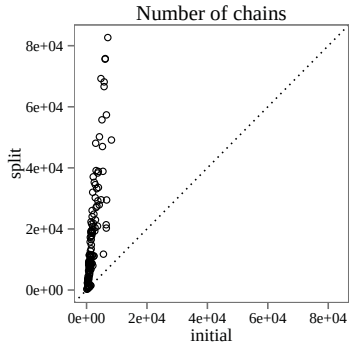
Definition 17 (A-Premise, B-Premise) Let R be a local (A, B)-refutation with partitioning function π, and let $v \in V_R$ such that $\pi(v) = A$. Then

$$
\begin{aligned}
&A\text{-premise}\,(v) \stackrel{\text{def}}{=} \\
&\quad \{u \,|\, (u, v) \in E_R \text{ and } \pi(u) = B \text{ or } u \text{ is initial}\} \cup \\
&\quad \bigcup \{A\text{-premise}\,(u) \,|\, (u, v) \in E_R \text{ and } \pi(u) = A\}\,.
\end{aligned}
$$

B-premise(v) is defined analogously.

Intuitively, A-premise(v) comprises the leaves of the largest sub-derivation S rooted at v such that $\pi(u) = A$ for all internal vertices $u \in V_S$.[5] If the underlying inference system is sound, we have $\{\ell(u) \,|\, u \in A\text{-premise}(v)\} \models \ell(v)$. If, moreover, $\ell(v)$ as well as all formulae of A-premise(v) are *closed*, we make the following observation (c.f. related results in [32, Lemma 1] and [20, Lemma 3]):

[5] In particular, it is possible to choose π_R in such a manner that S is the largest sub-derivation rooted at v in R such that $\ell_R(u) \in \mathcal{L}(A)$ for all $u \in V_S$. This corresponds to the setting in [32, Lemma 8].



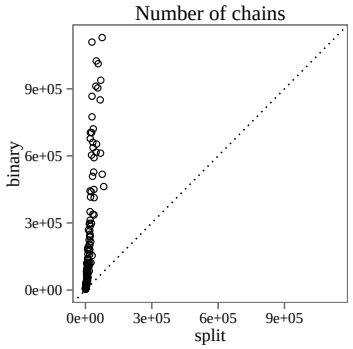


Fig. 13 Number of chains before and after splitting; binary resolution steps

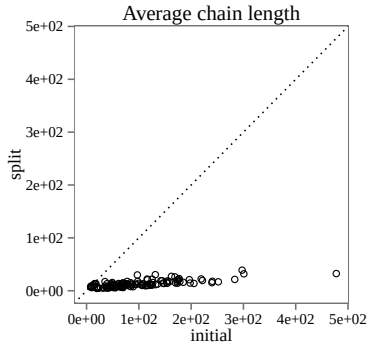


Fig. 14 Average chain length

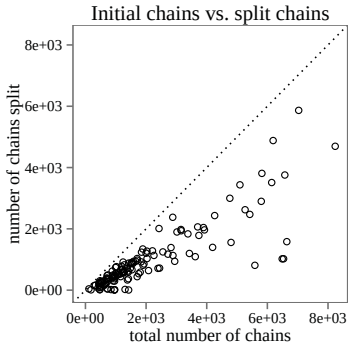


Fig. 15 Chains that need to be split (46%)

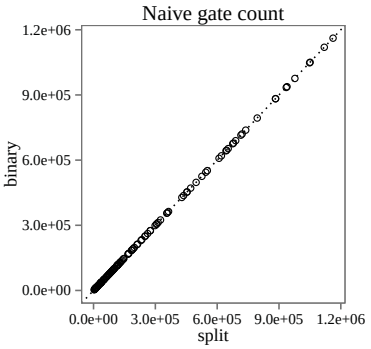


Fig. 16 Number of gates before reduction

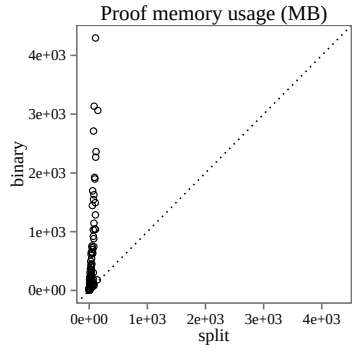
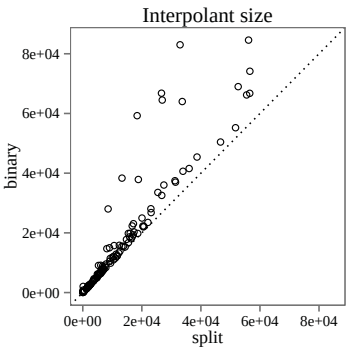


Fig. 17 Memory usage



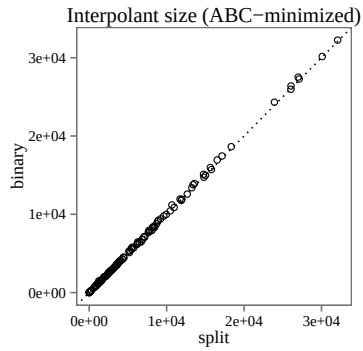


Fig. 18 Size of AIG circuit; before and after reduction by ABC

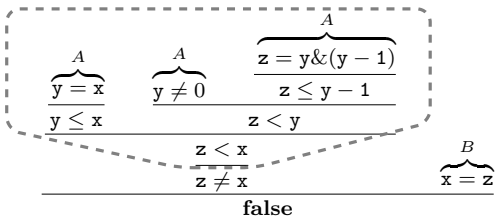


Fig. 19 Refutation of $(\mathtt{y} = \mathtt{x}) \wedge (\mathtt{y} \neq 0) \wedge (\mathtt{z} = \mathtt{y}\&(\mathtt{y}-1)) \ \wedge \ (\mathtt{x} = \mathtt{z})$; A-premise of $\mathtt{z} < \mathtt{x}$

Corollary 1 *Let R be a local closed refutation in a sound inference system, and let $v \in V_R$ an internal vertex such that $\pi_R(v) = A$. Then, the following Horn clause is a tautology:*

$$\bigvee_{u \in A\text{-}premise(v)} \neg \ell_R(u) \vee \ell_R(v) \tag{3}$$

A similar claim holds for the case in which $\pi(v) = B$.

Corollary 1 is a pivotal element in our proof of the following theorem:

Theorem 5 *(c.f. [30, Theorem 3]) Let R be a closed local (A, B)-refutation in a sound inference system. Then one can extract a Craig interpolant from R using a propositional interpolation system.*

Proof: Let $v \in V_R$ be such that $\pi(v) = A$. If v is initial, then either A or B contains the unit clause $C_v = \ell(v)$. Otherwise, according to Corollary 1, the clause $C_v = (\{\neg\ell(u) \,|\, u \in A\text{-premise}(v)\} \vee \ell(v))$ is tautological (and therefore implied by A). Moreover, it follows from Definition 16 that if $u \in A$-premise(v) is not an initial vertex of R then $\ell_R(u) \in \mathcal{L}(A) \cap \mathcal{L}(B)$ holds. Accordingly, $C_v \in \mathcal{L}(A)$, and we add C_v to A. A similar argument holds for $v \in V_R$ with $\pi(v) = B$.

By construction, the resulting set of clauses C_v, $v \in V_R$, is propositionally unsatisfiable [30,32]; also, each clause is implied by either A or B. Moreover, all literals with $t \in \mathcal{L}(A) \setminus \mathcal{L}(B)$ ($t \in \mathcal{L}(B) \setminus \mathcal{L}(A)$, respectively) are local to A (B, respectively). Accordingly, it is possible to construct an interpolant for (A, B) using the interpolation systems presented in § 2.2 and § 3. ■

Example 6 Figure 19 shows an (A, B)-refutation for $A \equiv (\mathtt{y} = \mathtt{x}) \wedge (\mathtt{y} \neq 0) \wedge (\mathtt{z} = \mathtt{y}\&(\mathtt{y}-1))$ and $B \equiv (\mathtt{x} = \mathtt{z})$, where $\mathtt{x}, \mathtt{y}, \mathtt{z}$ are bit-vectors and & denotes bit-wise conjunction. Let vertex v be such that $\ell(v) = (\mathtt{z} < \mathtt{x})$ and $\pi(v) = A$. The dashed line in Figure 19 indicates the sub-proof rooted at v, whose leaves constitute the A-premise of v. Following the construction in the proof of Theorem 5, we obtain the following hyper-resolution step with conclusion $\ell(v)$.

$$\frac{(\mathtt{y} = \mathtt{x}) \quad (\mathtt{y} \neq 0) \quad (\mathtt{z} = \mathtt{y}\&(\mathtt{y}-1)) \quad \overbrace{\left(\overline{(\mathtt{y} = \mathtt{x})} \vee \overline{(\mathtt{y} \neq 0)} \vee \overline{(\mathtt{z} = \mathtt{y}\&(\mathtt{y}-1))} \vee (\mathtt{z}{<}\mathtt{x})\right)}^{\text{tautology in } \mathcal{L}(A)}}{\mathtt{z}{<}\mathtt{x}}$$

Consider the vertex w with $\ell(w) = (\mathtt{z} \neq \mathtt{x})$ and $\pi(w) = B$. The corresponding B-premise is $\{v\}$, resulting in the resolution step $\mathrm{Res}(\{(\mathtt{z} < \mathtt{x})\}, \{\neg(\mathtt{z} < \mathtt{x}), (\mathtt{z} \neq \mathtt{x})\}, (\mathtt{z} < \mathtt{x}))$ with conclusion $(\mathtt{z} \neq \mathtt{x})$. ◁

Kovács and Voronkov avoid the explicit construction of a resolution proof by defining their interpolation system directly on the local proof [32, Theorem 11]:

For an initial vertex v

(A-clause) $\dfrac{}{\ell(v) \quad [\ell(v)]}$ if $\ell(v) \in A$ (B-clause) $\dfrac{}{\ell(v) \quad [\neg\ell(v)]}$ if $\ell(v) \in B$

For an internal vertex v with $\{v_1, \ldots, v_n\} = \pi(v)$-premise$(v)$ such that
$\ell(v_i) \in \mathcal{L}(A) \cap \mathcal{L}(B)$ for $1 \leq i \leq m \leq n$ and
$\ell(v_j) \notin \mathcal{L}(A) \cap \mathcal{L}(B)$ for $m < j \leq n$.

$$\frac{\ell(v_1) \quad [I_1] \quad \cdots \quad \ell(v_m) \quad [I_m] \qquad \ell(v_{m+1}) \quad \cdots \quad \ell(v_n)}{\ell(v) \quad [I]}$$

(A-justified) if $\pi(v) = A$, $I \stackrel{\text{def}}{=} \bigwedge_{i=1}^{m}(\ell(v_i) \vee I_i) \wedge \bigvee_{i=1}^{m} \neg\ell(v_i)$

(B-justified) if $\pi(v) = B$, $I \stackrel{\text{def}}{=} \bigwedge_{i=1}^{m}(\ell(v_i) \vee I_i)$

Fig. 20 Interpolation system Itp_{KV} for local proofs

Definition 18 Let R be a local and closed (A, B)-refutation. The interpolation system Itp_{KV} maps vertices $v \in V_R$, for which $\ell_R(v) \in \mathcal{L}(A) \cap \mathcal{L}(B)$ holds, to partial interpolants as defined in Figure 20.

Remark. In addition to the condition in Definition 16, Kovács and Voronkov require that for each $v \in V_R$ with predecessors $v_1, \ldots, v_n$, $\ell(v) \in \mathcal{L}(A) \cap \mathcal{L}(B)$ if $\ell(v_i) \in \mathcal{L}(A) \cap \mathcal{L}(B)$ for all $i \in \{1..n\}$. A local derivation satisfying this condition is *symbol-eliminating*, i.e., it does not introduce "irrelevant" symbols. This technical detail allows the leaves of R to be merely implied by A (or B) instead of being actual elements of A (B, respectively), while preserving the correctness of the interpolation system. This effectively enables interpolation for *non-closed* formulae (A, B).

We proceed to show one of the main results of this paper, namely that our interpolation system Itp from Definition 9 is able to simulate the interpolation system Itp_{KV}.

Theorem 6 *Let R be a local and closed (A, B)-refutation. Then we can construct a hyper-resolution refutation H of (A, B) and a locality preserving labelling function L such that for each $v \in V_R$ with $\ell_R(v) \in \mathcal{L}(A) \cap \mathcal{L}(B)$ there exists a corresponding vertex $u \in V_H$ such that $\mathsf{Itp}_{KV}(R)(v) \Leftrightarrow \mathsf{Itp}_1(L, H)(u)$.*

Proof sketch: We demonstrate that it is possible to construct a hyper-resolution refutation H of (A, B) in which each internal step of Itp_{KV} is simulated using *two* hyper-resolution steps. The induction hypothesis is that for each internal vertex $v \in V_R$ with $\{v_1, \ldots, v_n\} = \pi(v)$-premise$(v)$ and m as in Definition 18, we have vertices $\{u_1, \ldots, u_n\} \subseteq V_H$ such that

1. $\ell_H(u_i) = \ell_R(v_i)$ for $1 \leq i \leq n$, and
2. $\mathsf{Itp}_1(L, H)(u_i) \Leftrightarrow \mathsf{Itp}_{KV}(R)(v_i)$ for $1 \leq i \leq m$, and
3. $\mathsf{Itp}_1(L, H)(u_j) = \begin{cases} \mathsf{F} \text{ if } \ell(v_j) \in A \\ \mathsf{T} \text{ if } \ell(v_j) \in B \end{cases}$ for $m < j \leq n$.

We add an auxiliary vertex labelled with the clause $\neg\ell_H(u_1) \vee \cdots \vee \neg\ell_H(u_n) \vee \ell_R(v)$, which, by Corollary 1 and by Definition 16, can be regarded as element of

formula $\pi(v)$ (see proof of Theorem 5). The first hyper-resolution step eliminates the literals local to $\pi(v)$; the interpolants and labels are indicated for $\pi(v) = A$:

$$\frac{\overset{\mathsf{a}}{\ell_H(u_{m+1})}\ [\mathsf{F}]\ \cdots\ \overset{\mathsf{a}}{\ell_H(u_n)}\ [\mathsf{F}] \qquad (\neg\overset{\mathsf{a}}{\ell_H(u_{m+1})} \vee \cdots \vee \neg\overset{\mathsf{a}}{\ell_H(u_n)} \vee \cdots \vee \overset{\mathsf{a}}{\ell_R(v)})\ [\mathsf{F}]}{(\neg\overset{\mathsf{ab}}{\ell_H(u_1)} \vee \cdots \vee \neg\overset{\mathsf{ab}}{\ell_H(u_m)} \vee \overset{\mathsf{a}}{\ell_R(v)}) \quad [\mathsf{F}]}$$

The second hyper-resolution step eliminates the shared literals $\ell_H(u_i)$ (for $1 \leq i \leq m$). Again, the labels and interpolants are for the case that $\pi(v) = A$:

$$\frac{\ell_H(u_1)\,[I_1]\ \cdots\ \ell_H(u_m)\,[I_m] \qquad (\neg\overset{\mathsf{ab}}{\ell_H(u_1)} \vee \cdots \vee \neg\overset{\mathsf{ab}}{\ell_H(u_m)} \vee \overset{\mathsf{a}}{\ell_R(v)})\ [\mathsf{F}]}{\overset{\mathsf{a}}{\ell_R(v)} \quad [\bigwedge_{i=1}^{m}(\ell_H(u_i) \vee I_i) \wedge (\mathsf{F} \vee \bigvee_{i=1}^{m} \neg\ell_H(u_i))]}$$

The sink of this resolution step is the vertex $u \in V_H$ such that $\ell_H(u) = \ell_R(v)$ and $\mathsf{Itp}_1(L, H)(u) = \mathsf{Itp}_{KV}(v)$. ■

We proceed to show that our system for hyper-resolution also generalises another existing interpolation system for local refutations. In [55], we introduced the following variation of the interpolation system in Definition 18:

Definition 19 Let Itp_W be the interpolation system as described in Definition 18, except for the following modification:

(A-justified)	if $\pi(v) = A$,	$I \overset{\text{def}}{=}$	$\bigvee_{i=1}^{m}(\neg\ell(v_i) \wedge I_i)$
(B-justified)	if $\pi(v) = B$,	$I \overset{\text{def}}{=}$	$\bigvee_{i=1}^{m}(\neg\ell(v_i) \wedge I_i) \vee \bigwedge_{i=1}^{m}\ell(v_i)$

The following theorem states that the interpolation system in Definition 9 is powerful enough to simulate Itp_W.

Theorem 7 *Let R be a local and closed (A, B)-refutation. Then we can construct a hyper-resolution refutation H of (A, B) and a locality preserving labelling function L such that for each $v \in V_R$ with $\ell_R(v) \in \mathcal{L}(A) \cap \mathcal{L}(B)$ there exists a corresponding vertex $u \in V_H$ such that $\mathsf{Itp}_W(R)(v) \Leftrightarrow \mathsf{Itp}_2(L, H)(u)$.*

The proof is essentially equivalent to the proof of Theorem 6. Moreover, as a consequence of Theorem 2, Itp_{KV} is *stronger* than Itp_W.

Corollary 2 *Let R be a closed local (A, B)-refutation in a sound inference system. Then $\mathsf{Itp}_{KV}(R) \Rightarrow \mathsf{Itp}_W(R)$.*

6 Related Work

There is a vastly growing number of different interpolation techniques; a recent survey of interpolation in decision procedures is provided by [10]. An exposition of interpolation techniques for SMT solvers can be found in [13]. The work of Yorsh and Musuvathi [58] enables the combination of theory-specific and propositional interpolation techniques [28, 33, 42, 39, 16].

The novel interpolation system presented in Section 3 extends our prior work on propositional interpolation systems [16]. The idea of using labelling functions

(initially introduced in [50] in the context of LTL vacuity detection to determine the *peripherality* of variables in resolution proofs) is common to both approaches. In [16], the partial interpolants are determined by the labelling of the literals in the initial vertices, while the system presented in Section 3 adds an additional degree of freedom by allowing us to make a choice at each internal node.

Recent work by Vizel and Gurfinkel [24] addresses the construction of interpolants from clausal/DRUP proofs (whose size is reduced by means of trimming [25]). Their interpolation system splits partial interpolants into two components, one of which is kept in conjunctive normal form (CNF). Their algorithm restructures the DRUP proof on-the-fly in order to increase the size of the component kept in CNF. Earlier work by Vizel et al. [53] targets the construction of interpolants in CNF by first constructing an over-approximation of an interpolant, which is then refined using inductive strengthening [11].

There is a number of techniques to reduce the size of resolution proofs [3,19, 9]. These techniques target binary resolution proofs, however. The combination of labelled interpolation systems for binary resolution proofs and proof reduction has also been studied extensively by Rollini et al. [44,45].

A number of interpolation techniques rely on local proofs (e.g., [30,41,20,32, 36]). Not all interpolation techniques are based on local proofs, though: McMillan's interpolating inference system for equality logic with uninterpreted functions and linear arithmetic [40], for instance, performs an implicit conversion of the proof. In [35], propositional proofs of bit-vector formulas are lifted to proofs in equality logic. The approach presented in [47] avoids the construction of proofs altogether and handles theory combination by reduction to a base theory as in [51] or [52]. InterHorn [23] extracts interpolants from first-order resolution proofs generated by a Horn-clause solver. Sharma et al. show how to compute interpolants without proofs using machine learning techniques [48].

Hoder et al. [26] present a technique that enables the variation of interpolants by fine-tuning the partitioning in Definition 16. In Example 6, for instance, changing $\pi(w) = B$ to $\pi(w) = A$ results in propositional proof that does not contain the literal ($\mathtt{z}<\mathtt{x}$). Accordingly, the term does not occur in the resulting interpolant. This approach can be combined with our interpolation system in a straight forward manner.

An extension of [16] to sequence interpolants is presented in [46]. A survey of interpolation-based model checking techniques is provided in [54]. Interpolation-based synthesis is discussed in [31,27]. Other applications of interpolation algorithms include fault localization [59] and error explanation [18,57], where the quality of interpolants can impact the utility of the diagnosis.

7 Consequences and Conclusion

We present a novel interpolation system for hyper-resolution proofs which generalises our previous work [16]. We subsequently generalise this interpolation system to clausal proofs, generated by contemporary SAT solvers. By defining a rule that addresses hyper-resolution or clausal resolution steps (introduced by preprocessing [21] or extracted from resolution chains), we avoid the construction of intermediate partial interpolants, resulting in reduced memory consumption and smaller intermediate interpolants. As future work, we will investigate whether

proof restructuring [24] and heuristics based on proof analysis [9] can result in a further reduction of splitting.

By applying our technique to local proofs, we combine a number of first-order [32, 55] and propositional interpolation techniques [28, 33, 42, 39] into one *uniform* interpolation approach. As in [30], our approach avoids an explicit theory combination step [58]. Therefore, it enables the variation of interpolant strength and the elimination of non-essential literals across the theory boundary.

Acknowledgements. We would like to thank Armin Biere and his co-authors for providing TraceCheck and AIGER as open source software under a permissive license. We thank Adrián Rebola-Pardo for his helpful comments.

References

1. P. B. Andrews. Resolution with merging. *J. ACM*, 15(3):367–381, 1968.
2. F. Bacchus. Enhancing Davis Putnam with extended binary clause reasoning. In *IAAI*, pages 613–619. AAAI Press / MIT Press, 2002.
3. O. Bar-Ilan, O. Fuhrmann, S. Hoory, O. Shacham, and O. Strichman. Linear-time reductions of resolution proofs. Technical Report IE/IS-2008-02, Technion, 2008.
4. P. Beame, H. Kautz, and A. Sabharwal. Towards understanding and harnessing the potential of clause learning. *Journal of Artificial Intelligence Research*, 22(1):319–351, Dec. 2004.
5. A. Biere. PicoSAT essentials. *JSAT*, 4(2-4):75–97, 2008.
6. A. Biere, M. J. H. Heule, H. van Maaren, and T. Walsh, editors. *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, February 2009.
7. R. Bloem, S. Galler, B. Jobstmann, N. Piterman, A. Pnueli, and M. Weiglhofer. Specify, compile, run: Hardware from psl. *Electronic Notes in Theoretical Computer Science*, 190(4):3–16, 2007.
8. R. Bloem, R. Könighofer, and M. Seidl. Sat-based synthesis methods for safety specs. In K. McMillan and X. Rival, editors, *VMCAI*, volume 8318 of *LNCS*, pages 1–20. Springer Berlin Heidelberg, 2014.
9. R. Bloem, S. Malik, M. Schlaipfer, and G. Weissenbacher. Reduction of resolution refutations and interpolants via subsumption. In *Haifa Verification Conference*, page 188. Springer, 2014.
10. M. P. Bonacina and M. Johansson. On interpolation in decision procedures. In *TABLEAUX*, volume 6793 of *LNCS*, pages 1–16. Springer, 2011.
11. A. R. Bradley. SAT-based model checking without unrolling. In *VMCAI*, volume 6538 of *LNCS*, pages 70–87. Springer, 2011.
12. R. Brayton and A. Mishchenko. ABC: An academic industrial-strength verification tool. In *CAV*, volume 6174 of *LNCS*, pages 24–40. Springer, 2010.
13. A. Cimatti, A. Griggio, and R. Sebastiani. Efficient generation of Craig interpolants in satisfiability modulo theories. *TOCL*, 2010.
14. W. Craig. Linear reasoning. A new form of the Herbrand-Gentzen theorem. *J. Symbolic Logic*, 22(3):250–268, 1957.
15. V. D'Silva. Propositional interpolation and abstract interpretation. In *European Symposium on Programming*, volume 6012 of *LNCS*. Springer, 2010.
16. V. D'Silva, D. Kroening, M. Purandare, and G. Weissenbacher. Interpolant strength. In *VMCAI*, volume 5944 of *LNCS*, pages 129–145. Springer, 2010.
17. N. Eén and N. Sörensson. An extensible SAT-solver. In *SAT*, volume 2919, pages 502–518. Springer, 2004.
18. E. Ermis, M. Schäf, and T. Wies. Error invariants. In *Formal Methods*, volume 7436 of *LNCS*, pages 187–201. Springer, 2012.
19. P. Fontaine, S. Merz, and B. W. Paleo. Compression of propositional resolution proofs via partial regularization. In *CADE*, volume 6803 of *LNCS*. Springer, 2011.
20. A. Fuchs, A. Goel, J. Grundy, S. Krstić, and C. Tinelli. Ground interpolation for the theory of equality. In *TACAS*, volume 5005 of *LNCS*, pages 413–427. Springer, 2009.

21. R. Gershman and O. Strichman. Cost-effective hyper-resolution for preprocessing cnf formulas. In *SAT*, volume 3569 of *LNCS*, pages 423–429. Springer, 2005.
22. E. Goldberg and Y. Novikov. Verification of proofs of unsatisfiability for CNF formulas. In *DATE*, pages 886–891. IEEE, 2003.
23. A. Gupta, C. Popeea, and A. Rybalchenko. Generalised interpolation by solving recursion-free Horn clauses. *CoRR*, abs/1303.7378, 2013.
24. A. Gurfinkel and Y. Vizel. Druping for interpolants. In *Formal Methods in Computer-Aided Design*, pages 99–106. FMCAD Inc, 2014.
25. M. Heule, W. A. H. Jr., and N. Wetzler. Trimming while checking clausal proofs. In *Formal Methods in Computer-Aided Design*, pages 181–188. IEEE, 2013.
26. K. Hoder, L. Kovács, and A. Voronkov. Playing in the grey area of proofs. In *Principles of Programming Languages*, pages 259–272. ACM, 2012.
27. G. Hofferek, A. Gupta, B. Könighofer, J. R. Jiang, and R. Bloem. Synthesizing multiple boolean functions using interpolation on a single proof. In *Formal Methods in Computer-Aided Design*, pages 77–84. IEEE, 2013.
28. G. Huang. Constructing Craig interpolation formulas. In *Computing and Combinatorics*, volume 959 of *LNCS*, pages 181–190. Springer, 1995.
29. R. Jhala and K. L. McMillan. Interpolant-based transition relation approximation. In *CAV*, volume 3576 of *LNCS*, pages 39–51. Springer, 2005.
30. R. Jhala and K. L. McMillan. A practical and complete approach to predicate refinement. In *TACAS*, volume 3920 of *LNCS*, pages 459–473. Springer, 2006.
31. J.-H. R. Jiang, H.-P. Lin, and W.-L. Hung. Interpolating functions from large Boolean relations. In *ICCAD*, pages 779–784. ACM, 2009.
32. L. Kovács and A. Voronkov. Interpolation and symbol elimination. In *CADE*, volume 5663 of *LNCS*, pages 199–213. Springer, 2009.
33. J. Krajíček. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *J. Symbolic Logic*, 62(2):457–486, 1997.
34. D. Kroening and O. Strichman. *Decision Procedures: An Algorithmic Point of View*. Texts in Theoretical Computer Science. Springer, 2008.
35. D. Kroening and G. Weissenbacher. Lifting propositional interpolants to the word-level. In *Formal Methods in Computer-Aided Design*, pages 85–89. IEEE, 2007.
36. D. Kroening and G. Weissenbacher. An interpolating decision procedure for transitive relations with uninterpreted functions. In *Haifa Verification Conference*, volume 6405 of *LNCS*, pages 150–168. Springer, 2011.
37. S. Maehara. On the interpolation theorem of Craig. *Sûgaku*, 12:235–237, 1961.
38. S. Malik and G. Weissenbacher. Boolean satisfiability solvers: techniques and extensions. In *Software Safety and Security - Tools for Analysis and Verification*, NATO Science for Peace and Security Series. IOS Press, 2012.
39. K. L. McMillan. Interpolation and SAT-based model checking. In *CAV*, volume 2725 of *LNCS*, pages 1–13. Springer, 2003.
40. K. L. McMillan. An interpolating theorem prover. *Theoretical Comput. Sci.*, 345(1):101–121, 2005.
41. K. L. McMillan. Quantified invariant generation using an interpolating saturation prover. In *TACAS*, volume 4963 of *LNCS*, pages 413–427. Springer, 2008.
42. P. Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *J. Symbolic Logic*, 62(3):981–998, 1997.
43. J. Robinson. Automatic deduction with hyper-resolution. *J. Comp. Math.*, 1, 1965.
44. S. F. Rollini, L. Alt, G. Fedyukovich, A. E. J. Hyvärinen, and N. Sharygina. PeRIPLO: A framework for producing effective interpolants in SAT-based software verification. In *Logic for Programming, Artificial Intelligence, and Reasoning (LPAR)*, volume 8312 of *LNCS*, pages 683–693. Springer, 2013.
45. S. F. Rollini, R. Bruttomesso, N. Sharygina, and A. Tsitovich. Resolution proof transformation for compression and interpolation. *Formal Methods in System Design*, 45(1):1–41, 2014.
46. S. F. Rollini, O. Sery, and N. Sharygina. Leveraging interpolant strength in model checking. In *CAV*, volume 7358 of *LNCS*, pages 193–209. Springer, 2012.
47. A. Rybalchenko and V. Sofronie-Stokkermans. Constraint solving for interpolation. In *VMCAI*, volume 4349 of *LNCS*, pages 346–362. Springer, 2007.
48. R. Sharma, A. Nori, and A. Aiken. Interpolants as classifiers. In P. Madhusudan and S. Seshia, editors, *CAV*, volume 7358 of *LNCS*, pages 71–87. Springer Berlin Heidelberg, 2012.

49. J. P. M. Silva and K. A. Sakallah. GRASP - a new search algorithm for satisfiability. In *ICCAD*, pages 220–227, 1996.
50. J. Simmonds, J. Davies, A. Gurfinkel, and M. Chechik. Exploiting resolution proofs to speed up LTL vacuity detection for BMC. *STTT*, 12(5):319–335, 2010.
51. V. Sofronie-Stokkermans. Interpolation in local theory extensions. In *Automated Reasoning*, pages 235–250. Springer, 2006.
52. N. Totla and T. Wies. Complete instantiation-based interpolation. In *Principles of Programming Languages*, pages 537–548, New York, NY, USA, 2013. ACM.
53. Y. Vizel, V. Ryvchin, and A. Nadel. Efficient generation of small interpolants in CNF. In *CAV*, volume 8044 of *LNCS*, pages 330–346. Springer, 2013.
54. Y. Vizel, G. Weissenbacher, and S. Malik. Boolean satisfiability solvers and their applications in model checking. *Proceedings of the IEEE*, 103(11):2021–2035, November 2015.
55. G. Weissenbacher. *Program Analysis with Interpolants*. PhD thesis, Oxford, 2010.
56. G. Weissenbacher. Interpolant strength revisited. In *SAT*, volume 7317 of *LNCS*, pages 312–326. Springer, 2012.
57. G. Weissenbacher. Explaining heisenbugs. In *Runtime Verification*, volume 9333 of *LNCS*, page XV. Springer, 2015.
58. G. Yorsh and M. Musuvathi. A combination method for generating interpolants. In *CADE*, volume 3632 of *LNCS*, pages 353–368, 2005.
59. C. S. Zhu, G. Weissenbacher, and S. Malik. Silicon fault diagnosis using sequence interpolation with backbones. In *ICCAD*, pages 348–355. IEEE, 2014.

A Proofs

Remark. The *downward projection* of a clause $\ell(v) = C$ at vertex v with respect to $\mathsf{c} \in \mathcal{S}$ is defined as $\ell(v)\lfloor_{\mathsf{c},L} \stackrel{\text{def}}{=} \{t \in \ell(v) \mid L(v,t) \sqsubseteq \mathsf{c}\}$, the respective *upward projection* is $\ell(v)\lceil_{\mathsf{c},L} \stackrel{\text{def}}{=} \{t \in \ell(v) \mid \mathsf{c} \sqsubseteq L(v,t)\}$ (cf. § 2.2). It follows from condition 1 in Definition 4 that $\ell(v)\downarrow_{\perp,L} = \emptyset$ for all vertices v. Therefore, the following two equalities hold for any clause $C = \ell(v)$ in a refutation R:

- $C\lfloor_{\mathsf{b},L} = (C\lceil_{\mathsf{b},L} \setminus C\lceil_{\mathsf{a},L})$
- $C\lfloor_{\mathsf{a},L} = (C\lceil_{\mathsf{a},L} \setminus C\lceil_{\mathsf{b},L})$

We make repeated use of these equalities in this section. Moreover, our proofs use the following propositions:

Proposition 1 *The implication*

$$\left(\bigwedge_{i=1}^{n}(x_i \vee I_i) \wedge \left(I_{n+1} \vee \bigvee_{i=1}^{n}\overline{x}_i\right)\right) \Rightarrow \left(\bigvee_{i=1}^{n}(\overline{x}_i \wedge I_i) \vee \left(I_{n+1} \wedge \bigwedge_{i=1}^{n} x_i\right)\right)$$

is a tautology.

Proof: This follows from the fact that the conjunction

$$\left(\bigwedge_{i=1}^{n}(x_i \vee I_i) \wedge \left(I_{n+1} \vee \bigvee_{i=1}^{n}\overline{x}_i\right)\right) \wedge \left(\bigwedge_{i=1}^{n}(x_i \vee \neg I_i) \wedge \left(\neg I_{n+1} \vee \bigvee_{i=1}^{n}\overline{x}_i\right)\right)$$

is unsatisfiable (by hyper-resolution). Note that the implication in the other direction does not hold; a simple counterexample for the case $n = 2$ is the assignment $x_1 = x_2 = \mathsf{F}$, $I_1 = \mathsf{T}$, and $I_2 = I_3 = \mathsf{F}$. ∎

Proposition 2 *Let $\bigwedge_{i=1}^{n} P_i$ be unsatisfiable. Then the implication*

$$\left(\bigwedge_{i=1}^{n}(I_i \vee P_i)\right) \Rightarrow \left(\bigvee_{i=1}^{n}(I_i \wedge \neg P_i)\right)$$

is a tautology.

Proof: This follows from the fact that

$$\bigwedge_{i=1}^{n}(I_i \vee P_i) \wedge \bigwedge_{i=1}^{n}(\neg I_i \vee P_i) \quad \equiv \quad \bigwedge_{i=1}^{n}(I_i \vee P_i) \wedge (\neg I_i \vee P_i)$$

is unsatisfiable (since $\mathrm{Res}((I_i \vee P_i), (\neg I_i \vee P_i), I_i) = P_i$ and $\bigwedge_{i=1}^{n} P_i$ is unsatisfiable). ∎

Theorem 1 (Correctness) *For any (A,B)-refutation R (where R is a hyper-resolution proof) and locality preserving labelling function L, $\mathsf{Itp}(L,R)$ (if defined) is an interpolant for (A,B).*

Proof: By induction over the structure of the (A,B)-refutation R. Let I be the partial interpolant at a vertex v labelled with a clause $C = \ell(v)$. We show that every such I and C satisfy the following conditions:

1. $A \wedge \neg(C\lceil_{\mathsf{a},L}) \Rightarrow I$,
2. $B \wedge \neg(C\lceil_{\mathsf{b},L}) \Rightarrow \neg I$, and
3. $\mathrm{Var}(I) \subseteq \mathrm{Var}(A) \cap \mathrm{Var}(B)$.

For the sink v with $\ell(v) = \square$, this establishes Theorem 1. The labelling function L, being unique in this proof, is omitted from subscripts.

Base case. Let v be an initial vertex and let $C = \ell_R(v)$.

1. $C \in A$:
   (a) $A \wedge \neg(C\restriction_{\mathsf{a}}) \Rightarrow C\restriction_{\mathsf{b}}$, is equivalent to $A \Rightarrow (C\restriction_{\mathsf{b}} \setminus C\restriction_{\mathsf{a}} L) \vee C\restriction_{\mathsf{a}}$. This holds because $(C\restriction_{\mathsf{b}} \setminus C\restriction_{\mathsf{a}}) \vee C\restriction_{\mathsf{a}} = C$, and $A \Rightarrow C$ because $C \in A$.

   (b) $B \wedge \neg(C\restriction_{\mathsf{b}}) \Rightarrow \neg(C\restriction_{\mathsf{b}})$, is equivalent to $B \wedge (C\restriction_{\mathsf{b}} \setminus C\restriction_{\mathsf{a}}) \Rightarrow C\restriction_{\mathsf{b}}$. This holds because $(C\restriction_{\mathsf{b}} \setminus C\restriction_{\mathsf{a}}) \subseteq C\restriction_{\mathsf{b}}$ and clauses represent disjunctions.

   (c) For all literals $t \in (C\restriction_{\mathsf{b}} \setminus C\restriction_{\mathsf{a}})$ the following conditions hold:
   - $\mathrm{var}(t) \in \mathrm{Var}(A)$, since $C \in A$.
   - $L(v,t) = \mathsf{b}$. Therefore, by Definition 5, $\mathrm{Var}(t) \in \mathrm{Var}(B)$.

   This establishes that $\mathrm{Var}(C\restriction_{\mathsf{b}} \setminus C\restriction_{\mathsf{a}}) \subseteq \mathrm{Var}(A) \cap \mathrm{Var}(B)$.

2. $C \in B$: Symmetric to $C \in A$.

Induction step. We first prove a useful equality. Let v be an internal vertex of R with ancestors $\{v_1^+, \ldots, v_n^+, v^-\}$. We claim that

$$\bigvee_{i=1}^{n} (\ell(v_i^+) \setminus \{x_i\})\restriction_{\mathsf{c}} \vee (\ell(v^-) \setminus \{\overline{x}_1, \ldots, \overline{x}_n\})\restriction_{\mathsf{c}} = \ell(v)\restriction_{\mathsf{c}} \tag{4}$$

holds for a symbol $\mathsf{c} \in \{\mathsf{a}, \mathsf{b}\}$. This is because for any $t \in \ell(v_i^+)$ (for $1 \leq i \leq n$), if $\mathrm{var}(t) \neq x_i$, then $L(v_i^+, t) \sqsubseteq L(v,t)$. The same holds for $t \in \ell(v^-)$. Thus, if $\mathrm{var}(t) \neq x_i$ and $t \in \ell(u)\restriction_{\mathsf{c}}$ for $u \in \{v_1^+, \ldots, v_n^+, v^-\}$, then $t \in \ell(v)\restriction_{\mathsf{c}}$. Conversely, if $t \in \ell(v)\restriction_{\mathsf{c}}$, then $\mathsf{c} \sqsubseteq L(v,t)$ by the definition of projection. From Definition 5, $L(v,t) = L(v_1^+, t) \sqcup \cdots \sqcup L(v_n^+, t) \sqcup L(v^-, t)$, thus, if $\mathsf{c} \sqsubseteq L(v,t)$ and $\mathsf{c} \neq \mathsf{ab}$, then $\exists i \,.\, \mathsf{c} \sqsubseteq L(v_i^+, t)$ or $\mathsf{c} \sqsubseteq L(v^-, t)$. It follows that $\exists i \,.\, t \in (\ell(v_i^+) \setminus \{x_i\})\restriction_{\mathsf{c}}$ or $t \in (\ell(v^-) \setminus \{\overline{x}_1, \ldots, \overline{x}_n\})\restriction_{\mathsf{c}}$.

For the induction step, let $\ell(v_i^+) = (x_i \vee C_i)$ and $\ell(v^-) = (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee D)$. Due to the requirement that $\mathsf{ltp}(L, R)$ is defined, we may assume that $\forall i \in \{1..n\} \,.\, L(v_i^+, x_i) \sqcup L(v^-, \overline{x}_i) = \mathsf{c}$ (for a fixed c) and perform a case split on c:

1. $\forall i \in \{1..n\} \,.\, L(v_i^+, x_i) \sqcup L(v^-, \overline{x}_i) = \mathsf{a}$:

   Induction hypothesis:

satellites ($i \in \{1..n\}$)	nucleus
$A \wedge \overline{x}_i \wedge \neg(C_i\restriction_{\mathsf{a}}) \Rightarrow I_i$	$A \wedge x_1 \wedge \ldots \wedge x_n \wedge \neg(D\restriction_{\mathsf{a}}) \Rightarrow I_{n+1}$
$B \wedge \neg(C_i\restriction_{\mathsf{b}}) \Rightarrow \neg I_i$	$B \wedge \neg(D\restriction_{\mathsf{b}}) \Rightarrow \neg I_{n+1}$

   It follows that $A \wedge \neg(C_i\restriction_{\mathsf{a}}) \Rightarrow (x_i \vee I_i)$ for $i \in \{1..n\}$ and $A \wedge \neg(D\restriction_{\mathsf{a}}) \Rightarrow (I_{n+1} \vee \bigvee_{i=1}^{n} \overline{x}_i)$, and therefore

$$A \wedge \underbrace{\bigwedge_{i=1}^{n} \neg(C_i\restriction_{\mathsf{a}}) \wedge \neg(D\restriction_{\mathsf{a}})}_{\neg(\bigvee_{i=1}^{n} C_i \vee D)\restriction_{\mathsf{a}},\ \text{by (4)}} \Rightarrow \bigwedge_{i=1}^{n} (x \vee I_i) \wedge (I_{n+1} \vee \bigvee_{i=1}^{n} \overline{x}_i) \,.$$

   By applying hyper-resolution on the right-hand side of the implication we conclude that

$$A \wedge \bigwedge_{i=1}^{n} \neg(C_i\restriction_{\mathsf{a}}) \wedge \neg(D\restriction_{\mathsf{a}}) \Rightarrow \bigvee_{i=1}^{n+1} I_i \,.$$

   Similarly, we derive from the induction hypothesis that

$$A \wedge \neg(\bigvee_{i=1}^{n} C_i \vee D)\restriction_{\mathsf{a}} \Rightarrow \neg I_{n+1} \wedge \bigwedge_{i=1}^{n} \neg I_i \,,$$

and thus

$$A \wedge \neg(\bigvee_{i=1}^{n} C_i \vee D)\restriction_{\mathsf{a}} \Rightarrow \neg \bigvee_{i=1}^{n+1} I_i \,.$$

$\mathrm{Var}(\bigvee_{i=1}^{n+1} I_i) \subseteq \mathrm{Var}(A) \cap \mathrm{Var}(B)$ holds because $\mathrm{Var}(I_i) \subseteq \mathrm{Var}(A) \cap \mathrm{Var}(B)$ for all $i \in \{1..n\}$.

2. $\forall i \in \{1..n\} \,.\, L(v_i^+, x_i) \sqcup L(v^-, \overline{x}_i) = \mathsf{b}$: The proof is symmetric to the first case.

3. $\forall i \in \{1..n\} \,.\, L(v_i^+, x_i) \sqcup L(v^-, \overline{x}_i) = \mathsf{ab}$:

   Induction hypothesis: Let $L(v_i^+, x_i) = \mathsf{a}$ and $L(v^-, \overline{x}_i) = \mathsf{b}$, $1 \leq i \leq n$, for instance. We obtain the following induction hypothesis:

satellites	nucleus
$A \wedge \overline{x}_i \wedge \neg(C_i\restriction_{\mathsf{a}}) \Rightarrow I_i$	$A \wedge \neg(D\restriction_{\mathsf{a}}) \Rightarrow I_{n+1}$
$B \wedge \neg(C_i\restriction_{\mathsf{b}}) \Rightarrow \neg I_i$	$B \wedge x_1 \wedge \ldots \wedge x_n \wedge \neg(D\restriction_{\mathsf{b}}) \Rightarrow \neg I_{n+1}$

   In general, for an arbitrary labelling function L, this can always be extended to the induction hypothesis for $L(v_i^+, x_i) = L(v^-, \overline{x}_i) = \mathsf{ab}$, $1 \leq i \leq n$:

satellites	nucleus
$A \wedge \overline{x}_i \wedge \neg(C_i\restriction_{\mathsf{a}}) \Rightarrow I_i$	$A \wedge x_1 \wedge \ldots \wedge x_n \wedge \neg(D\restriction_{\mathsf{a}}) \Rightarrow I_{n+1}$
$B \wedge \overline{x}_i \wedge \neg(C_i\restriction_{\mathsf{b}}) \Rightarrow \neg I_i$	$B \wedge x_1 \wedge \ldots \wedge x_n \wedge \neg(D\restriction_{\mathsf{b}}) \Rightarrow \neg I_{n+1}$

   From this, it follows immediately that

$$A \wedge \bigwedge_{i=1}^{n} \neg(C_i\restriction_{\mathsf{a}}) \wedge \neg(D\restriction_{\mathsf{a}}) \Rightarrow \bigwedge_{i=1}^{n} (x_i \vee I_i) \wedge (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee I_{n+1}) \,,$$

   and by applying the equality (4) we conclude that

$$A \wedge \neg(\bigvee_{i=1}^{n} C_i \vee D)\restriction_{\mathsf{a}} \Rightarrow \bigwedge_{i=1}^{n} (x_i \vee I_i) \wedge (I_{n+1} \vee \bigvee_{i=1}^{n} \overline{x}_i)$$

   holds. This establishes the first condition for case 1 of AB-HyRes; case 2 is covered by applying Proposition 1.

   Similarly, we derive

$$B \wedge \neg(\bigvee_{i=1}^{n} C_i \vee D)\restriction_{\mathsf{b}} \Rightarrow \bigwedge_{i=1}^{n} (x_i \vee \neg I_i) \wedge (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee \neg I_{n+1}) \,.$$

   Note that this already establishes condition 2 for case 2 of AB-HyRes. By repeated application of resolution, one can further show that the right-hand side of this implication is inconsistent with $\bigwedge_{i=1}^{n}(x_i \vee I_i) \wedge (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee I_{n+1})$. It follows that

$$B \wedge \neg(\bigvee_{i=1}^{n} C_i \vee D)\restriction_{\mathsf{b}} \Rightarrow \neg \left(\bigwedge_{i=1}^{n} (x_i \vee I_i) \wedge (I_{n+1} \vee \bigvee_{i=1}^{n} \overline{x}_i) \right) ,$$

   which covers case 1 of AB-HyRes.

   Note that $x_i \in \mathrm{Var}(A) \cap \mathrm{Var}(B)$ due to $L(v_i^+, x_i) \sqcup L(v^-, \overline{x}_i) = \mathsf{ab}$ (for $1 \leq i \leq n$) and Definition 5, and therefore $\mathrm{Var}(\bigwedge_{i=1}^{n}(x_i \vee I_i) \wedge (I_{n+1} \vee \bigvee_{i=1}^{n} \overline{x}_i)) \subseteq \mathrm{Var}(A) \cap \mathrm{Var}(B)$ holds.

■

Lemma 1 *Let L and L' be labelling functions for an (A, B)-refutation R. If $L(v, t) \preceq L'(v, t)$ holds for all initial vertices v and literals $t \in \ell(v)$, then $L \preceq L'$.*

Proof: We show that $L(v, t) \preceq L'(v, t)$ for all v in R by structural induction.

Base case. If v in R is an initial vertex, $L(v,t) \preceq L'(v,t)$ holds by assumption.

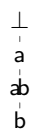
Induction hypothesis: For an internal vertex v and literal t:

satellites $(1 \leq i \leq n)$	nucleus
$L(v_i^+, t) \preceq L'(v_i^+, t)$	$L(v^-, t) \preceq L'(v^-, t)$

Induction step. Let v be an internal vertex in R with ancestors $\{v_1^+, \ldots, v_n^+, v^-\}$, and let $\ell_R(v_i^+) = C_i \vee x_i$ and $\ell_R(v^-) = D \vee \overline{x}_1 \vee \cdots \vee \overline{x}_n$.

We consider two cases:

1. If $t \notin \ell(v)$, then $L(v,t) = L'(v,t) = \bot$.
2. If $t \in \ell(v)$, there are three cases:
   - If $L(v,t) = \mathsf{b}$, then $L(v,t) \preceq L'(v,t)$ because b is the infimum of $(\mathcal{S}, \preceq)$, as indicated in the Hasse diagram to the right.
   - If $L(v,t) = \mathsf{ab}$ then $\mathsf{ab} \preceq L'(v,t)$. If not, $L'(v,t)$ must be b, implying that $L'(v_i^+, t)$ with $1 \leq i \leq n$ and $L'(v^-, t)$ are all b by the definition of $\sqcup$. By the induction hypothesis, we further conclude that for all $i \in \{1..n\}$ it holds that $L(v_i^+, t) = L(v^-, t) = \mathsf{b}$, leading to a contradiction.
   - If $L(v,t) = \mathsf{a}$ then, by the induction hypothesis, $L'(v_i^+, t)$ (where $1 \leq i \leq n$) and $L'(v^-, t)$ are either a or $\bot$. In all cases, the lemma holds.



■

Theorem 3 *If L and L' are labelling functions for an (A,B)-refutation R (R being a hyper-resolution proof) and $L \preceq L'$ such that $\mathsf{ltp}_i(L,R)$ as well as $\mathsf{ltp}_i(L',R)$ are defined, then $\mathsf{ltp}_i(L,R) \Rightarrow \mathsf{ltp}_i(L',R)$ (for a fixed $i \in \{1,2\}$).*

Proof: We prove Theorem 3 by structural induction over R. For any vertex v in R, let I_v and I'_v be the partial interpolants due to $\mathsf{ltp}(L,R)$ and $\mathsf{ltp}(L',R)$, respectively. We show that $I_v \Rightarrow I'_v \vee \ell_R(v)|_{\mathsf{ab},L,L'}$ for all vertices v, where

$$\ell_R(v)|_{\mathsf{ab},L,L'} \stackrel{\text{def}}{=} \{t \in \ell_R(v) \mid L(v,t) \sqcup L'(v,t) = \mathsf{ab}\}\,.$$

This establishes $I_v \Rightarrow I'_v$ for the sink to show that $\mathsf{ltp}(L,R) \Rightarrow \mathsf{ltp}(L',R)$.

Base case. Let v be an initial vertex and let $\ell_R(v) = C$.

1. If $C \in A$, then $I_v = C\lfloor_{\mathsf{b},L}$ and $I'_v = C\lfloor_{\mathsf{b},L'}$. We need to show that $C\lfloor_{\mathsf{b},L} \subseteq C\lfloor_{\mathsf{b},L'} \vee C|_{\mathsf{ab},L,L'}$. For any $t \in C\lfloor_{\mathsf{b},L}$, if $L(v,t) \neq L'(v,t)$ then $L(v,t) \sqcup L'(v,t) = \mathsf{ab}$, and therefore $t \in C|_{\mathsf{ab},L,L'}$. Otherwise, $L(v,t) = L'(v,t) = \mathsf{b}$, and therefore $t \in C\lfloor_{\mathsf{b},L'}$.
2. If $C \in B$, then $I_v = \neg(C\lfloor_{\mathsf{a},L})$ and $I'_v = \neg(C\lfloor_{\mathsf{a},L'})$. The proof that $C\lfloor_{\mathsf{a},L'} \subseteq C\lfloor_{\mathsf{a},L} \vee C|_{\mathsf{ab},L,L'}$ is analogous to the first case.

Induction step. We first prove a useful intermediate result. Let v be an internal vertex of R with ancestors $\{v_1^+, \ldots, v_n^+, v^-\}$. We claim that

$$\left(\bigvee_{i=1}^{n} (\ell(v_i^+) \setminus \{x_i\})|_{\mathsf{ab},L,L'} \vee (\ell(v^-) \setminus \{\overline{x}_1, \ldots, \overline{x}_n\})|_{\mathsf{ab},L,L'}\right) \Rightarrow \ell(v)|_{\mathsf{ab},L,L'} \tag{5}$$

for arbitrary locality preserving labelling functions L and L'. We prove (5) by showing that any t contained in clause of the left-hand side of the implication must also be contained in $\ell(v)|_{\mathsf{ab},L,L'}$. This holds because for any $u \in \{v_1^+, \ldots, v_n^+, v^-\}$ and $t \in \ell(u)$ with $\mathrm{var}(t) \neq x_i$ (for $1 \leq i \leq n$) if $L(u,t) \sqcup L'(u,t) = \mathsf{ab}$ then $L(v,t) \sqcup L'(v,t) = \mathsf{ab}$ since, according to Definition

1, $L(v,t) = L(v_1^+,t) \sqcup \cdots \sqcup L(v_n^+,t) \sqcup L(v^-,t)$ and $L'(v,t) = L'(v_1^+,t) \sqcup \cdots \sqcup L'(v_n^+,t) \sqcup L'(v^-,t)$.

For the induction step, let v be an internal vertex in R and let $\ell_R(v_i^+) = (x_i \vee C_i)$ and $\ell_R(v^-) = (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee D)$, and let $\ell_R(v) = \bigvee_{i=1}^n C_i \vee D$. Partial interpolants are indicated as before.

Induction hypothesis:

satellites $(1 \leq i \leq n)$	nucleus
$I_{v_i^+} \Rightarrow I'_{v_i^+} \vee (x_i \vee C_i)\vert_{\mathsf{ab},L,L'}$	$I_{v^-} \Rightarrow I'_{v^-} \vee (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee D)\vert_{\mathsf{ab},L,L'}$

Recall from the proof of Lemma 1, that if $L(v_i^+,x) \preceq L'(v_i^+,x)$ (for all $i \in \{1..n\}$) and $L(v^-,\overline{x}) \preceq L'(v^-,\overline{x})$, then,

$$L(v_1^+,x) \sqcup \cdots \sqcup L(v_n^+,x) \sqcup L(v^-,\overline{x}) \quad \preceq \quad L'(v_1^+,x) \sqcup \cdots \sqcup L'(v_n^+,x) \sqcup L'(v^-,\overline{x})\,. \tag{6}$$

For the induction step, let $\ell(v_i^+) = (x_i \vee C_i)$ and $\ell(v^-) = (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee D)$. We assume that $\forall i \in \{1..n\}\,.\, L(v_i^+,x_i) \sqcup L(v^-,\overline{x}_i) = \mathsf{c}$ (for a fixed c) and perform a case split on c. I and I' denote the partial interpolants due to $\mathsf{Itp}(L,R)$ and $\mathsf{Itp}(L',R)$, respectively.

1. $\forall i \in \{1..n\}\,.\, L(v_i^+,x_i) \sqcup L(v^-,\overline{x}_i) = \mathsf{a}$:
   Then $I_v = \bigvee_{i=1}^n I_{v_i^+} \vee I_{v^-}$, and from (6) we conclude that for all $i \in \{1..n\}$ it holds that $L'(v_i^+,x_i) \sqcup L'(v^-,\overline{x}_i) = \mathsf{a}$, and therefore $I'_v = \bigvee_{i=1}^n I'_{v_i^+} \vee I'_{v^-}$. Moreover, $L'(v_i^+,x_i) = L'(v^-,\overline{x}_i) = \mathsf{a}$ for $1 \leq i \leq n$, and therefore, the induction hypothesis can be simplified to

satellites $(1 \leq i \leq n)$	nucleus
$I_{v_i^+} \Rightarrow I'_{v_i^+} \vee C_i\vert_{\mathsf{ab},L,L'}$	$I_{v^-} \Rightarrow I'_{v^-} \vee D\vert_{\mathsf{ab},L,L'}\,.$

   We derive $I_v \Rightarrow \left(\bigvee_{i=1}^n (I'_{v_i^+} \vee C_i\vert_{\mathsf{ab},L,L'}) \vee (I'_{v^-} \vee D\vert_{\mathsf{ab},L,L'})\right)$, and by applying (5), we obtain $I_v \Rightarrow \bigvee_{i=1}^n I'_{v_i^+} \vee I'_{v^-} \vee \left(\bigvee_{i=1}^n C_i \vee D\right)\vert_{\mathsf{ab},L,L'}$, which is equivalent to $I_v \Rightarrow I'_v \vee \ell(v)\vert_{\mathsf{ab},L,L'}$.

2. $\forall i \in \{1..n\}\,.\, L(v_i^+,x_i) \sqcup L(v^-,\overline{x}_i) = \mathsf{ab}$:
   We distinguish two cases for AB-HyRes.
   In the first case, $I_v = \bigwedge_{i=1}^n (x_i \vee I_{v_i^+}) \wedge (I_{v^-} \vee \bigvee_{i=1}^n \overline{x}_i)$, and by applying the induction hypothesis we derive

   $$I_v \Rightarrow \left(\bigwedge_{i=1}^n (x_i \vee I'_{v_i^+} \vee (x_i \vee C_i)\vert_{\mathsf{ab},L,L'}) \wedge (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee I'_{v^-} \vee (\overline{x}_1 \vee \cdots \overline{x}_n \vee D)\vert_{\mathsf{ab},L,L'})\right)\,.$$

   Note that the right-hand side of this implication is equivalent to

   $$\left(\bigwedge_{i=1}^n (x_i \vee I'_{v_i^+} \vee C_i\vert_{\mathsf{ab},L,L'}) \wedge (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee I'_{v^-} \vee D\vert_{\mathsf{ab},L,L'})\right)\,,$$

   which in turn implies

   $$\bigwedge_{i=1}^n (x_i \vee I'_{v_i^+} \vee (\bigvee_{i=1}^n C_i\vert_{\mathsf{ab},L,L'} \vee D\vert_{\mathsf{ab},L,L'})) \wedge (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee I'_{v^-} \vee (\bigvee_{i=1}^n C_i\vert_{\mathsf{ab},L,L'} \vee D\vert_{\mathsf{ab},L,L'}))\,.$$

By applying (5), we obtain

$$I_v \Rightarrow \bigwedge_{i=1}^{n} (x_i \vee I'_{v^+}) \wedge (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee I'_{v^-}) \vee (\bigvee_{i=1}^{n} C_i \vee D)|_{\mathsf{ab},L,L'} \,,$$

which establishes $I_v \Rightarrow I'_v \vee \ell(v)|_{\mathsf{ab},L,L'}$ for the case in which $L'(v_i^+, x) \sqcup L'(v^-, \overline{x}_i) = \mathsf{ab}$ for all $i \in \{1..n\}$. Moreover, since $\bigwedge_{i=1}^{n}(x_i \vee I'_{v^+}) \wedge (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee I'_{v^-})$ implies $(\bigvee_{i=1}^{n} I'_{v_i^+} \vee I'_{v^-})$ (by hyper-resolution), it also holds that $I_v \Rightarrow I'_v \vee \ell(v)|_{\mathsf{ab},L,L'}$ if $L'(v_i^+, x_i) \sqcup L'(v^-, \overline{x}_i) = \mathsf{a}$ (for $1 \leq i \leq n$) and $I'_v = (\bigvee_{i=1}^{n} I'_{v_i^+} \vee I'_{v^-})$. From (6) we conclude that $\nexists i \,.\, L'(v_i^+, x) \sqcup L'(v^-, \overline{x}_i) = \mathsf{b}$.

In the second case, $I_v = \bigvee_{i=1}^{n}(\overline{x}_i \wedge I_{v_i^+}) \vee (I_{v^-} \wedge \bigwedge_{i=1}^{n} x_i)$, and by applying the induction hypothesis we derive

$$I_v \Rightarrow \left(\bigvee_{i=1}^{n} (\overline{x}_i \wedge (I'_{v_i^+} \vee (x_i \vee C_i)|_{\mathsf{ab},L,L'})) \vee \right.$$
$$\left. (x_1 \wedge \cdots \wedge x_n \wedge (I'_{v^-} \vee (\overline{x}_1 \vee \cdots \overline{x}_n \vee D)|_{\mathsf{ab},L,L'})) \right) \,,$$

which implies

$$I_v \Rightarrow \left(\bigvee_{i=1}^{n} ((\overline{x}_i \wedge I'_{v_i^+}) \vee C_i|_{\mathsf{ab},L,L'}) \vee ((x_1 \wedge \cdots \wedge x_n \wedge I'_{v^-}) \vee D|_{\mathsf{ab},L,L'}) \right) \,,$$

and by (5) we derive $I_v \Rightarrow I'_v \vee (\bigvee_{i=1}^{n} C \vee D)|_{\mathsf{ab},L,L'}$, which establishes the correctness of our claim for the case in which $L'(v_i^+, x) \sqcup L'(v^-, \overline{x}_i) = \mathsf{ab}$ for all $i \in \{1..n\}$. Moreover, since $\bigvee_{i=1}^{n}(\overline{x}_i \wedge I'_{v^+}) \vee (x_1 \wedge \cdots \wedge x_n \wedge I'_{v^-})$ implies $(\bigvee_{i=1}^{n} I'_{v_i^+} \vee I'_{v^-})$, it also holds that $I_v \Rightarrow I'_v \vee \ell(v)|_{\mathsf{ab},L,L'}$ if $L'(v_i^+, x_i) \sqcup L'(v^-, \overline{x}_i) = \mathsf{a}$ (for $1 \leq i \leq n$) and $I'_v = (\bigvee_{i=1}^{n} I'_{v_i^+} \vee I'_{v^-})$. As previously, $\nexists i \,.\, L'(v_i^+, x) \sqcup L'(v^-, \overline{x}_i) = \mathsf{b}$ holds.

3. $\forall i \in \{1..n\} \,.\, L(v_i^+, x_i) \sqcup L(v^-, \overline{x}_i) = \mathsf{b}$:
   Then $I_v = \bigwedge_{i=1}^{n} I_{v_i^+} \wedge I_{v^-}$. By (6), we need to distinguish three cases:

   (a) $\forall i \in \{1..n\} \,.L'(v_i^+, x_i) \sqcup L'(v^-, \overline{x}_i) = \mathsf{b}$:
   Then, as in case 1, the induction hypothesis can be simplified to $I_{v_i^+} \Rightarrow I'_{v_i^+} \vee C_i|_{\mathsf{ab},L,L'}$ (where $1 \leq i \leq n$) and $I_{v^-} \Rightarrow I'_{v^-} \vee D|_{\mathsf{ab},L,L'}$, and we obtain

   $$\bigwedge_{i=1}^{n} I_{v_i^+} \wedge I_{v^-} \Rightarrow \left(\bigwedge_{i=1}^{n} (I'_{v_i^+} \vee C_i|_{\mathsf{ab},L,L'}) \wedge (I'_{v^-} \vee D|_{\mathsf{ab},L,L'}) \right) .$$

   By an argument similar to the one made in case 2 we derive $I_v \Rightarrow I'_v \vee (\bigvee_{i=1}^{n} C_i \vee D)|_{\mathsf{ab},L,L'}$.

   (b) $\forall i \in \{1..n\} \,.L'(v_i^+, x_i) \sqcup L'(v^-, \overline{x}_i) = \mathsf{ab}$:
   Then $I'_v = \bigwedge_{i=1}^{n}(x_i \vee I'_{v_i^+}) \wedge (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee I'_{v^-})$ in the first case of AB-HyRes, and by applying the induction hypothesis we derive

   $$I_v \Rightarrow \bigwedge_{i=1}^{n} (I'_{v_i^+} \vee (x_i \vee C_i)|_{\mathsf{ab},L,L'}) \wedge (I'_{v^-} \vee (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee D)|_{\mathsf{ab},L,L'}) \,.$$

   The right-hand side of the implication in turn implies

   $$\bigwedge_{i=1}^{n} (x_i \vee I'_{v_i^+} \vee C_i|_{\mathsf{ab},L,L'}) \wedge (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee I'_{v^-} \vee D|_{\mathsf{ab},L,L'}) \,, \tag{7}$$

and by further weakening (7) and applying (5) we derive

$$I_v \Rightarrow \bigwedge_{i=1}^{n} (x_i \vee I'_{v_i^+}) \wedge (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee I'_{v^-}) \vee (\bigvee_{i=1}^{n} C_i \vee D)|_{\mathsf{ab},L,L'} \,. \tag{8}$$

Finally, (8) also establishes $I_v \Rightarrow I'_v \vee (\bigvee_{i=1}^{n} C \vee D)|_{\mathsf{ab},L,L'}$ for case 2 of AB-HyRes (by Proposition 1).

(c) $\forall i \in \{1..n\} \,.\, L'(v_i^+, x_i) \sqcup L'(v^-, \overline{x}_i) = \mathsf{a}$:
Then $I'_v = (\bigvee_{i=1}^{n} I'_{v_i^+} \vee I'_{v^-})$. Since we have previously shown (8) and since furthermore $\bigwedge_{i=1}^{n}(x_i \vee I'_{v_i^+}) \wedge (\overline{x}_1 \vee \cdots \vee \overline{x}_n \vee I'_{v^-})$ implies (by Proposition 1) $\bigvee_{i=1}^{n}(\overline{x}_i \vee I'_{v_i^+}) \vee (x_1 \wedge \cdots \wedge x_n \wedge I'_{v^-})$, which in turn implies $\bigvee_{i=1}^{n} I'_{v_i^+} \vee I'_{v^-}$, we conclude that $I_v \Rightarrow I'_v \vee \ell(v)|_{\mathsf{ab},L,L'}$.

■

Theorem 4 (Correctness) *For any (A,B)-refutation R (where R is a clausal proof) and locality preserving labelling function L, $\mathsf{ltp}(L,R)$ (if defined) is an interpolant for (A,B).*

Proof: Analogous to the proof of Theorem 1 by induction over the structure of the (A,B)-refutation R. Let I be the partial interpolant at a vertex v labelled with a clause $C = \ell(v)$. We show that every such I and C satisfy the following conditions:

1. $A \wedge \neg(C\restriction_{\mathsf{a},L}) \Rightarrow I$,
2. $B \wedge \neg(C\restriction_{\mathsf{b},L}) \Rightarrow \neg I$, and
3. $\mathrm{Var}(I) \subseteq \mathrm{Var}(A) \cap \mathrm{Var}(B)$.

For the sink v with $\ell(v) = \square$, this establishes Theorem 4. The labelling function L, being unique in this proof, is omitted from subscripts.

Base case. As in the proof of Theorem 1.

Induction step: We perform a case split for the labelling of the pivots:

1. (A-TCRes): $\forall i \in \{1..n\} \,.\, t \in P_i \Rightarrow L(v_i, t) = \mathsf{a}$
   Induction hypothesis (for $i \in \{1 \ldots n\}$):

$$A \wedge \neg(P_i\restriction_{\mathsf{a}}) \wedge \neg(C_i\restriction_{\mathsf{a}}) \Rightarrow I_i$$
$$B \wedge \neg(C_i\restriction_{\mathsf{b}}) \Rightarrow \neg I_i$$

   In this case $P_i\restriction_{\mathsf{a}} = P_i$, so it follows that $A \wedge \neg(C_i\restriction_{\mathsf{a}}) \Rightarrow (P_i \vee I_i)$ and therefore:

$$A \wedge \bigwedge_{i=1}^{n} \neg(C_i\restriction_{\mathsf{a}}) \Rightarrow \bigwedge_{i=1}^{n} (P_i \vee I_i)$$

   Since $\bigwedge_{i=1}^{n} P_i$ is unsatisfiable, we can apply chain resolution to the right and side to derive $\bigwedge_{i=1}^{n}(P_i \vee I_i) \equiv \bigvee_{i=1}^{n} I_i$. By furthermore rewriting the left hand side, we obtain

$$A \wedge \neg(\bigvee_{i=1}^{n} C_i)\restriction_{\mathsf{a}} \Rightarrow \bigvee_{i=1}^{n} I_i \,,$$

   satisfying the first condition. Similarly, we derive

$$B \wedge \bigwedge_{i=1}^{n} \neg(C_i\restriction_{\mathsf{b}}) \Rightarrow \bigwedge_{i=1}^{n} \neg I_i \qquad \text{and} \qquad B \wedge \neg(\bigvee_{i=1}^{n} C_i)\restriction_{\mathsf{b}} \Rightarrow \neg \bigvee_{i=1}^{n} I_i$$

   from the induction hypothesis, which satisfies the second condition. The third condition is satisfied because for all $i \in \{1 \ldots n\}, \mathrm{Var}(I_i) \subseteq \mathrm{Var}(A) \cap \mathrm{Var}(B)$ and therefore $\mathrm{Var}(\bigvee_{i=1}^{n} I_i) \subseteq \mathrm{Var}(A) \cap \mathrm{Var}(B)$ holds as well.

2. (B-TCRes): $\forall i \in \{1..n\} \,.\, t \in P_i \Rightarrow L(v_i, t) = \mathsf{b}$: The proof is symmetric to the first case.

3. (AB-TCRes): $\forall i, j \in \{1..n\},\, i \neq j \,.\, x \in P_i \wedge \overline{x} \in P_j \Rightarrow L(v_j, x_i) \sqcup L(v_i, \overline{x}_i) = \mathsf{ab}$:
   Note that for an arbitrary labelling function L, we have $\neg(P_i\restriction_{\mathsf{a}}) \Rightarrow \neg P_i$ and $\neg(P_i\restriction_{\mathsf{b}}) \Rightarrow \neg P_i$. Therefore, we can strengthen the induction hypotheses to

$$A \wedge \neg P_i \wedge \neg(C_i\restriction_{\mathsf{a}}) \Rightarrow I_i$$
$$B \wedge \neg P_i \wedge \neg(C_i\restriction_{\mathsf{b}}) \Rightarrow \neg I_i$$

   It follows immediately that

$$A \wedge \neg(\bigvee_{i=1}^{n} C_i\restriction_{\mathsf{a}}) \Rightarrow \bigwedge_{i=1}^{n} (P_i \vee I_i)\,,$$

   establishing the first condition for case 1 of AB-TCRes. By applying Proposition 2 to weaken the right hand side (which yields $A \wedge \neg(\bigvee_{i=1}^{n} C_i\restriction_{\mathsf{a}}) \Rightarrow \bigvee_{i=1}^{n}(\neg P_i \wedge I_i)$), we establish the condition 1 for case 2 of AB-TCRes.

   Analogously we derive

$$B \wedge \neg(\bigvee_{i=1}^{n} C_i\restriction_{\mathsf{b}}) \Rightarrow \bigwedge_{i=1}^{n} (P_i \vee \neg I_i)\,, \tag{9}$$

   which establishes the second condition for case 2 of AB-TCRes ($B \wedge \neg(\bigvee_{i=1}^{n} C_i\restriction_{\mathsf{b}}) \Rightarrow \neg \bigvee_{i=1}^{n}(\neg P_i \wedge I_i)$). By Proposition 2, $\neg \bigvee_{i=1}^{n}(\neg P_i \wedge I_i)$ implies $\neg \bigwedge_{i=1}^{n}(P_i \vee I_i)$. Therefore, it follows that

$$B \wedge \neg(\bigvee_{i=1}^{n} C_i\restriction_{\mathsf{b}}) \Rightarrow \neg \bigwedge_{i=1}^{n} (P_i \vee I_i)\,,$$

   which establishes the second condition for case 1 of AB-TCRes. The third condition is established by the fact that all the pivot literals in P_i are shared.

■

Theorem 6 *Let R be a local and closed (A, B)-refutation. Then we can construct a hyper-resolution refutation H of (A, B) and a locality preserving labelling function L such that for each $v \in V_R$ with $\ell_R(v) \in \mathcal{L}(A) \cap \mathcal{L}(B)$ there exists a corresponding vertex $u \in V_H$ such that $\mathsf{Itp}_{KV}(R)(v) \Leftrightarrow \mathsf{Itp}_1(L, H)(u)$.*

Proof: By induction over the structure of the (A, B)-refutation R. Let I_v be the partial interpolant at a vertex v.

Base case. For each initial vertex $v \in V_R$ we construct a vertex $u \in V_H$ with $\ell_H(u) = \ell_R(v)$. First, consider the case that $\ell_R(v) \in \mathcal{L}(A) \cap \mathcal{L}(B)$. We distinguish two cases:

1. $\ell_R(v) \in A$: Then $I_v = \ell_R(v)$. Let $L(u, \ell_R(v)) = \mathsf{b}$, and since $\ell_H(u) \in A$ we have $I_u = I_v$.
2. $\ell_R(v) \in B$: Then $I_v = \neg\ell_R(v)$. Let $L(u, \ell_R(v)) = \mathsf{a}$, and since $\ell_H(u) \in B$ we have $I_u = I_v$.

Otherwise, $\ell_R(v) \notin \mathcal{L}(A) \cap \mathcal{L}(B)$. Then $L(u, \ell_H(u)) = \mathsf{a}$ if $\ell_R(v) \in A$ and $L(u, \ell_H(u)) = \mathsf{b}$ if $\ell_R(v) \in B$, and therefore and $I_u = \mathsf{F}$ or $I_u = \mathsf{T}$, respectively.

Induction step. Let $v \in V_R$ be an internal vertex such that $\ell_R(v) \in \mathcal{L}(A) \cap \mathcal{L}(B)$ and $\{v_1, \ldots, v_n\} = \pi(v)$-premise$(v)$, and $\ell(v_i) \in \mathcal{L}(A) \cap \mathcal{L}(B)$ for $1 \leq i \leq m \leq n$ and $\ell(v_j) \notin \mathcal{L}(A) \cap \mathcal{L}(B)$ for $m < j \leq n$.

Induction hypothesis. There are $\{u_1, \ldots, u_n\} \subseteq V_H$ such that

1. $\ell_H(u_i) = \ell_R(v_i)$ for $1 \leq i \leq n$, and
2. $I_{u_i} \Leftrightarrow I_{v_i}$ for $1 \leq i \leq m$, and
3. $I_{u_j} = \mathsf{F}$ if $\ell(v_j) \in A$ ($I_{u_j} = \mathsf{T}$ if $\ell(v_j) \in B$, respectively) for $m < j \leq n$.

By Corollary 1, the clause $C = \neg\ell_H(u_1) \vee \cdots \vee \neg\ell_H(u_n) \vee \ell_R(v)$ is a tautology. We distinguish two cases:

1. (A-justified) $\pi(v) = A$.
   Add C to A and add w_1 to V_H such that $\ell_H(w_1) = C$. Let $L(w_1, \neg\ell(u_i)) = \mathsf{ab}$ for $1 \leq i \leq m$, $L(w_1, \neg\ell(u_j)) = \mathsf{a}$ for $m < j \leq n$, and $L(w_1, \ell(v)) = \mathsf{a}$. Then, add w_2 to V_H and perform the hyper-resolution step

   $$\frac{\ell_H(u_{m+1}) \quad [\mathsf{F}] \quad \cdots \quad \ell_H(u_n) \quad [\mathsf{F}] \qquad \ell_H(w_1) \quad [\mathsf{F}]}{\ell_H(w_2) \quad [\mathsf{F}]},$$

   where $\ell_H(w_2) = \ell_H(u_1) \vee \cdots \ell_H(u_m) \vee \ell_R(v)$ and $I_{w_2} = \mathsf{F}$. Then, add u to V_H and perform the hyper-resolution step

   $$\frac{\ell_H(u_1) \quad [I_{u_1}] \quad \cdots \quad \ell_H(u_m) \quad [I_{u_m}] \qquad \ell_H(w_2) \quad [\mathsf{F}]}{\ell_H(u) \quad [I_u]},$$

   such that $\ell_H(u) = \ell_R(v)$ and $I_u = \bigwedge_{i=1}^{m}(\ell_H(u_i) \vee I_{u_i}) \wedge (\mathsf{F} \vee \bigvee_{i=1}^{m} \neg\ell_H(u_i))$.

2. (B-justified) $\pi(v) = B$. Analogous to the first case, but $I_{w_2} = \mathsf{T}$.

■