Department of Distributed and Dependable Systems
Technical report no. 2013/05
November 11, 2013

On Interpolants and Variable Assignments

Pavel Jančík, Jan Kofroň

Abstract: Craig interpolants are widely used in program verification as a means of abstraction. In this paper, we introduce *Partial Variable Assignment Interpolants (PVAIs)* as a generalization of Craig interpolants. Variable assignment focuses computed interpolants by restricting the set of clauses taken into account during interpolation. PVAIs can be for example employed in the context of DAG interpolation, in order to prevent unwanted out-of-scope variables to appear in interpolants. Furthermore, we (i) present a way to compute PVAIs for propositional logic based on an extension of the Labeled Interpolation Systems, and (ii) analyze the strength of computed interpolants and prove the conditions under which they have the path interpolation property.

This work is partially supported by the Grant Agency of the Czech Republic project P202/11/0312, and Charles University Foundation grant 203-10/253297.

1 Introduction

In software model checking Craig interpolants play an important role. They are typically used to refine an abstraction of a program. Many techniques have been introduced to compute interpolants for various theories such as propositional logic, conjunctive fragments of linear arithmetic, and octagon domain. For propositional logic, McMillan's [7] and symmetric Pudlák's [9] interpolation systems are well established. They are generalized by the Labeled Interpolation Systems [5] (LISs), which permit to systematically compute interpolants of different logical strength from the same refutation.

Given two formulas A and B such that $A \wedge B$ is unsatisfiable, a Craig interpolant is a formula I such that A implies I, I is inconsistent with B and I is defined over the common variables of both A and B. In other words, I is an over-approximation of A (that is why it can be effectively used for abstracting a set of behaviors of a system, represented by A) which is disjoint from B (which often represents the errors).

In this paper, we introduce *Partial Variable Assignment Interpolants (PVAIs)* – a generalization of Craig interpolants – which, in addition to the standard subdivision of an unsatisfiable formula into A and B parts, takes as input also the specification of a partial variable assignment (PVA). The assignment is used to restrict the original interpolation problem being solved to the cases conforming to the assignment (*sub-problem*). A sub-problem is a part of the original problem where the clauses (constraints) satisfied by the assignment are removed. Due to the above restriction the interpolants contain only variables relevant to the sub-problem, i.e. those shared between the A and B parts of the sub-problem.

In the motivation example below we show how the PVAIs apply to program verification. For instance, in the context of DAG interpolation [1] (and abstract reachability graphs), the sub-problem can be seen as computing a *node interpolant* – an over-approximation of states reachable by any path to a given node. Then, the properties of PVAIs guarantee that the interpolant contains only in-scope program variables.

An alternative approach could be to solve each sub-problem directly by means of multiple calls to a SAT/SMT solver and of standard Craig interpolation. The method we propose allows to perform a single call to a solver on a problem which encompasses all the sub-problems, thus processing the parts common to multiple sub-problems only once, and to generate a single proof from which all the interpolants are computed. The presence of a single proof, in turn, enables the application of existing techniques in order to generate collections of interpolants which satisfy properties relevant to verification, such as path interpolation [11, 6]; in the case of PVAIs, a collection may consist of the interpolants associated with the different sub-problems. Moreover, if one is interested in systematically generating interpolants of different logical strength (a feature intuitively relevant to verification since it affects the coarseness of the over-approximations realized by interpolants [10]), then the approach proposed in this paper allows to do that by extending and adapting the framework of LISs.

In regards to that, we present the new framework of *Labeled Partial Assignment Interpolation Systems* (LPAISs) – a generalization of LISs, which computes PVAIs for propositional logic. The interpolants computed by LPAISs are of smaller size compared to the ones yielded by LISs from the same proof.

We define the notion of logical strength for LPAISs and show how introducing a partial order over LPAISs allows to systematically compare the strength of the generated interpolants. We also show how LPAISs can be used to generate collections of interpolants which enjoy the path interpolation property. Moreover we generalize the results even for different sub-problems. In the context of abstract reachability graphs (ARG), node interpolants for different nodes come from different sub-problems. Due to the generalization, node interpolants for any path in abstract reachability graph have the path interpolation property.

2 Motivation

In the following, we illustrate a possible application of PVAIs, which originally motivated this work. Later, we generalize the idea to make it applicable in other contexts.

As an example, consider the source code at the left-hand side of Fig. 1 and the corresponding *abstract reachability graph (ARG)* at the right-hand side. The node 1 is the initial node, while the node 6 is the node representing an error location. The *edge constraints* τ_{ij} encode the semantics of corresponding program statements. Note that τ_{12} comes from the call of the max function in main (at the line 6). Further, in the node 3, the parameter i is the only live variable; similarly in the node 4 the parameter j is the only live variable. In the context of program verification, an important question is what is the set of reachable

```
1: int max(int i, int j) {
2:    if (i > j)
3:       return i;
      else
4:       return j;
5: }
   // The main function
6: assert(max(random(), 0) >= 0);
```

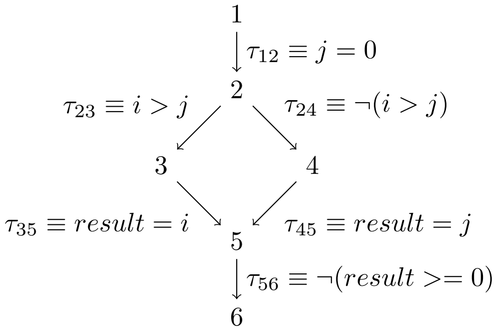
1
$\tau_{12} \equiv j = 0$
2
$\tau_{23} \equiv i > j$
$\tau_{24} \equiv \neg(i > j)$
3
4
$\tau_{35} \equiv result = i$
$\tau_{45} \equiv result = j$
5
$\tau_{56} \equiv \neg(result >= 0)$
6

Figure 1: *Motivation example*

states (on paths ending) at a particular node [3, 8] known as a *reachability problem*.

The ARG is converted into the Cond condition[1], which represents all execution paths in the ARG. Additional *structure-encoding* Boolean variables v_i correspond to the nodes in the ARG. For each (but the final) node, a *node formula* μ_i encoding the actions on the outgoing edges is created; the resulting Cond formula can be found in Fig. 2.

We describe the meaning of μ_2. The first conjunct $v_2 \Rightarrow (v_3 \vee v_4)$ guarantees that if a path reaches the node 2, an outgoing edge is taken so the path does not terminate in 2. The second conjunct $(v_2 \wedge v_3) \Rightarrow \tau_{23}$ guarantees that if a path goes via the edge $2 \rightarrow 3$, the semantics of the edge is preserved (i.e. edge-constraint τ_{23} holds). Similarly, the third conjunct preserves the semantics of the edge $2 \rightarrow 4$.

The Cond formula is satisfiable (a model exists) if and only if a feasible error trace exists. If Cond is unsatisfiable, one can split the nodes into A and B, divide Cond correspondingly, and compute a Craig interpolant which includes *all paths* from A to B. Thus, it over-approximates the states reachable at *any node* on the boundary between A and B. For instance, let us assume that $A = v_1 \wedge \mu_1 \wedge \mu_2$ and $B = \mu_3 \wedge \mu_4 \wedge \mu_5$. The interpolant over-approximates the states reachable at the nodes 3 or 4; it may contain both the variables being in-scope at the node 3 (variable i) and at the node 4 (variable j)[2]. Preferably, the solution of the reachability problem for a node should not contain out-of-scope program variables. For the node 3 interpolant this means the variable j should be removed (e.g., eliminated by quantification which is a well-known bottleneck in verification).

Using PVAIs, the problem of out-of-scope program variables can be effectively solved. We consider only such variable assignments where the structure encoding variables v_i are assigned. By setting a variable v_i to False, the paths via the node i are blocked. Moreover, the node formula μ_i is satisfied, thus not present in the sub-problem. To compute an interpolant for the node 3, the assignment is $\pi_3 \equiv \overline{v_4}$. The assignment must block exactly the paths not going through the node 3 – here it is only the path via the node 4. In the A part, the sub-problem for the node 3 contains only the edge actions (the program state variables) at the paths to the node 3, and in the B part only those actions at the paths out of that node. The program state variables shared by the A and B parts of the sub-problem represent the in-scope variables, which are exactly those that may appear in PVA Interpolants.

$$\begin{array}{ll}
\mu_1 \equiv (v_1 \Rightarrow v_2) & \wedge\, ((v_1 \wedge v_2) \Rightarrow \tau_{12}) \\
\mu_2 \equiv (v_2 \Rightarrow (v_3 \vee v_4)) & \wedge\, ((v_2 \wedge v_3) \Rightarrow \tau_{23}) \wedge \\
 & \wedge\, ((v_2 \wedge v_4) \Rightarrow \tau_{24}) \\
\mu_3 \equiv (v_3 \Rightarrow v_5) & \wedge\, ((v_3 \wedge v_5) \Rightarrow \tau_{35}) \\
\mu_4 \equiv (v_4 \Rightarrow v_5) & \wedge\, ((v_4 \wedge v_5) \Rightarrow \tau_{45}) \\
\mu_5 \equiv (v_5 \Rightarrow v_6) & \wedge\, ((v_5 \wedge v_6) \Rightarrow \tau_{56})
\end{array}$$

$$\mathsf{Cond} \equiv v_1 \wedge \mu_1 \wedge \mu_2 \wedge \mu_3 \wedge \mu_4 \wedge \mu_5$$

Figure 2: *The* Cond *formula*

3 Preliminaries

Resolution Refutations. A *literal* is a variable or its negation. A clause is a finite disjunction of literals. Let Θ be a set of literals. We write $\langle\Theta\rangle$ for the clause containing the literals from Θ. A clause can be also composed from multiple sets of literals, as in $\langle\Theta, \Theta'\rangle$ or $\langle\Theta, \{l\}\rangle$ – we represent the latter as $\langle\Theta, l\rangle$ for brevity. Let $\langle\Theta, p\rangle$ and $\langle\Theta', \overline{p}\rangle$ be clauses. Their *resolvent* is the clause $\langle\Theta, \Theta'\rangle$ and p is the *pivot*. We write $Res(\langle\Theta, p\rangle, \langle\Theta', \overline{p}\rangle, p)$ for the resolvent of clauses $\langle\Theta, p\rangle$ and $\langle\Theta', \overline{p}\rangle$ with pivot p. We assume that a clause does not contain both a literal and its negation. In the following, we consider propositional formulas in *conjunctive normal form*, i.e., as conjunctions (or equivalently sets) of clauses. For a literal l or a set of

[1] The Cond has the same meaning as ArgCond in [2].

[2] This is an inherent property of Craig interpolants, independent from ARG encoding.

propositional formulas A, $\mathsf{Var}(l)$ and $\mathsf{Var}(A)$ respectively denote the variable of l or the set of variables in the formulas of A.

Definition 3.1 (Resolution proof (taken from [5]))**.** *A resolution proof R is a tuple (V, E, cl, piv, s), where V is a set of vertices, E is a set of edges, piv is a pivot function, cl a clause function, and $s \in V$ a sink vertex.*

$(V,\ E)$ form a full binary DAG (i.e., all the vertices except for the leaves have the in-degree 2), the sink has the out-degree 0. For each inner node v there exist edges $(v_1, v), (v_2, v) \in E$ and it holds: $cl(v) = Res(cl(v_1), cl(v_2), piv(v))$.

We drop the subscripts if clear from the context. A proof is a *resolution refutation proof* if $cl(s) = \bot$. To distinguish between vertices in resolution proofs and in abstract reachability graphs, we use the term *node* for ARG, and the term *vertex* for resolution proofs.

Craig Interpolants. The *Craig interpolant* [4] for the pair of formulas (A, B) such that $A \wedge B$ is unsatisfiable is a formula I such that (1) $A \Rightarrow I$, (2) $B \wedge I \Rightarrow \bot$, and (3) $\mathsf{Var}(I) \subseteq \mathsf{Var}(A) \cap \mathsf{Var}(B)$.

An *interpolant sequence* for the unsatisfiable formula $A_1 \wedge A_2 \wedge ... \wedge A_n$ is a tuple of formulas $(I_0, I_1, ...I_n)$, where $I_0 \equiv \top$ is an interpolant for $(\top, A_1 \wedge ... \wedge A_n)$, I_i is an interpolant for $(A_1 \wedge ... \wedge A_i, A_{i+1} \wedge ... \wedge A_n)$, and $I_n \equiv \bot$ is an interpolant for $(A_1 \wedge ... \wedge A_n, \top)$. If, for all i, $I_i \wedge A_i \Rightarrow I_{i+1}$ (*inductive step*), then $(I_0, I_1, ...I_n)$ is also said to satisfy the *path interpolation property*. In [6], it was proved that path interpolation property holds for any LISs, including the well-known McMillan's and Pudlák's systems, whenever the interpolant sequence is computed from the same proof.

Variable assignments. Let A be a set of formulas. A *variable assignment* assigns either True $(\top)$ or False $(\bot)$ to each variable in the $\mathsf{Var}(A)$ set. Alternatively, the variable assignment can be seen as a conjunction of literals. A *partial variable assignment* (PVA) π assigns values only to a subset of variables in $\mathsf{Var}(A)$. A PVA π can be used as an assumption w.r.t. a formula ϕ (i.e., $\pi \models \phi$) to restrict the set of models of ϕ to those compatible with π.

Definition 3.2 (Clauses under assignment)**.** *Let A be a set of clauses and π be a PVA over* $\mathsf{Var}(A)$. *We define the sets of* satisfied *clauses A_π and of* unsatisfied *clauses $A_{\overline{\pi}}$ as follows:*

$$A_\pi = \{\langle\Theta\rangle | \langle\Theta\rangle \in A \wedge \pi \models \langle\Theta\rangle\} \qquad A_{\overline{\pi}} = \{\langle\Theta\rangle | \langle\Theta\rangle \in A \wedge \pi \not\models \langle\Theta\rangle\}$$

In the rest of the paper, we make use of the following simple facts: For any set A and any PVA π it holds that $A = A_\pi \cup A_{\overline{\pi}}$ and $A_\pi \cap A_{\overline{\pi}} = \emptyset$. A satisfied clause in A_π contains at least one literal set to True by π, while, for an unsatisfied clause in $A_{\overline{\pi}}$, every literal is either unassigned or falsified. The unsatisfied clauses $A_{\overline{\pi}}$ determine the sub-problem. We use $\pi \models l$ to express that a literal l evaluates to True in a given PVA π.

4 Partial Variable Assignment Interpolants

In this section, we extend the standard notion of Craig interpolation to that of *partial variable assignment interpolation*, which, in addition to the subdivision of an unsatisfiable formula into an A and a B parts, requires the specification of a PVA. Based on this new concept, in Sect. 4.1, we present the framework of *Labeled Partial Assignment Interpolation Systems*, a generalization of [5], and prove its soundness; next, in Sect. 4.2, we relate interpolation to logical strength, and show how the introduction of a partial order on LPAISs allows to systematically compare the strength of the generated interpolants.

Definition 4.1 (Partial Variable Assignment Interpolant)**.** *Let R be a (A, B)-refutation and π a partial variable assignment over* $\mathsf{Var}(A \wedge B)$. *The* partial variable assignment interpolant *(PVAI) is a formula I such that:*

(D4.1.1) $\pi \models A \Rightarrow I$

(D4.1.2) $\pi \models B \wedge I \Rightarrow \bot$

(D4.1.3) $\mathsf{Var}(I) \subseteq \mathsf{Var}(A_{\overline{\pi}}) \cap \mathsf{Var}(B_{\overline{\pi}})$

(D4.1.4) $\mathsf{Var}(I) \cap \mathsf{Var}(\pi) = \emptyset$

Since $\pi \models A \Leftrightarrow A_{\overline{\pi}}$, D4.1.1 and D4.1.2 can be equivalently rewritten as $\pi \models A_{\overline{\pi}} \Rightarrow I$ and $\pi \models B_{\overline{\pi}} \wedge I \Rightarrow \bot$. In the following we use (A, B, π) to denote that the PAI is computed from (A, B)-refutation using the partial assignment π. Note that a PAI cannot be computed as a standard interpolant followed by the application of a partial assignment ($I[\pi]$). The reason is that, according to D4.1.3, the PVAI excludes not only the variables assigned by π, but, e.g., also all unassigned variables that occur in satisfied clauses only, which can instead appear in $I[\pi]$.

A partial assignment can be viewed as a way of removing constraints (satisfied clauses) from being considered during interpolant computation. Thus, there is no need for the variables local to removed constraints to occur in the interpolant itself. However the price to pay is an additional assumption in the form of a partial assignment.

4.1 Labeled Partial Assignment Interpolation System

The idea of computing interpolants of various strength as introduced by the Labeled Interpolation Systems (LIS) [5] can be extended to compute partial variable assignment interpolants. First, we need to extend the original definitions to take variable assignments into account.

Definition 4.2 (Labeling function [5])**.** *Let $L = (S, \sqsubseteq, \sqcap, \sqcup)$ be the lattice of Fig. 3, where $S = \{\bot, a, b, ab, d^+\}$ and $\bot$ is the least element and let $R = (V, E, cl, piv, s)$ be a resolution proof over a set of literals* Lit. *A function* $\mathsf{Lab}_{R,L} : V \times \mathsf{Lit} \to S$ *is called* labeling function *for a refutation proof R iff $\forall v \in V$ and $\forall l \in$* $\mathsf{Lit}, \mathsf{Lab}_{R,L}$ *satisfies the following:*

(D4.2.1) $\mathsf{Lab}_{R,L}(v, l) = \bot$ *if and only if $l \notin cl(v)$ (literal not in the vertex clause), and*

(D4.2.2) $\mathsf{Lab}_{R,L}(v, l) = \mathsf{Lab}_{R,L}(v_1, l) \sqcup \mathsf{Lab}_{R,L}(v_2, l)$*, where v_1, v_2 are the predecessor vertices with positive and negative pivot, respectively, if $l \in cl(v)$*

The label d^+ is used only for literals satisfied by an assignment. From the condition D4.2.2 it follows that the labeling function is fully determined once the labels of the literals in the leaves have been specified. Note that we omit subscripts R and L if clear.

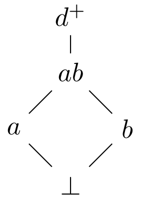

Figure 3: *Lattice of labels (according $\sqsubseteq$)*

Naming conventions. Let us assume a pair of sets of clauses (A, B) and a PVA π. The clause sets are split into four groups, the unsatisfied clauses $A_{\overline{\pi}}$ and $B_{\overline{\pi}}$ which specify the sub-problem and are taken into account during interpolation, and the satisfied clauses A_π and B_π, which are disregarded.

We define the following properties (including the standard locality and sharedness) of variables. We say that a variable v is *unassigned* if $v \notin \mathsf{Var}(\pi)$. An unassigned variable v is:

$A_{\overline{\pi}}$-local	if $v \in \mathsf{Var}(A_{\overline{\pi}})$ and $v \notin \mathsf{Var}(B_{\overline{\pi}})$
$B_{\overline{\pi}}$-local	if $v \notin \mathsf{Var}(A_{\overline{\pi}})$ and $v \in \mathsf{Var}(B_{\overline{\pi}})$
$A_{\overline{\pi}}B_{\overline{\pi}}$-shared	if $v \in \mathsf{Var}(A_{\overline{\pi}})$ and $v \in \mathsf{Var}(B_{\overline{\pi}})$
$A_{\overline{\pi}}B_{\overline{\pi}}$-clean	if $v \notin \mathsf{Var}(A_{\overline{\pi}})$ and $v \notin \mathsf{Var}(B_{\overline{\pi}})$

The properties above are independent from occurrence of the variable in $\mathsf{Var}(A_\pi)$ resp. $\mathsf{Var}(B_\pi)$. The "clean" variables occur only in the satisfied clauses. We say that a variable v is *McMillan-labeled*[3] if the fact that v is $A_{\overline{\pi}}B_{\overline{\pi}}$-shared or $A_{\overline{\pi}}B_{\overline{\pi}}$-clean implies it is labeled b. A variable v is labeled *consistently* if all occurrences of the variable in the proof are labeled by the same label. Formally:
$\forall x, x' \in V, l \in cl(x), l' \in cl(x') : \mathsf{Var}(l) = \mathsf{Var}(l') = x \Rightarrow \mathsf{Lab}(v, l) = \mathsf{Lab}(v', l')$

Locality. The meaning of the locality preserving labeling is the same as in LIS, (only) the locality preserving labeling functions (are guaranteed to) yield interpolants.

Definition 4.3 (Locality preserving labeling)**.** *A labeling function* Lab *for a (A, B, π)-refutation R is* locality preserving *iff $\forall v \in V, \forall l \in cl(v)$:*

(D4.3.1) (satisfied literals) $\mathsf{Lab}(v, l) = d^+ \Leftrightarrow \pi \models l$

[3] Please refer to the labeling of McMillan's interpolation system as defined in [5].

(D4.3.2) ($A_{\overline{\pi}}$-locality) $\mathsf{Var}(l)$ *is unassigned and $A_{\overline{\pi}}$-local*$\Rightarrow$ $\mathsf{Lab}(v, l) = a$

(D4.3.3) ($B_{\overline{\pi}}$-locality) $\mathsf{Var}(l)$ *is unassigned and $B_{\overline{\pi}}$-local*$\Rightarrow$ $\mathsf{Lab}(v, l) = b$

(D4.3.4) ($A_{\overline{\pi}}B_{\overline{\pi}}$-cleanness – satisfied clauses)
$\mathsf{Var}(l)$ *is unassigned and $A_{\overline{\pi}}B_{\overline{\pi}}$-clean$\Rightarrow$ it is consistently labeled as a or b.*

The label of $A_{\overline{\pi}}B_{\overline{\pi}}$-shared variables can be set freely to a, b, or ab. The same holds for falsified (not satisfied) literals; their labels are irrelevant since they are removed by the assignment filter (defined later).

From the D4.3.1 requirement (and the shape of the lattice L) it follows that the assumption (assignment) π can be used in the resolutions where the pivot is labeled d^+. The D4.3.2 and D4.3.3 requirements are equivalent to the standard ones meaning that A-local and B-local variables have to have a or b labels, respectively. However, in our case the locality is considered only over unsatisfied clauses ($A_{\overline{\pi}}$ resp. $B_{\overline{\pi}}$).

The D4.3.4 requirement is specific to the PVAI and deals with variables which occur in the satisfied clauses only. For such a variable it is required that the label is consistently either a or b. This requirement guarantees that such variables do not occur in the interpolant due to ab-resolution.

Filters. For a clause $\langle\Theta\rangle$, a labeling function Lab, a resolution-proof vertex $v \in V$, and a label c, we define the *match filter* $|$ as $\langle\Theta\rangle|_{c,v,\mathsf{Lab}} = \{l \in \langle\Theta\rangle \mid c = \mathsf{Lab}(v, l)\}$ which preserves only literals with specified label and similarly we define the *upward filter* $\upharpoonright$ as $\langle\Theta\rangle\upharpoonright_{c,v,\mathsf{Lab}} = \{l \in \langle\Theta\rangle \mid c \sqsubseteq \mathsf{Lab}(v, l)\}$.

Moreover given a partial assignment π and a clause $\langle\Theta\rangle$ we define the *assignment filter* $[\,]$ as $\langle\Theta\rangle[\pi] = \{l \in \langle\Theta\rangle \mid \mathsf{Var}(l) \notin \mathsf{Var}(\pi))\}$. The filter removes all the assigned literals (satisfied and falsified ones).

Note that we remove the labeling function and vertex from the subscript if clear from the context. Moreover we assume that negation has a higher precedence than filters and (as usual for unary operators) filters have a higher precedence than other binary logical operators. E.g., the $\neg\langle\Theta\rangle[\pi]\upharpoonright_a \wedge\langle\Theta'\rangle|_b$ can be rewritten as $(\neg(((\langle\Theta\rangle[\pi])\upharpoonright_a)) \wedge (\langle\Theta'\rangle|_b)$.

Interpolation system. An interpolation system is a procedure of computing an interpolant from a refutation. It assigns a partial interpolant to each vertex of the refutation proof, while yielding the final interpolant in the sink vertex.

Definition 4.4 (Labeled Partial Assignment Interpolation System). *Let* Lab *be a locality preserving labeling function for a valid (A, B, π)-refutation R.*

The Labeled Partial Assignment Interpolation System $\mathsf{Lpaltp}(\mathsf{Lab}, R)$ *is defined as follows:*

$$\begin{array}{|ll|}\hline \textit{Leaf } v: & \langle\Theta\rangle, [I] \\ \hline I = \begin{cases} \ \langle\Theta\rangle[\pi]|_{b,v,\mathsf{Lab}} & \textit{if } \langle\Theta\rangle \in A_{\overline{\pi}} \\ \neg\ \langle\Theta\rangle[\pi]|_{a,v,\mathsf{Lab}} & \textit{if } \langle\Theta\rangle \in B_{\overline{\pi}} \\ \ \top & \textit{if } \langle\Theta\rangle \in A_{\pi} \cup B_{\pi} \end{cases} & \\ \hline \end{array}$$

$$\begin{array}{|ll|}\hline \textit{Inner vertex } v: & \dfrac{v_1 : \langle p, \Theta_1\rangle, [I_1] \qquad v_2 : \langle \overline{p}, \Theta_2\rangle, [I_2]}{\langle\Theta_1, \Theta_2\rangle, [I]} \\ \hline I = \begin{cases} I_1 \vee I_2 & \textit{if } \mathsf{Lab}(v_1, p) \sqcup \mathsf{Lab}(v_2, \overline{p}) = a \\ I_1 \wedge I_2 & \textit{if } \mathsf{Lab}(v_1, p) \sqcup \mathsf{Lab}(v_2, \overline{p}) = b \\ (I_1 \vee p) \wedge (I_2 \vee \overline{p}) & \textit{if } \mathsf{Lab}(v_1, p) \sqcup \mathsf{Lab}(v_2, \overline{p}) = ab \\ I_2 & \textit{if } \mathsf{Lab}(v_1, p) = d^+ \\ I_1 & \textit{if } \mathsf{Lab}(v_2, \overline{p}) = d^+ \end{cases} & \\ \hline \end{array}$$

The main difference comparing to LIS are additional d^+ resolution rules. For instance, consider the last rule, where $\mathsf{Lab}(v_2, \overline{p}) = d^+$. In contrast to the standard rules, the partial interpolant is simpler, because it does not contain I_2. Generally, the d^+ rules *cut out* the satisfied sub-tree of the proof. Usually, the later in the refutation proof the assigned variable is resolved, the bigger sub-tree is cut out and the smaller the resulting interpolant is.

Theorem 4.5 (Correctness). $\mathsf{Lpaltp}(\mathsf{Lab}, R)$*, for a valid ($A$, B, π)-refutation R and a locality preserving labeling function* Lab*, generates a partial variable assignment interpolant.*

Proof. The main idea of the proof is the same as the one for LIS. By structural induction we show that for each vertex v of a resolution proof the following invariants hold:

(T4.5.Inv1) $\pi \models A \wedge \neg\langle\Theta\rangle\upharpoonright_{a,v,\mathsf{Lab}} \Rightarrow I$

(T4.5.Inv2) $\pi \models B \wedge \neg\langle\Theta\rangle\upharpoonright_{b,v,\mathsf{Lab}} \Rightarrow \neg I$

(T4.5.Inv3) $\mathsf{Var}(I) \subseteq \mathsf{Var}(A_{\overline{\pi}}) \cap \mathsf{Var}(B_{\overline{\pi}})$

where I is the partial interpolant of the vertex v and $cl(v) = \langle\Theta\rangle$.

These invariants are equivalent to the PVAI constraints for the sink node (where the $\neg\langle\Theta\rangle = \top$). We omit the labeling function Lab from subscripts (since it is unique in the proof) and vertex if clear.

Base cases. The base cases take place in the leaf vertices of the proof where the hypotheses operations are applied.

Hyp-$A_{\overline{\pi}}$: $\langle\Theta\rangle \in A_{\overline{\pi}}$ so $I = \langle\Theta\rangle[\pi\rrbracket_b$

(T4.5.Inv1) $\pi \models A \wedge \neg\langle\Theta\rangle\upharpoonright_a \Rightarrow \langle\Theta\rangle[\pi\rrbracket_b$ holds because $A \Rightarrow \langle\Theta\rangle$ and $\langle\Theta\rangle \Leftrightarrow (\langle\Theta\rangle\upharpoonright_a \vee \langle\Theta\rangle|_b)$ so $\langle\Theta\rangle \wedge \neg\langle\Theta\rangle\upharpoonright_a \Rightarrow \langle\Theta\rangle|_b$. Moreover it holds $\pi \models \langle\Theta\rangle|_b \Leftrightarrow \langle\Theta\rangle[\pi\rrbracket_b$ because the $\langle\Theta\rangle$ so even $\langle\Theta\rangle|_b$ is not satisfied by the partial assignment π, so all the assigned literals evaluate to $\bot$.

(T4.5.Inv2) $\pi \models B \wedge \neg\langle\Theta\rangle\upharpoonright_b \Rightarrow \neg\langle\Theta\rangle[\pi\rrbracket_b$ holds because $\neg\langle\Theta\rangle\upharpoonright_b \Rightarrow \neg\langle\Theta\rangle|_b$. Moreover it holds $\pi \models \neg\langle\Theta\rangle|_b \Leftrightarrow \langle\Theta\rangle[\pi\rrbracket_b$. The reason is the same as above, all the assigned literals evaluate to $\bot$.

(T4.5.Inv3) $\mathsf{Var}(\langle\Theta\rangle[\pi\rrbracket_b) \subseteq \mathsf{Var}(A_{\overline{\pi}}) \cap \mathsf{Var}(B_{\overline{\pi}})$. The label b for literals implies that literal variables are $A_{\overline{\pi}}B_{\overline{\pi}}$-shared. Otherwise the locality preserving requirement D4.3.2 yields to a contradiction since it requires the label a. Moreover the assignment filter is applied so the partial interpolant does not contain any assigned variable.

Hyp-$B_{\overline{\pi}}$: $\langle\Theta\rangle \in B_{\overline{\pi}}$ so $I = \neg\langle\Theta\rangle[\pi\rrbracket_a$. The situation is symmetric to Hyp-$A_{\overline{\pi}}$ case.

(T4.5.Inv1) $\pi \models A \wedge \neg\langle\Theta\rangle\upharpoonright_a \Rightarrow \neg\langle\Theta\rangle|_a$ holds because $\neg\langle\Theta\rangle\upharpoonright_a \Rightarrow \neg\langle\Theta\rangle|_a$. Moreover $\pi \models \langle\Theta\rangle|_a \Leftrightarrow \langle\Theta\rangle[\pi\rrbracket_a$ because all the assigned literals in the $\langle\Theta\rangle$ clause evaluate to $\bot$ under the assignment π.

(T4.5.Inv2) $\pi \models B \wedge \neg\langle\Theta\rangle\upharpoonright_b \Rightarrow \langle\Theta\rangle|_a$ holds because $B \Rightarrow \langle\Theta\rangle$ and $\langle\Theta\rangle \Leftrightarrow (\langle\Theta\rangle\upharpoonright_b \vee \langle\Theta\rangle|_a)$. It follows that $\langle\Theta\rangle \wedge \neg\langle\Theta\rangle\upharpoonright_b \Rightarrow \langle\Theta\rangle|_a$. Moreover as shown above $\pi \models \langle\Theta\rangle|_a \Leftrightarrow \langle\Theta\rangle[\pi\rrbracket_a$.

(T4.5.Inv3) $\mathsf{Var}(\neg\langle\Theta\rangle[\pi\rrbracket_a) \subseteq \mathsf{Var}(A_{\overline{\pi}}) \cap \mathsf{Var}(B_{\overline{\pi}})$. The label a for literals implies that literal variables are $A_{\overline{\pi}}B_{\overline{\pi}}$-shared. Otherwise the locality preserving requirements D4.3.3 yields to a contradiction since it requires label b. Moreover the assignment filter is applied so the partial interpolant does not contain any assigned variable.

Hyp-A_π, Hyp-B_π: $\langle\Theta\rangle \in A_\pi \cup B_\pi$ so $I = \top$.

(T4.5.Inv1) $\pi \models A \wedge \neg\langle\Theta\rangle\upharpoonright_a \Rightarrow \top$ holds trivially.

(T4.5.Inv2) $\pi \models B \wedge \neg\langle\Theta\rangle\upharpoonright_b \Rightarrow \bot$.

We will show that the assumptions of the implication are unsatisfied. The reason is that $\neg\langle\Theta\rangle\upharpoonright_b$ evaluates (is equivalent) to $\bot$ given assignment π.

From $\langle\Theta\rangle \in A_\pi$ it follows that $\exists l \in \Theta$ such that $\pi \models l$ (The literal l makes the clause $\langle\Theta\rangle$ satisfied by π). The label of l is d^+ (locality of labeling function – D4.3.1) so the literal is preserved by the upward-filter$\upharpoonright_b$.

Thus $\pi \models \neg\langle\Theta\rangle\upharpoonright_b \Leftrightarrow \bot$.

(T4.5.Inv3) $\mathsf{Var}(\top) \subseteq \mathsf{Var}(A) \cap \mathsf{Var}(B)$ holds trivially.

□

Before the proof of the theorem 4.5 will continue (moving from leaves to inner vertices), auxiliary lemmas are needed. The first one is used to introduce upward-filter for pivot variables. The second lemma connects the assumptions of the current vertex and the assumptions of its predecessor (the assumptions in the induction hypothesis).

Lemma 4.6 (Introducing upward-filters)**.** *Let p be a variable and $c \in L$ be a label and v a vertex. It holds:*

$$\models p \Rightarrow \neg\langle\overline{p}\rangle{\upharpoonright_{c,v}} \qquad \textit{and} \qquad \models \overline{p} \Rightarrow \neg\langle p\rangle{\upharpoonright_{c,v}}$$

Proof. The upward-filter ${\upharpoonright_{c,v}}$ can either preserve the literal $\overline{p}$ or filter it out. In the first case the filter evaluates to $\neg\langle\overline{p}\rangle$ which is equivalent to p and the implication $\models p \Rightarrow p$ holds trivially. In the second case the filter evaluates into the empty clause, i.e. $\neg\mathsf{False}$ and the implication $\models p \Rightarrow \neg\mathsf{False}$ holds trivially.

For the second equation of the lemma the same reasoning applies. □

Lemma 4.7 (Filters in the predecessor vertices)**.** *Let $\langle\Theta_1, \Theta_2\rangle$ be a clause of an inner vertex v. Let $\langle p, \Theta_1\rangle$, $\langle\overline{p}, \Theta_2\rangle$ be the clauses of the (positive and negative) predecessors of v (called v_1, v_2). Let $\mathsf{Lab}_{R,L}$ be a labeling function for the given proof and $c \in L$ a label. Then it holds:*

$$\begin{aligned} &\neg\langle p\rangle{\upharpoonright_{c,v_1}} \wedge \neg\langle\Theta_1, \Theta_2\rangle{\upharpoonright_{c,v}} \Rightarrow \neg\langle p, \Theta_1\rangle{\upharpoonright_{c,v_1}} \qquad \textit{and} \\ &\neg\langle\overline{p}\rangle{\upharpoonright_{c,v_2}} \wedge \neg\langle\Theta_1, \Theta_2\rangle{\upharpoonright_{c,v}} \Rightarrow \neg\langle\overline{p}, \Theta_2\rangle{\upharpoonright_{c,v_2}} \end{aligned}$$

Proof (Lemma 4.7). The upward-filter $\upharpoonright$ filter preserves all the literals whose labels are equal to or above the given label (e.g., ${\upharpoonright_a}$ preserves literals with labels a, ab, d^+). From D4.2.1 and D4.2.2 it follows that $\forall l \in \langle\Theta_1, \Theta_2\rangle$ it holds $\mathsf{Lab}(v_1, l) \sqsubseteq \mathsf{Lab}(v, l)$, so the literals preserved by the upward filter in the vertex v_1 (excluding pivot) are also preserved by the upward filter in the successor vertex v. From the above it follows that $\langle\Theta_1, \Theta_2\rangle{\upharpoonright_{c,v_1}} \Rightarrow \langle\Theta_1, \Theta_2\rangle{\upharpoonright_{c,v}}$, which can be equivalently rewritten into contrapositive implication $\neg\langle\Theta_1, \Theta_2\rangle{\upharpoonright_{c,v}} \Rightarrow \neg\langle\Theta_1, \Theta_2\rangle{\upharpoonright_{c,v_1}} \Rightarrow \neg\langle\Theta_1\rangle{\upharpoonright_{c,v_1}}$

The final implication $\neg\langle p\rangle{\upharpoonright_{c,v_1}} \wedge \neg\langle\Theta_1, \Theta_2\rangle{\upharpoonright_{c,v}} \Rightarrow \neg\langle p, \Theta_1\rangle{\upharpoonright_{c,v_1}}$ with the pivot p holds because the same filter is applied on the pivot (the pivot is either filtered out or preserved by both filters).

And symmetric facts hold for the negative successor. □

Proof (Continuation of Theorem 4.5 – Correctness).

Induction hypothesis. Now, we focus on the inductive step. Be v_1 positive predecessor of the inner vertex v and v_2 its negative predecessor. Let p be the pivot variable. From the induction hypothesis we know that for the predecessor vertices the following invariants hold:

$$\begin{array}{llll} \pi \models A \wedge \neg\langle p, \Theta_1\rangle{\upharpoonright_{a,v_1}} \Rightarrow I_1 & \text{and} & \pi \models B \wedge \neg\langle p, \Theta_1\rangle{\upharpoonright_{b,v_1}} \Rightarrow \neg I_1 & \text{and} \\ \pi \models A \wedge \neg\langle\overline{p}, \Theta_2\rangle{\upharpoonright_{a,v_2}} \Rightarrow I_2 & \text{and} & \pi \models B \wedge \neg\langle\overline{p}, \Theta_2\rangle{\upharpoonright_{b,v_2}} \Rightarrow \neg I_2 & \end{array} \tag{IH}$$

For each type of resolution, we establish the induction invariants for the vertex v.

Res-a: $\mathsf{Lab}(v_1, p) \sqcup \mathsf{Lab}(v_2, \overline{p}) = a$. In this case the pivot variable has the label a in both predecessors v_1 and v_2.

(T4.7.Inv1) It follows that:

$$\begin{aligned} \pi \models \overline{p} \wedge A \wedge \neg\langle\Theta_1, \Theta_2\rangle{\upharpoonright_{a,v}} &\overset{\text{(L4.6)}}{\Rightarrow} \neg\langle p\rangle{\upharpoonright_{a,v_1}} \wedge A \wedge \neg\langle\Theta_1, \Theta_2\rangle{\upharpoonright_{a,v}} \overset{\text{(L4.7)}}{\Rightarrow} \\ &\Rightarrow A \wedge \neg\langle p, \Theta_1\rangle{\upharpoonright_{a,v_1}} \overset{\text{(IH)}}{\Rightarrow} I_1 \\ \pi \models p \wedge A \wedge \neg\langle\Theta_1, \Theta_2\rangle{\upharpoonright_{a,v}} &\overset{\text{(L4.6)}}{\Rightarrow} \neg\langle\overline{p}\rangle{\upharpoonright_{a,v_2}} \wedge A \wedge \neg\langle\Theta_1, \Theta_2\rangle{\upharpoonright_{a,v}} \overset{\text{(L4.7)}}{\Rightarrow} \\ &\Rightarrow A \wedge \neg\langle\overline{p}, \Theta_2\rangle{\upharpoonright_{a,v_2}} \overset{\text{(IH)}}{\Rightarrow} I_2 \end{aligned}$$

The first implication is application of Lemma 4.6. The second implication is application of Lemma 4.7 and the last one is the induction hypothesis.

From the previous implications it directly follows that:

$$\pi \models A \wedge \neg\langle\Theta_1, \Theta_2\rangle{\upharpoonright_{a,v}} \Leftrightarrow (\overline{p} \vee p) \wedge A \wedge \neg\langle\Theta_1, \Theta_2\rangle{\upharpoonright_{a,v}} \Rightarrow (I_1 \vee I_2)$$

The first equivalence is a simple logical consequence since $p \vee \overline{p} \Leftrightarrow \top$, the second implication is a consequence of the two equations above.

(T4.7.Inv2) From the fact that the label of the pivot in the predecessor is a, it follows $\neg\langle p\rangle\upharpoonright_{b,v_1}\Leftrightarrow \neg\langle\overline{p}\rangle\upharpoonright_{b,v_2}\Leftrightarrow \top$ so Lemma 4.7 can be applied directly without any additional assumptions.

$$\pi \models B \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{b,v}\Leftrightarrow \neg\langle p\rangle\upharpoonright_{b,v_1} \wedge B \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{b,v} \overset{\text{(L4.7)}}{\Rightarrow} B \wedge \neg\langle p,\Theta_1\rangle\upharpoonright_{b,v_1} \overset{\text{(IH)}}{\Rightarrow} \neg I_1$$

$$\pi \models B \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{b,v}\Leftrightarrow \neg\langle\overline{p}\rangle\upharpoonright_{b,v_2} \wedge B \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{b,v} \overset{\text{(L4.7)}}{\Rightarrow} B \wedge \neg\langle\overline{p},\Theta_2\rangle\upharpoonright_{b,v_2} \overset{\text{(IH)}}{\Rightarrow} \neg I_2$$

$$\pi \models B \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{b,v}\Rightarrow (\neg I_1 \wedge \neg I_2) \Leftrightarrow \neg(I_2 \vee I_2)$$

The first implication is a consequence of the two equations above and the second equivalence is factoring out the negation.

(T4.7.Inv3) The third requirement (shared variables only) holds trivially. (We do not add any new variables into the partial interpolant.)

Res-b: $\mathsf{Lab}(v_1,p) \sqcup \mathsf{Lab}(v_2,\overline{p}) = b$. The proof is symmetric to the Res-a case. In this case the pivot variable has the label b in both predecessors v_1 and v_2.

(T4.7.Inv1) The label of the pivot in the predecessor vertices is b so $\neg\langle p\rangle\upharpoonright_{a,v_1}\Leftrightarrow \neg\langle\overline{p}\rangle\upharpoonright_{a,v_2}\Leftrightarrow \top$. Thus Lemma 4.7 can be applied directly without any additional assumptions. The third implication comes from the hypothesis.

$$\pi \models A \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{a,v}\Leftrightarrow \neg\langle p\rangle\upharpoonright_{a,v_1} \wedge A \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{a,v} \overset{\text{(L4.7)}}{\Rightarrow} A \wedge \neg\langle p,\Theta_1\rangle\upharpoonright_{a,v_1} \overset{\text{(IH)}}{\Rightarrow} I_1$$

$$\pi \models A \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{a,v}\Leftrightarrow \neg\langle\overline{p}\rangle\upharpoonright_{a,v_2} \wedge A \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{a,v} \overset{\text{(L4.7)}}{\Rightarrow} A \wedge \neg\langle\overline{p},\Theta_2\rangle\upharpoonright_{a,v_2} \overset{\text{(IH)}}{\Rightarrow} I_2$$

The equations above directly yield the result:

$$\pi \models A \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{a,v}\Rightarrow (I_1 \wedge I_2)$$

(T4.7.Inv2) It follows that:

$$\begin{aligned}\pi \models \overline{p} \wedge B \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{b,v} \overset{\text{(L4.6)}}{\Rightarrow} \neg\langle p\rangle\upharpoonright_{b,v_1} \wedge B \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{b,v} \overset{\text{(L4.7)}}{\Rightarrow}&\\ \Rightarrow B \wedge \neg\langle p,\Theta_1\rangle\upharpoonright_{b,v_1} \overset{\text{(IH)}}{\Rightarrow} \neg I_1&\\ \pi \models p \wedge B \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{b,v} \overset{\text{(L4.6)}}{\Rightarrow} \neg\langle\overline{p}\rangle\upharpoonright_{b,v_2} \wedge B \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{b,v} \overset{\text{(L4.7)}}{\Rightarrow}&\\ \Rightarrow B \wedge \neg\langle\overline{p},\Theta_2\rangle\upharpoonright_{b,v_2} \overset{\text{(IH)}}{\Rightarrow} \neg I_2&\end{aligned}$$

The first implication is application of Lemma 4.6. The second implication is application of Lemma 4.7 and the last one is the induction hypothesis.

From the previous implications, it directly follows that:

$$\begin{aligned}\pi \models B \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{b,v}\Leftrightarrow (\overline{p} \vee p) \wedge B \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{b,v}\Rightarrow&\\ \Rightarrow (\neg I_1 \vee \neg I_2) \Leftrightarrow \neg(I_1 \wedge I_2)&\end{aligned}$$

The first equivalence is a simple logical step due to the fact that $p \vee \overline{p} \Leftrightarrow \top$; the second implication is a consequence of the two equations above and the last equivalence factors out the negation.

Res-ab: $\mathsf{Lab}(v_1,p) \sqcup \mathsf{Lab}(v_2,\overline{p}) = ab$

(T4.7.Inv1) It follows that:

$$\begin{aligned}\pi \models A \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{a,v}\Rightarrow p \vee (\overline{p} \wedge A \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{a,v}) \overset{\text{(L4.6)}}{\Rightarrow}&\\ \Rightarrow p \vee (\neg\langle p\rangle\upharpoonright_{a,v_1} \wedge A \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{a,v}) \overset{\text{(L4.7)}}{\Rightarrow}&\\ \Rightarrow p \vee (A \wedge \neg\langle p,\Theta_1\rangle\upharpoonright_{a,v_1}) \overset{\text{(IH)}}{\Rightarrow} (p \vee I_1)&\\ \pi \models A \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{a,v}\Rightarrow \overline{p} \vee (p \wedge A \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{a,v}) \overset{\text{(L4.6)}}{\Rightarrow}&\\ \Rightarrow \overline{p} \vee (\neg\langle\overline{p}\rangle\upharpoonright_{a,v_2} \wedge A \wedge \neg\langle\Theta_1,\Theta_2\rangle\upharpoonright_{a,v}) \overset{\text{(L4.7)}}{\Rightarrow}&\\ \Rightarrow \overline{p} \vee (A \wedge \neg\langle\overline{p},\Theta_2\rangle\upharpoonright_{a,v_2}) \overset{\text{(IH)}}{\Rightarrow} (\overline{p} \vee I_2)&\end{aligned}$$

The first implication comes from the fact that $p \vee \overline{p} \Leftrightarrow \top$. The second implication is application of Lemma 4.6. The third implication is application of Lemma 4.7 and the last one is the induction hypothesis.

From the previous implications it directly follows that:

$$\pi \models A \wedge \neg\langle \Theta_1, \Theta_2 \rangle\upharpoonright_{a,v}) \Rightarrow (p \vee I_1) \wedge (\overline{p} \vee I_2)$$

(T4.7.Inv2) Similarly to the previous case:

$$\begin{aligned}
\pi \models \overline{p} \wedge B \wedge \neg\langle \Theta_1, \Theta_2 \rangle\upharpoonright_{b,v} &\overset{\text{(L4.6)}}{\Rightarrow} \overline{p} \wedge (\neg\langle p \rangle\upharpoonright_{b,v_1} \wedge B \wedge \neg\langle \Theta_1, \Theta_2 \rangle\upharpoonright_{b,v}) \overset{\text{(L4.7)}}{\Rightarrow} \\
&\Rightarrow \overline{p} \wedge (B \wedge \neg\langle p, \Theta_1 \rangle\upharpoonright_{b,v_1}) \overset{\text{(IH)}}{\Rightarrow} \\
&\Rightarrow \overline{p} \wedge (\neg I_1) \Leftrightarrow \neg(p \vee I_1) \\
\pi \models p \wedge B \wedge \neg\langle \Theta_1, \Theta_2 \rangle\upharpoonright_{b,v} &\overset{\text{(L4.6)}}{\Rightarrow} p \wedge (\neg\langle \overline{p} \rangle\upharpoonright_{b,v_2} \wedge B \wedge \neg\langle \Theta_1, \Theta_2 \rangle\upharpoonright_{b,v}) \overset{\text{(L4.7)}}{\Rightarrow} \\
&\Rightarrow p \wedge (B \wedge \neg\langle \overline{p}, \Theta_2 \rangle\upharpoonright_{b,v_2}) \overset{\text{(IH)}}{\Rightarrow} \\
&\Rightarrow p \wedge (\neg I_2) \Leftrightarrow \neg(\overline{p} \vee I_2)
\end{aligned}$$

The first implication holds since just $\overline{p}$ is duplicated and then Lemma 4.6 is used. The second implication is application of Lemma 4.7, the third one comes from the induction hypothesis and the last one is simply a logical equality.

The same technique as in the Res-a T4.7.Inv1 case is used to prove the second requirement.

$$\begin{aligned}
\pi \models B \wedge \neg\langle \Theta_1, \Theta_2 \rangle\upharpoonright_{b,v} &\Leftrightarrow (\overline{p} \vee p) \wedge B \wedge \neg\langle \Theta_1, \Theta_2 \rangle\upharpoonright_{b,v} \Rightarrow \\
&\Rightarrow \neg(p \vee I_1) \vee \neg(\overline{p} \vee I_2) \Leftrightarrow \neg((p \vee I_1) \wedge (\overline{p} \vee I_2))
\end{aligned}$$

The first equivalence is a simple logical consequence since $p \vee \overline{p} \Leftrightarrow \top$, the second implication is a consequence of the two equations above. The last equivalence just factors out the negation.

(T4.7.Inv3) The only new variable p is added into the interpolant. p is shared, thus the requirements hold. Moreover, the variable p is not assigned. If it would be assigned it would be labeled d^+ in one of the predecessors, which would lead to the Res-d resolution.

Res-d: This case is the resolution step over an assigned pivot variable. From the locality preserving labeling constraint D4.3.1 follows that there is exactly one predecessor where the pivot is labeled d^+. Assume that $\mathsf{Lab}(v_1, p) = d^+$, so it holds $\pi \models p$. The case $\mathsf{Lab}(v_2, \overline{p}) = d^+$ is symmetric.

(T4.7.Inv1) It follows that:

$$\pi \models A \wedge \neg\langle \Theta_1, \Theta_2 \rangle\upharpoonright_{a,v} \Leftrightarrow \neg\langle \overline{p} \rangle\upharpoonright_{a,v_2} \wedge A \wedge \neg\langle \Theta_1, \Theta_2 \rangle\upharpoonright_{a,v} \overset{\text{(L4.7)}}{\Rightarrow} A \wedge \neg\langle \overline{p}, \Theta_2 \rangle\upharpoonright_{a,v_2} \overset{\text{(IH)}}{\Rightarrow} I_2$$

The first equivalence holds because $\pi \models \neg\langle \overline{p} \rangle\upharpoonright_{a,v_2}$, because the $\neg\langle \overline{p} \rangle\upharpoonright_{a,v_2}$ is either directly $\top$ if the $\overline{p}$ literal is not preserved by the filter or it is p if the $\overline{p}$ literal is preserved by the filter $\upharpoonright_{a,v_2}$ and in this case p is satisfied from assumptions.

The second implication is application of Lemma 4.7 and the last one comes from the induction hypothesis.

(T4.7.Inv2) Similarly to the previous case:

$$\pi \models B \wedge \neg\langle \Theta_1, \Theta_2 \rangle\upharpoonright_{b,v} \Leftrightarrow \neg\langle \overline{p} \rangle\upharpoonright_{b,v_2} \wedge B \wedge \neg\langle \Theta_1, \Theta_2 \rangle\upharpoonright_{b,v} \overset{\text{(L4.7)}}{\Rightarrow} B \wedge \neg\langle \overline{p}, \Theta_2 \rangle\upharpoonright_{b,v_2} \overset{\text{(IH)}}{\Rightarrow} \neg I_2$$

(T4.7.Inv3) It holds trivially from the induction hypothesis. (We do not add any new variables into the partial interpolants.)

□

Symmetry. The attentive reader may notice that the locality labeling conditions as well as the way interpolants are computed are *symmetric* for the A_π and B_π sets. It permits us to articulate the *strength* theorem in an elegant way. Given a fixed π, the satisfied clauses can be moved freely between A_π and B_π sets and the locality of the labeling function will be preserved as well as the generated interpolants.

4.2 Strength

A LIS allows one to choose the labels of shared variables. The logical strength of the interpolants generated by two LISs can be compared by comparing the strength of the corresponding labelings. We generalize the notion of strength for partial assignment interpolants. It is not surprising that the *strength ordering* is similar to the one used in [5].

Definition 4.8 (Strength order)**.** *Let $\preceq$ be a pre-order relation defined on the set of labels $S = \{\bot, a, b, ab, d^+\}$ as: $b \preceq ab = d^+ \preceq a \preceq \bot$ (Fig. 4).*

Let Lab *and* Lab' *be labeling functions for a refutation R.* Lab *is stronger than* Lab'*, denoted as* $\mathsf{Lab} \preceq \mathsf{Lab}'$*, if for all vertices $v \in V$ and for all literals $l \in cl(v)$ it holds that* $\mathsf{Lab}(v, l) \preceq \mathsf{Lab}'(v, l)$

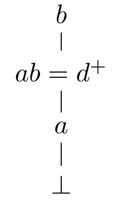

Figure 4: *Strength ordering ($\preceq$)*

Note that label ab and d^+ are of the same strength and can be exchanged if the locality requirements permit.

Weakened-labels filter. Let Lab and Lab' be labeling functions to be compared by strength. For the proofs in this section it is necessary to introduce a new type of filter, which preserves the literals whose labels are weaker in the Lab' labeling. For a vertex $v \in V$, a clause $\langle\Theta\rangle$, and sets of labels $C_1, C_2 \subseteq L$, we define the *label change filter* $||^{\mathsf{Lab},\mathsf{Lab}'}_{v,C_1\Rightarrow C_2}$ as follows: $\langle\Theta\rangle||^{\mathsf{Lab},\mathsf{Lab}'}_{v,C_1\Rightarrow C_2} = \{l \in \Theta \mid \mathsf{Lab}(v, l) \in C_1 \text{ and } \mathsf{Lab}'(v, l) \in C_2\}$.

For the literal being preserved by the filter, the set C_1 specifies permitted literal labels for Lab and the set C_2 specifies permitted labels for the labeling function Lab'.

In the rest of the paper we use a short-cut called *weakened-labels filter* $||\lfloor^{\mathsf{Lab},\mathsf{Lab}}_v$ defined as follows $||\lfloor^{\mathsf{Lab},\mathsf{Lab}'}_v = ||^{\mathsf{Lab},\mathsf{Lab}'}_{v,\{b,ab,d^+\}\Rightarrow\{ab,d^+,a\}}$, which preserves all the literals whose labels are weaker[4] in the primed labeling function according to the strength ordering $\preceq$. The vertex and labeling functions are omitted if clear from the context.

First we show the weaker version of the interpolant strength theorem which assumes that both labeling functions use the same variable assignment π. Later on we will remove this requirement.

Theorem 4.9 (Interpolant strength (weaker version))**.** *Let R be a (A, B, π)-refutation and* $\mathsf{Lab} \preceq \mathsf{Lab}'$ *be locality preserving labeling functions. Let I be a partial assignment interpolant for* $\mathsf{LpaItp}(\mathsf{Lab}, R)$ *and I' be a partial assignment interpolant for* $\mathsf{LpaItp}(\mathsf{Lab}', R)$*. Then $\pi \models I \Rightarrow I'$.*

Before the theorem 4.9 is proved, auxiliary lemmas are shown. The first lemma about the weakened-labels filters is similar to the 4.7 one and it has the same usage. The latter one is used to introduce assignment filters.

Lemma 4.10 (Weakened-labels filters in the predecessor vertices)**.** *Let $\langle\Theta_1, \Theta_2\rangle$ be a clause of an inner vertex v. Let $\langle p, \Theta_1\rangle$ and $\langle\overline{p}, \Theta_2\rangle$ be the clauses of the (positive and negative) predecessors of the vertex v (called v_1 and v_2). Let* Lab *and* Lab' *be labeling functions for the given proof. Then it holds:*

$$\neg\langle p\rangle||\lfloor_{v_1} \wedge \neg\langle\Theta_1, \Theta_2\rangle||\lfloor_v \Rightarrow \neg\langle p, \Theta_1\rangle||\lfloor_{v_1} \qquad \textit{and}$$
$$\neg\langle\overline{p}\rangle||\lfloor_{v_2} \wedge \neg\langle\Theta_1, \Theta_2\rangle||\lfloor_v \Rightarrow \neg\langle\overline{p}, \Theta_2\rangle||\lfloor_{v_2}$$

Proof (Lemma 4.10). First we show that if a literal $l \in \langle\Theta_1, \Theta_2\rangle$ is preserved by the weakened-labels filter in the predecessor vertex v_1 then it is also preserved by the weakened-labels filter in the vertex v where its label is a result of the $\sqcup$ operation (by D4.2.2).

It is easy to see that the sets $\{b, ab, d^+\}$ and $\{ab, d^+, a\}$ (used by the filter $||\lfloor$) are closed under the $\sqcup$ operation. Formally it means that $\forall c \in L$ and $\forall c' \in \{b, ab, d^+\}$ it holds $c \sqcup c' \in \{b, ab, d^+\}$

[4] Note that the weakened-labels filter also preserves some equally strong literals, i.e., those labeled ab or d^+ by both labeling functions.

Let l be preserved by the weakened-labels filter in the vertex v_1. It means that the first labeling function assigns to the literal l a label from the $\{b, ab, d^+\}$ set (formally $\mathsf{Lab}(v_1, l) \in \{b, ab, d^+\}$) and for the second labeling function it holds that $\mathsf{Lab}'(v_1, l) \in \{ab, d^+, a\}$. Because these set are closed under the $\sqcup$ operation, the same holds even in the vertex v (formally $\mathsf{Lab}(v, l) \in \{b, ab, d^+\}$ and $\mathsf{Lab}'(v, l) \in \{ab, d^+, a\}$), thus the literal l is also preserved by the weakened-labels filter in the vertex v.

This gets us that $\langle \Theta_1, \Theta_2 \rangle\!| \,|\!\lfloor_{v_1} \Rightarrow \langle \Theta_1, \Theta_2 \rangle\!| \,|\!\lfloor_{v}$. The implication can be equivalently rewritten into the contrapositive form $\neg\langle \Theta_1, \Theta_2 \rangle\!| |\!\lfloor_{v} \Rightarrow \neg\langle \Theta_1, \Theta_2 \rangle\!| |\!\lfloor_{v_1} \Rightarrow \neg\langle \Theta_1 \rangle\!| |\!\lfloor_{v_1}$.

The final implication $\neg\langle p \rangle\!| \,|\!\lfloor_{v_1} \wedge \neg\langle \Theta_1, \Theta_2 \rangle\!| \,|\!\lfloor_{v} \Rightarrow \neg\langle p, \Theta_1 \rangle\!| \,|\!\lfloor_{v_1}$ with the pivot p holds because the same filter is applied on the pivot (the pivot is either filtered or preserved by both filters).

And symmetrically for the negative child. □

Lemma 4.11 (Introducing the assignment filter)**.** *Let π be a partial variable assignment and $\langle \Theta \rangle$ be a clause. Let the clause is not satisfied by the partial assignment, i.e. $\pi \not\models \langle \Theta \rangle$.*

Then it holds: $\pi \models \langle \Theta \rangle \Leftrightarrow \langle \Theta \rangle[\pi]$.

Proof. It is possible to split the set of literals Θ into two disjoint sets, the set Θ_1 of the literals over the assigned variables (the literals being filtered-out by the assignment filter), and the set Θ_2 of the remaining literals over the not-assigned variables. So $\langle \Theta \rangle \Leftrightarrow \langle \Theta_1 \rangle \vee \langle \Theta_2 \rangle$

From the assumption $\pi \not\models \langle \Theta \rangle$ it follows that all the literals over assigned variables evaluate to $\perp$ under the assignment π, thus it holds $\pi \models \langle \Theta_1 \rangle \Leftrightarrow \perp$. From the definition of the assignment filter it directly follows that $\langle \Theta_2 \rangle \equiv \langle \Theta \rangle[\pi]$.

So finally we get $\pi \models \langle \Theta \rangle \Leftrightarrow \langle \Theta_1 \rangle \vee \langle \Theta_2 \rangle \Leftrightarrow \perp \vee \langle \Theta_2 \rangle \Leftrightarrow \langle \Theta \rangle[\pi]$. □

Proof (Interpolant strength). By structural induction we show that for each vertex v of the resolution proof R the following invariant holds:

$$\pi \models I_v \wedge \neg\langle \Theta \rangle \,|\!\lfloor_v \Rightarrow I'_v$$

where $\langle \Theta \rangle = cl(v)$ is the vertex clause, I_v and I'_v are the partial interpolants for given vertex as generated by our interpolation systems $\mathsf{LpaItp}(\mathsf{Lab}, (A, B, \pi))$ and $\mathsf{LpaItp}(\mathsf{Lab}', (A, B, \pi))$, respectively.

The proof considers all the combinations of rules, that can be used to define partial interpolants I_v and I'_v.

Base cases. The base cases take place in the leaf vertices of the proof. Neither the A and B sets nor the assignment π changed so the type of hypotheses is the same in both interpolants. Moreover the literals labeled d^+ are the same in both labeling functions and assignment filter removes the same literals in both cases.

Hyp-$A_{\overline{\pi}}$: $\langle \Theta \rangle \in A_{\overline{\pi}}$ so $I = \langle \Theta \rangle[\pi]\!|_b$. It holds that:

$$\langle \Theta \rangle\!|_{b,v,\mathsf{Lab}} \wedge \neg\langle \Theta \rangle\!| |\!\lfloor_v \Rightarrow \langle \Theta \rangle\!|_{b,v,\mathsf{Lab}'}$$

because all the literals which are labeled b by the labeling function Lab either lost the label b and then they are preserved by the weakened-labels filter $|\,|\!\lfloor_v$, or have the label b assigned also by the labeling function Lab'. Note that in the first case, due to the locality conditions, the labeling Lab' can assign only labels a or ab to the literal. So from Lemma 4.11 it follows:

$$\pi \models \langle \Theta \rangle[\pi]\!|_{b,v,\mathsf{Lab}} \wedge \neg\langle \Theta \rangle \,|\!\lfloor_v \Rightarrow \langle \Theta \rangle[\pi]\!|_{b,v,\mathsf{Lab}'}$$

The same literals are removed from all the clauses by the assignment filter.

Hyp-$B_{\overline{\pi}}$: $\langle \Theta \rangle \in B_{\overline{\pi}}$ so $I = \neg\langle \Theta \rangle[\pi]\!|_a$. It holds that:

$$\neg\langle \Theta \rangle\!|_{a,v,\mathsf{Lab}} \wedge \neg\langle \Theta \rangle\!| |\!\lfloor_v \Rightarrow \neg\langle \Theta \rangle\!|_{a,v,\mathsf{Lab}'}$$

All the literals which are labeled a by the labeling function Lab' either have the label a assigned also by the labeling function Lab or have a stronger label assigned by Lab in which case the literal is preserved by the weakened-labels filter $|\,|\!\lfloor_v$. Note that in the second case, due to the locality conditions, the labeling Lab can assign only labels b or ab to the literal. So from Lemma 4.11 it follows:

$$\pi \models \neg\langle \Theta \rangle[\pi]\!|_{a,v,\mathsf{Lab}} \wedge \neg\langle \Theta \rangle \,|\!\lfloor_v \Rightarrow \neg\langle \Theta \rangle[\pi]\!|_{a,v,\mathsf{Lab}'}$$

Hyp-A_π, Hyp-B_π: $\langle\Theta\rangle \in A_\pi \cup B_\pi$ so $I = \top$. It holds trivially.

$$\pi \models \top \wedge \neg\langle\Theta\rangle \,||_{v} \Rightarrow \top$$

Inductive step.

Naming conventions. The inductive step takes place in the inner vertices of the proof where resolutions are applied. Due to the label changes, the type/label of the resolution can be different for each interpolant. We will decorate the label in the same way as the labeling function is decorated so label b refers to labeling by the function Lab while ab' refers to the labeling by the function Lab'. Moreover we use this notation in the names of resolution to denote the type of the resolution when computing the first and second interpolant. E.g. Res-b-a' means that in the first interpolant (where Lab is used) the resolution is Res-b, and in the second interpolant (where Lab' is used) the resolution is Res-a.

Invalid combinations. First we show which combinations of the resolutions cannot occur due to the assumption $\mathsf{Lab} \preceq \mathsf{Lab}'$.

There cannot occur Res-a-d'^{+} because in the Res-a case the labels of the pivot in both predecessors has to be a (formally $\mathsf{Lab}(v_1, p) = a$ and $\mathsf{Lab}(v_2, \overline{p}) = a$), while in the second Res-d'^{+} case exactly one pivot literal has to be labeled d'^{+} (formally $\mathsf{Lab}'(v_1, p) = d^{+}$ or $\mathsf{Lab}'(v_2, \overline{p}) = d^{+}$). Note that second fact follows from the structure of the lattice L, because $\sqcup$ results into d^{+} if and only if one of its parameters is d^{+}. Using the facts above it follows that one of the pivots changes the label from a into d'^{+} thus contradicting the fact $\mathsf{Lab} \preceq \mathsf{Lab}'$.

Similar arguments hold for the Res-a-ab' and Res-a-b' cases. In the first case from the Res-ab' it follows that there is a (at least one) pivot labeled either ab' or b'. In the second Res-b' case it holds an even stronger fact – that both pivots are labeled b'. Using the facts above it follows that at least one of the pivots changes the label from a into the stronger label ab' or b', which contradicts the assumption $\mathsf{Lab} \preceq \mathsf{Lab}'$.

Let us consider the Res-ab-b' case. In Res-ab there is at least one pivot labeled either a or ab In the Res-b' both pivots have to be labeled b'. Using the facts above it follows that one of the pivots changes the label from a or ab into the stronger label b' which contradicts the assumption $\mathsf{Lab} \preceq \mathsf{Lab}'$.

Let us consider the Res-d^{+}-b' case. In the Res-d^{+} case exactly one pivot is labeled d^{+}. In Res-b' both pivots have to be labeled b'. Using the facts above it follows that one of the pivots has changed the label from d^{+} into the stronger label b' which contradicts the fact that $\mathsf{Lab} \preceq \mathsf{Lab}'$.

Now we use the assumption that the *same assignment* is used to compute both interpolants. This blocks Res-ab-d'^{+} and Res-d^{+}-ab' resolutions.

It follows directly from the following two facts:

(1) both labeling functions are locality preserving so from the requirement D4.3.1 it follows that the d^{+} labeled literals are the same in both labeling functions, and

(2) the $\sqcup$ operator results into d^{+} if and only if one of its parameters is d^{+} so the Res-d^{+} resolutions are the same in both proofs.

From the facts above it follows that only the Res-d^{+}-d'^{+} case is possible, however, we use the fact only to deny the Res-ab-d'^{+} and Res-d^{+}-ab' resolutions, since the other resolutions involving the label d^{+} are denied also due to the strength requirement.

Induction hypothesis. Be v_1 the positive predecessor of the inner vertex v and v_2 its negative predecessor. Let p be the pivot variable. From the induction hypothesis we know that for the predecessor vertices the invariants hold:

$$\pi \models I_1 \wedge \neg\langle p, \Theta_1\rangle ||_{v_1} \Rightarrow I'_1 \qquad \text{and} \qquad \pi \models I_2 \wedge \neg\langle \overline{p}, \Theta_2\rangle ||_{v_2} \Rightarrow I'_2 \tag{IH}$$

For the rest of permitted resolutions, the induction invariants are established for the vertex v using the induction hypothesis.

Res-a-a': $\mathsf{Lab}(v_1, p) \sqcup \mathsf{Lab}(v_2, \overline{p}) = a$ and $\mathsf{Lab}'(v_1, p) \sqcup \mathsf{Lab}'(v_2, \overline{p}) = a'$

The label of the pivot in both predecessors is a so it is not preserved by the weakened-labels filters $\|\lfloor_{v_1}$ and $\|\lfloor_{v_2}$. (Note that the filters require that the first labeling function assigns a label from the set $\{b, ab, d^+\}$ to the pivot.) Thus $\neg\langle p\rangle\|\lfloor_{v_1}$ equals $\top$.

In this case the following auxiliary implications are needed:

$$\pi \models I_1 \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v \Leftrightarrow \neg\langle p\rangle\|\lfloor_{v_1} \wedge I_1 \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v \overset{L4.10}{\Rightarrow} I_1 \wedge \neg\langle p, \Theta_1\rangle\|\lfloor_{v_1} \overset{IH}{\Rightarrow} I_1'$$
$$\pi \models I_2 \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v \Leftrightarrow \neg\langle\overline{p}\rangle\|\lfloor_{v_2} \wedge I_2 \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v \overset{L4.10}{\Rightarrow} I_2 \wedge \neg\langle\overline{p}, \Theta_2\rangle\|\lfloor_{v_2} \overset{IH}{\Rightarrow} I_2'$$

The first equivalence is shown above, the following implication is application of Lemma 4.10 and the last one is the induction hypothesis.

From the previous implications it directly follows that:

$$\pi \models (I_1 \vee I_2) \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v \Rightarrow (I_1' \vee I_2')$$

Res-b-b': $\mathsf{Lab}(v_1, p) \sqcup \mathsf{Lab}(v_2, \overline{p}) = b$ and $\mathsf{Lab}'(v_1, p) \sqcup \mathsf{Lab}'(v_2, \overline{p}) = b$

The label in both predecessors is b so the pivot is not preserved by the weakened-labels filters $\|\lfloor_{v_1}$ and $\|\lfloor_{v_2}$. (Note that the filters require that the second labeling function Lab' assigns a label from the set $\{ab, d^+, a\}$ to the pivot.) Thus $\neg\langle p\rangle\|\lfloor_{v_1}$ equals $\top$.

Thus the same auxiliary implications as in the previous Res-a-a' case hold:

$$\pi \models I_1 \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v \Leftrightarrow \neg\langle p\rangle\|\lfloor_{v_1} \wedge I_1 \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v \overset{L4.10}{\Rightarrow} I_1 \wedge \neg\langle p, \Theta_1\rangle\|\lfloor_{v_1} \overset{IH}{\Rightarrow} I_1'$$
$$\pi \models I_2 \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v \Leftrightarrow \neg\langle\overline{p}\rangle\|\lfloor_{v_2} \wedge I_2 \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v \overset{L4.10}{\Rightarrow} I_2 \wedge \neg\langle\overline{p}, \Theta_2\rangle\|\lfloor_{v_2} \overset{IH}{\Rightarrow} I_2'$$

The first equivalence is shown above, the following implication is application of Lemma 4.10 and the last one is the induction hypothesis.

From the previous implications it directly follows that:

$$\pi \models (I_1 \wedge I_2) \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v \Rightarrow (I_1' \wedge I_2')$$

Res-ab-ab', Res-ab-a':

Note that in this case the proof does not depend on the labels of the pivot variables, so it can be safely used to show other cases such as Res-ab-a', Res-b-ab', and Res-b-a'. Moreover the case Res-ab-ab' directly implies the Res-ab-a' case.

We have to show that:

$$\pi \models (p \vee I_1) \wedge (\overline{p} \vee I_2) \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v \Rightarrow (p \vee I_1') \wedge (\overline{p} \vee I_2')$$

To show it, the following auxiliary implications are used:

$$\begin{aligned}
\pi \models I_1 \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v \Rightarrow p \vee (\overline{p} \wedge I_1 \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v) \Rightarrow \\
\Rightarrow p \vee (\neg\langle p\rangle\|\lfloor_{v_1} \wedge I_1 \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v) \overset{L4.10}{\Rightarrow} \\
\Rightarrow p \vee (I_1 \wedge \neg\langle p, \Theta_1\rangle\|\lfloor_{v_1}) \overset{IH}{\Rightarrow} p \vee I_1' \\
\pi \models I_2 \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v \Rightarrow \overline{p} \vee (p \wedge I_2 \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v) \Rightarrow \\
\Rightarrow \overline{p} \vee (\neg\langle\overline{p}\rangle\|\lfloor_{v_1} \wedge I_2 \wedge \neg\langle\Theta_1, \Theta_2\rangle\|\lfloor_v) \overset{L4.10}{\Rightarrow} \\
\Rightarrow \overline{p} \vee (I_2 \wedge \neg\langle\overline{p}, \Theta_1\rangle\|\lfloor_{v_2}) \overset{IH}{\Rightarrow} \overline{p} \vee I_2'
\end{aligned}$$

The first implication comes from the fact that $p \vee \overline{p} \Leftrightarrow \top$. The second implication holds because either the literal $\overline{p}$ is preserved by the filter $\|\lfloor_{v_1}$ and then it holds that $\neg\langle p\rangle\|\lfloor_{v_1} \Leftrightarrow \overline{p}$, or the literal $\overline{p}$ is not preserved by the weakened-labels filter and then it holds that $\neg\langle p\rangle\|\lfloor_{v_1} \Leftrightarrow \top$. The third implication is application of Lemma 4.10 and the last one comes from the induction hypothesis.

Now the proof is split into three cases. It holds that:

$$(p \vee I_1) \wedge (\overline{p} \vee I_2) \Leftrightarrow (p \wedge I_2) \vee (\overline{p} \wedge I_1) \vee (I_1 \wedge I_2)$$

Using the above auxiliary implications, we proof that each case leads to the formula $(p \vee I'_1) \wedge (\overline{p} \vee I'_2)$.

$$\begin{aligned}
\pi &\models (\overline{p} \wedge I_1) \wedge \neg\langle\Theta_1,\Theta_2\rangle|\lfloor_v \Rightarrow \overline{p} \wedge (p \vee I'_1) && \Rightarrow (p \vee I'_1) \wedge (\overline{p} \vee I'_2)\\
\pi &\models (p \wedge I_2) \wedge \neg\langle\Theta_1,\Theta_2\rangle|\lfloor_v \Rightarrow p \wedge (\overline{p} \vee I'_2) && \Rightarrow (p \vee I'_1) \wedge (\overline{p} \vee I'_2)\\
\pi &\models (I_1 \wedge I_2) \wedge \neg\langle\Theta_1,\Theta_2\rangle|\lfloor_v && \Rightarrow (p \vee I'_1) \wedge (\overline{p} \vee I'_2)
\end{aligned}$$

The first implication is application of the above auxiliary implication. The second implication (if present) is a simple logical consequence.

Getting everything together we have showed that:

$$\pi \models (p \vee I_1) \wedge (\overline{p} \vee I_2) \wedge \neg\langle\Theta_1,\Theta_2\rangle|\lfloor_v \Rightarrow (p \vee I'_1) \wedge (\overline{p} \vee I'_2)$$

Res-b-ab', Res-b-a':

The Res-b-ab' holds because $I_1 \wedge I_2 \Rightarrow (p \vee I_1) \wedge (\overline{p} \vee I_2)$ so the Res-ab-ab' case can be directly used. Moreover the case Res-b-ab' directly implies the Res-b-a' case.

Res-d^+-d'^+:

Assume that $\mathsf{Lab}(v_1, p) = d^+$ thus it holds that $\pi \models p$. The case $\mathsf{Lab}(v_2, \overline{p}) = d^+$ is symmetric.

It holds that $\pi \models \neg\langle\overline{p}\rangle|\lfloor_{v_2}$. $\neg\langle\overline{p}\rangle|\lfloor_{v_2}$ is either $\top$ (if the literal $\overline{p}$ is filtered out by the weaken filter) or p (if the literal is not filtered out). In the latter case the assumption $\pi \models p$ is used.

Using the fact above it follows that:

$$\pi \models I_2 \wedge \neg\langle\Theta_1,\Theta_2\rangle|\lfloor_v \Leftrightarrow \neg\langle\overline{p}\rangle|\lfloor_{v_2} \wedge I_2 \wedge \neg\langle\Theta_1,\Theta_2\rangle|\lfloor_v \overset{L4.10}{\Rightarrow} I_2 \wedge \neg\langle\overline{p},\Theta_2\rangle|\lfloor_{v_2} \overset{IH}{\Rightarrow} I'_2$$

The first equivalence is explained above, the second implication follows from Lemma 4.10 and the last one is the induction hypothesis.

Also note that only the Res-d^+-d'^+ case and the hypothesis (via Lemma 4.11) use the partial variable assignment assumption (the fact that $\pi \models p$). □

Moreover the notion of strength can be extended to different partial assignments when preserving the A and B sets as stated in the following theorem.

Theorem 4.12 (Interpolant strength). *Let* Lab *be a locality preserving labeling function for the (A, B, π)-refutation R, and* Lab' *be a locality preserving labeling function for (A, B, π')-refutation R. Let I be a partial variable assignment interpolant for* $\mathsf{LpaItp}(\mathsf{Lab}, R)$ *and I' be a PVAI for* $\mathsf{LpaItp}(\mathsf{Lab}', R)$.

If $\mathsf{Lab} \preceq \mathsf{Lab}'$ *then* $\pi, \pi' \models I \Rightarrow I'$.

Note that when π and π' are *empty* assignments, we obtain exactly the theorem on interpolant strength from [5]. Also note that the theorem permits different variable assignments for the interpolants. Thus it relates the interpolants even between different sub-problems (e.g., interpolants considering different possible paths to a given program location). Since both π and π' are in assumptions of the formula $I \Rightarrow I'$, the theorem applies to cases common to both sub-problems (e.g., to the shared paths). *Proof.* We will extend the previous proof to cover new cases arisen from partial assignment changes. We use structural induction and also (nearly) the same invariant.

For each vertex v of the resolution proof R the following invariant holds:

$$\pi, \pi' \models I_v \wedge \neg\langle\Theta\rangle|\lfloor_v \Rightarrow I'_v$$

where $\langle\Theta\rangle = cl(v)$ is the vertex clause, I_v and I'_v are the partial interpolants for the given vertex as generated by our interpolation systems $\mathsf{LpaItp}(\mathsf{Lab}, R)$ and $\mathsf{LpaItp}(\mathsf{Lab}', R)$, respectively.

Base cases. The sets A and B are the same for both interpolants I and I'. So a clause from A can only move between the satisfied clause set A_π and the unsatisfied $A_{\overline{\pi}}$. The same holds for the B_π and $B_{\overline{\pi}}$ sets.

The Hyp-$A_{\overline{\pi}}$-$A_{\overline{\pi}'}$, Hyp-A_π-$A_{\pi'}$, Hyp-$B_{\overline{\pi}}$-$B_{\overline{\pi}'}$ and Hyp-B_π-$B_{\pi'}$ cases have been shown in the previous proof, hence the idea is applied here as well.

Hyp-$A_{\overline{\pi}}$-$A_{\overline{\pi}'}$: $\langle\Theta\rangle \in A_{\overline{\pi}}, \langle\Theta\rangle \in A_{\overline{\pi}'}, I_v = \langle\Theta\rangle[\pi]\!]_{b,v,\mathsf{Lab}}$ and $I'_v = \langle\Theta\rangle[\pi']\!]_{b,v,\mathsf{Lab}'}$
From the previous proof we know that:

$$\langle\Theta\rangle\!]_{b,v,\mathsf{Lab}} \wedge \neg\langle\Theta\rangle\!|\!\downharpoonleft_v \Rightarrow \langle\Theta\rangle\!]_{b,v,\mathsf{Lab}'}$$

Using Lemma 4.11 the invariant is derived:

$$\pi, \pi' \models \langle\Theta\rangle[\pi]\!]_{b,v,\mathsf{Lab}} \wedge \neg\langle\Theta\rangle\!|\!\downharpoonleft_v \Rightarrow \langle\Theta\rangle[\pi']\!]_{b,v,\mathsf{Lab}'}$$

Hyp-$B_{\overline{\pi}}$-$B_{\overline{\pi}'}$: $\langle\Theta\rangle \in B_{\overline{\pi}}, \langle\Theta\rangle \in B_{\overline{\pi}'}, I_v = \neg\langle\Theta\rangle[\pi]\!]_{a,v,\mathsf{Lab}}$ and $I'_v = \neg\langle\Theta\rangle[\pi']\!]_{a,v,\mathsf{Lab}'}$
From the previous proof we know that:

$$\neg\langle\Theta\rangle\!]_{a,v,\mathsf{Lab}} \wedge \neg\langle\Theta\rangle\!|\!\downharpoonleft_v \Rightarrow \neg\langle\Theta\rangle\!]_{a,v,\mathsf{Lab}'}$$

Using Lemma 4.11 the invariant is derived:

$$\neg\langle\Theta\rangle[\pi]\!]_{a,v,\mathsf{Lab}} \wedge \neg\langle\Theta\rangle\!|\!\downharpoonleft_v \Rightarrow \neg\langle\Theta\rangle[\pi']\!]_{a,v,\mathsf{Lab}'}$$

Hyp-A_{π}-$A_{\overline{\pi}'}$, Hyp-B_{π}-$B_{\overline{\pi}'}$: $I_v = \top$. The proof does not depend on the I'_v.

We show that the assumption of the invariant $\pi, \pi' \models I_v \wedge \neg\langle\Theta\rangle\!|\!\downharpoonleft_v$ cannot be satisfied, thus the invariant holds. The clause $\langle\Theta\rangle$ is in the A_π or B_π set so there exists a literal $l \in \langle\Theta\rangle$ such that $\pi \models l$. From the locality of the labeling function (D4.3.1) it follows that $\mathsf{Lab}(v, l) = d^+$. Moreover, from $\mathsf{Lab} \preceq \mathsf{Lab}'$ it follows that in the second interpolant the literal l can have only a weaker label so $\mathsf{Lab}'_{v,l} \in \{d^+, ab, a\}$. Thus the literal l is preserved be the $|\!\downharpoonleft_v$ filter. Formally it holds that $l \Rightarrow \langle\Theta\rangle\!|\!\downharpoonleft_v$ and so $\neg\langle\Theta\rangle\!|\!\downharpoonleft_v \Rightarrow \neg l$.

The literal $\neg l$ must hold to satisfy the assumptions of the invariant. However in any model $\neg l$ evaluates to $\bot$ (since $\pi \models l$). Thus the invariant holds since its assumptions cannot be satisfied.

Hyp-$A_{\overline{\pi}}$-$A_{\pi'}$, Hyp-$B_{\overline{\pi}}$-$B_{\pi'}$: $I'_v = \top$. The proof does not depend on the I_v.

The cases hold trivially since anything implies $\top$.

Inductive step. Since all the resolution steps in the previous proof except the Res-d^+-d^+ does not depend on the partial assignment π, they can be reused in this proof immediately. Moreover since our composed assumptions π, π' are more restricting than the one in the previous proof, even the case Res-d^+-d^+ holds directly and can be reused.

In this case, we do not have the requirement on the same partial assignment, so we cannot use the arguments to block the Res-ab-d'^+ and the Res-d^+-ab' resolutions, which are shown bellow.

Res-ab-d'^+: $\mathsf{Lab}(v_1, p) \sqcup \mathsf{Lab}'(v_2, \overline{p}) = ab$ and $\mathsf{Lab}(v_1, p) \sqcup \mathsf{Lab}'(v_2, \overline{p}) = d'^+$
Assume that $\mathsf{Lab}'_{v_1,p} = d^+$ thus $\pi' \models p$. The situation is symmetric in $\mathsf{Lab}'_{v_2,\overline{p}} = d^+$. In this case we need to show that:

$$\pi, \pi' \models (p \vee I_1) \wedge (\overline{p} \vee I_2) \wedge \neg\langle\Theta_1, \Theta_2\rangle\!|\!\downharpoonleft_v \Rightarrow I'_2$$

From the proof of the Res-ab-ab' case (which can be shown without any additional assumption on the labels of the pivot) it holds:

$$\pi, \pi' \models (p \vee I_1) \wedge (\overline{p} \vee I_2) \wedge \neg\langle\Theta_1, \Theta_2\rangle\!|\!\downharpoonleft_v \Rightarrow (p \vee I'_1) \wedge (\overline{p} \vee I'_2)$$

Due to the fact that $\pi' \models p$ it holds:

$$\pi, \pi' \models (p \vee I'_1) \wedge (\overline{p} \vee I'_2) \Leftrightarrow I'_2$$

The two implications above if connected yield the result as needed.

Res-d^{+}-ab': $\mathsf{Lab}(v_1,p) \sqcup \mathsf{Lab}'(v_2,\overline{p}) = d^{+}$ and $\mathsf{Lab}(v_1,p) \sqcup \mathsf{Lab}'(v_2,\overline{p}) = ab$

Assume that $\mathsf{Lab}(v_1,p) = d^{+}$ thus $\pi \models p$. The situation is symmetric in $\mathsf{Lab}_{v_2,\overline{p}} = d^{+}$. The proof is similar to the previous case Res-ab-d'^{+}. First, the assumption $\pi \models p$ is used to introduce the partial ab-interpolant and then the Res-ab-ab' case is reused. In this case we need to show that:

$$\pi, \pi' \models I_2 \wedge \neg\langle\Theta_1, \Theta_2\rangle\!\|\!\downharpoonleft_v \Rightarrow (p \vee I_1') \wedge (\overline{p} \vee I_2')$$

Due to the fact that $\pi' \models p$ it holds:

$$\pi, \pi' \models I_2 \Leftrightarrow (p \vee I_1) \wedge (\overline{p} \vee I_2)$$

From the proof of the Res-ab-ab' case (which can be shown without any additional assumption on the labels of the pivot) it holds:

$$\pi, \pi' \models (p \vee I_1) \wedge (\overline{p} \vee I_2) \wedge \neg\langle\Theta_1, \Theta_2\rangle\!\|\!\downharpoonleft_v \Rightarrow (p \vee I_1') \wedge (\overline{p} \vee I_2')$$

The two implications above if connected yield the result as needed.

□

5 Path interpolation property

Many verification approaches such as [2, 8, 12] depend on the *path interpolation* property (PI). In [11] the authors show that LISs can be employed to generate an interpolation sequence by providing a sequence of labeling functions that are non strictly decreasing in terms of strength. First, the PI property is shown to hold if the same partial assignment along the sequence is used to compute the interpolants (i.e., solving the same sub-problem). Later on, the result is generalized to permit different partial assignments for particular interpolants.

5.1 Fixed partial assignment.

To show the interpolation sequence property, it is enough to prove that $I \wedge S \Rightarrow I'$ (*inductive step*), where I is an interpolant for $(A, S \cup B, \pi)$, I' is an interpolant for $(A \cup S, B, \pi)$, and S is a set of clauses.

For LISs, Rollini et al. [11] define the labeling constraints on the labeling functions used to compute the interpolants I and I'. If the labeling constraints are satisfied, the interpolants satisfy the PI property. Due to the complexity of labeling constraints for PVAI, we describe it in a different way. Given a labeling function used to compute the interpolant I, we define the strongest labeling function which can be used to compute the successor interpolant I'.

Definition 5.1 (Strongest successor labeling). *Let* Lab *be a labeling function for the* $(A, S \cup B, \pi)$*-refutation* R*. The* strongest successor labeling Lab^{S} *(induced by the set* S*) is defined as follows:* $\forall v \in V$ *and* $\forall l \in cl(v)$

$$\mathsf{Lab}^{S}(v,l) = \begin{cases} a & \textit{if}\, \mathsf{Var}(l) \in \mathsf{Var}(S_{\overline{\pi}}) \wedge \mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\pi}}) \wedge \\ & \qquad\qquad\qquad\qquad \wedge \mathsf{Var}(l) \notin \mathsf{Var}(\pi) \qquad (D5.1.1) \\ \mathsf{Lab}(v,l) & \textit{otherwise} \qquad\qquad\qquad\qquad\qquad\quad\; (D5.1.2) \end{cases}$$

Note that Def. 5.1 defines a partial function (in the undefined points – the cases if a literal l is not present in the vertex clause – $\mathsf{Lab}^{S}(v,l) = \bot$). The labeling Lab^{S} can be used to compute the interpolant for $(A \cup S, B, \pi)$. The first alternative (D5.1.1) forces the label a for all literals which are $(A_{\overline{\pi}} \cup S_{\overline{\pi}})$-local due to the shift of the clauses in S to the A part.

First we show that a valid labeling function is defined. The requirement D4.2.1 is trivial (no label is defined if $l \notin cl(v)$). To show the requirement D4.2.2, we make a simple observation. The first alternative (D5.1.1) does not depend on the vertex being considered. Thus a given variable is consistently labeled in all the vertices either by the first alternative or by the second one. The requirement holds trivially for the first alternative since $(a \sqcup a) = (a \sqcup \bot) = (\bot \sqcup a) = a$. In the latter case the requirement D4.2.2 holds, because Lab is a valid labeling function.

Now we show that Lab^{S} preserves the locality property. The fact that Lab^{S} is locality preserving is important since only the locality labeling functions are used to compute interpolants.

Lemma 5.2 (Locality of strongest successor labeling). *Let* Lab *be a* locality preserving *labeling function for (*A*,* $S \cup B$*,* π*)-refutation* R*. Let* Lab^{S} *be the strongest successor labeling function induced by* S*.*

Then Lab^{S} *is a* locality preserving *labeling for (*$A \cup S$*,* B*,* π*)-refutation* R*.*

Proof. We show that all the locality requirements are satisfied.

Satisfied literals – the requirement D4.3.1.

Let $\pi \models l$ then we know that the second alternative is used so $\mathsf{Lab}^S(v,l) = \mathsf{Lab}(v,l)$. Moreover locality of Lab gives us that $\mathsf{Lab}^S(v,l) = \mathsf{Lab}(v,l) = d^+$. On the other hand if $\pi \not\models l$ then if the first alternative (D5.1.1) is used then the label a is assigned and the requirement $D4.3.1$ holds; if the second alternative (D5.1.2) is used then, again, locality of Lab directly implies that the label d^+ is not assigned.

$(A_{\overline{\pi}} \cup S_{\overline{\pi}})$-locality – the requirement D4.3.2.

We want to show that if the variable is $(A_{\overline{\pi}} \cup S_{\overline{\pi}})$-local then the label a is assigned to it. Formally we want to show that:

$$\begin{aligned} \mathsf{Var}(l) \notin \mathsf{Var}(\pi) \wedge \mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\pi}} \cup S_{\overline{\pi}}) \wedge \mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\pi}}) \Rightarrow& \\ \Rightarrow \mathsf{Lab}^S(v,l) = a& \end{aligned} \tag{1}$$

If the first alternative (D5.1.1) applies then the requirement is satisfied trivially, since it sets the label a as required. If the second alternative (D5.1.2) is used then either the variable is $A_{\overline{\pi}}$-local and then from locality of Lab it follows that label a is assigned or the variable was not $A_{\overline{\pi}}$-local. In the latter case the variable cannot be $(A_{\overline{\pi}} \cup S_{\overline{\pi}})$-local so this requirement does not impose any restriction on the label.

Now we formally show that if a variable is $(A_{\overline{\pi}} \cup S_{\overline{\pi}})$-local and the second alternative is used then the labeling function Lab assigns the label a. Since the first alternative (D5.1.1) is not used then one of the assumption must be violated; it is either:

(L5.2.Inv1) $\mathsf{Var}(l) \notin \mathsf{Var}(S_{\overline{\pi}})$, or

(L5.2.Inv2) $\mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\pi}})$, or

(L5.2.Inv3) $\mathsf{Var}(l) \notin \mathsf{Var}(\pi)$

In the first case (L5.2.Inv1), it holds:

$$\begin{aligned} \mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\pi}} \cup S_{\overline{\pi}}) &\overset{(L5.2.Inv1)}{\Rightarrow} \mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\pi}}) \\ \mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\pi}}) &\overset{(L5.2.Inv1)}{\Rightarrow} \mathsf{Var}(l) \notin \mathsf{Var}(S_{\overline{\pi}} \cup B_{\overline{\pi}}) \end{aligned}$$

So the variable is $A_{\overline{\pi}}$-local and due to locality of the labeling function Lab it assigns the label a to that variable.

In the second case (L5.2.Inv2) the condition $\mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\pi}})$ directly contradicts the assumptions of the equation (1), and thus there is no requirement to assign the label a (since the variable is not $(A_{\overline{\pi}} \cup S_{\overline{\pi}})$-local) in this case. In the third case (L5.2.Inv3) as in the previous case, the condition $\mathsf{Var}(l) \notin \mathsf{Var}(\pi)$ directly contradicts the assumptions of the equation (1), so again there is no requirement to assign the label a (since the variable is assigned and thus it cannot be $(A_{\overline{\pi}} \cup S_{\overline{\pi}})$-local) in this case.

$B_{\overline{\pi}}$-locality – the requirement D4.3.3.

We want to show that if the variable is $B_{\overline{\pi}}$-local then the label b is assigned. Formally we want to show that:

$$\begin{aligned} \mathsf{Var}(l) \notin \mathsf{Var}(\pi) \wedge \mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\pi}}) \wedge \mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\pi}} \cup S_{\overline{\pi}}) \Rightarrow& \\ \Rightarrow \mathsf{Lab}^S(v,l) = b& \end{aligned} \tag{2}$$

Informally, if the first alternative (D5.1.1) is used, the variable is not $B_{\overline{\pi}}$-local, thus this requirement does not apply in this case. Moreover for the second alternative the $B_{\overline{\pi}}$-local variables are also the $(B_{\overline{\pi}} \cup S_{\overline{\pi}})$-local variables, so the locality of the labeling function Lab gives us that the assigned label is b.

Formally, from the assumptions of the first alternative (D5.1.1) we know that $\mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\pi}})$ which directly contradicts the assumptions of this case so there is no requirement on the label in this case. Let the second alternative (D5.1.2) is used set the label; without any additional assumptions it holds:

$$\begin{aligned} \mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\pi}}) &\Rightarrow \mathsf{Var}(l) \in \mathsf{Var}(S_{\overline{\pi}} \cup B_{\overline{\pi}}) \\ \mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\pi}} \cup S_{\overline{\pi}}) &\Rightarrow \mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\pi}}) \end{aligned}$$

So the variable is also $(B_{\overline{\pi}} \cup S_{\overline{\pi}})$-local and locality of the labeling function Lab gives us that Lab assigns the label b to the variable as needed.

Satisfied clauses – the requirement D4.3.4.

We want to show that the $(A_{\overline{\pi}} \cup S_{\overline{\pi}})B_{\overline{\pi}}$-clean variables ($\pi$-local variables) have the label a or b assigned consistently. Formally we want to show that:

$$\begin{aligned}&\text{if } \mathsf{Var}(l) \notin \mathsf{Var}(\pi) \wedge \mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\pi}} \cup S_{\overline{\pi}}) \wedge \mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\pi}})\\ &\qquad\qquad\qquad\text{then } \forall v' \in V, \forall l' \in cl(v') : \mathsf{Var}(l) = \mathsf{Var}(l') \Rightarrow\\ &\qquad\qquad\qquad\qquad\qquad\Rightarrow \mathsf{Lab}^S(v,l) = \mathsf{Lab}^S(v',l') \in \{a,b\}\end{aligned}$$

If the first alternative (D5.1.1) is used then the variable is not $(A_{\overline{\pi}} \cup S_{\overline{\pi}})B_{\overline{\pi}}$-clean ($\pi$-local), because $\mathsf{Var}(l) \in \mathsf{Var}(S_{\overline{\pi}})$ so $\mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\pi}} \cup S_{\overline{\pi}})$. So this requirement does not apply in this case. Let the second alternative (D5.1.2) is used. Informally, since the same assignment is used for both functions Lab^S and Lab, the π-local variables are the same. Thus the consistent assignment of the permitted label comes from the locality of the labeling function Lab. Formally, since the first alternative is not used then one of the assumptions L5.2.Inv1-3 must be violated. In the second (L5.2.Inv2) and third (L5.2.Inv3) cases the variable is not $(A_{\overline{\pi}} \cup S_{\overline{\pi}})B_{\overline{\pi}}$-clean, thus the requirement does not apply in these cases. In the first case (L5.2.Inv1) it holds that:

$$\begin{aligned}\mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\pi}} \cup S_{\overline{\pi}}) &\Rightarrow \mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\pi}})\\ \mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\pi}}) &\overset{(L5.2.Inv1)}{\Rightarrow} \mathsf{Var}(l) \notin \mathsf{Var}(S_{\overline{\pi}} \cup B_{\overline{\pi}})\end{aligned}$$

So the variable is $A_{\overline{\pi}}(S_{\overline{\pi}} \cup B_{\overline{\pi}})$-clean and locality of the labeling function Lab gives us that Lab assigns the label a or b consistently. Moreover the first (D5.1.1) or second (D5.1.1) alternative is used consistently for the variable across the whole proof. (This follows from the condition of the first alternative which is independent of the proof vertex where the label is assigned). Thus even Lab^S assigns the label a or b to the variable consistently as needed. □

Looking at Def. 5.1, we can see that $\mathsf{Lab} \preceq \mathsf{Lab}^S$, because the labels are either equal or the weakest label a is used in the labeling Lab^S.

The following lemma states the PI property for the strongest successor labeling.

Lemma 5.3. *Let* Lab *be a locality preserving labeling function for a (A, S∪B, π)-refutation R and let* $\mathsf{Lpaltp}(\mathsf{Lab}, R) = I$*. Let* Lab^S *be the strongest successor labeling of* Lab *induced by S and* $\mathsf{Lpaltp}(\mathsf{Lab}^S, (A \cup S, B, \pi)) = I'$*.*

Then $\pi \models I \wedge S \Rightarrow I'$*.*

Proof. By structural induction over the resolution proof we show that for each vertex $v \in V$ of the refutation proof the following invariant holds:

$$\pi \models I \wedge S \wedge \neg\langle\Theta\rangle|\lfloor_v \Rightarrow I'$$

where $\langle\Theta\rangle = cl(v)$ is the vertex clause, I_v and I'_v are the partial interpolants for the vertex v as generated by our interpolation system using the labeling functions Lab and Lab^S, respectively.

Bases cases. The base cases take place in the leaf vertices of the proof where the hypotheses operations are applied. Because the partial variable assignment π is the same for both interpolants, the only possible shifts are between (un)satisfied clause sets. It is not possible to shift the unsatisfied clause from $B_{\overline{\pi}}$ into the satisfied set B_{π}.

Hyp-$A_{\overline{\pi}}$-$(A_{\overline{\pi}} \cup S_{\overline{\pi}})$: $\langle\Theta\rangle \in A_{\overline{\pi}}$, so $I_v = \langle\Theta\rangle[\pi]\!]_{b,v,\mathsf{Lab}}$ and $I'_v = \langle\Theta\rangle[\pi]\!]_{b,v,\mathsf{Lab}^S}$.

At first we will show that:

$$\langle\Theta\rangle|_{b,v,\mathsf{Lab}} \wedge \neg\langle\Theta\rangle|\lfloor_v \Rightarrow \langle\Theta\rangle|_{b,v,\mathsf{Lab}^S}$$

Let a literal be labeled b by the labeling Lab and let it loose the label b (Lab^S assigns a different label to it). So the literal is removed by the match$|_{b,v,\mathsf{Lab}^S}$ filter. Such a literal has assigned the label a (using the first alternative (D5.1.1) in the definition of Lab^S), thus the literal is preserved by the weakened-labels filter $|\,|\lfloor_v$ (which is negated in the formula above and so guarantees that the literal does not satisfy the clause $\langle\Theta\rangle|_{b,v,\mathsf{Lab}^S}$).

The assignment filter $[\pi]$ is applied to reduce the set of literals being considered. It holds that:

$$\langle\Theta\rangle[\pi]|_{b,v,\mathsf{Lab}} \wedge \neg\langle\Theta\rangle|\downharpoonleft_v \Rightarrow \langle\Theta\rangle[\pi]|_{b,v,\mathsf{Lab}^S}$$

This follows from the fact that the assignment filter $[\pi]$ removes the same literals from both partial interpolants. The implication above is even stronger than the invariant, since it does not require S in the assumptions of the implication.

Hyp-($S_{\overline{\pi}} \cup B_{\overline{\pi}}$)-$B_{\overline{\pi}}$: $\langle\Theta\rangle \in B_{\overline{\pi}}$ so $I = \neg\langle\Theta\rangle[\pi]|_{a,v,\mathsf{Lab}}$ and $I' = \neg\langle\Theta\rangle[\pi]|_{a,v,\mathsf{Lab}^S}$.
This case is similar to the previous one. First we show that:

$$\neg\langle\Theta\rangle|_{a,v,\mathsf{Lab}} \wedge \neg\langle\Theta\rangle|\downharpoonleft_v \Rightarrow \neg\langle\Theta\rangle|_{a,v,\mathsf{Lab}^S}$$

Let the literal be labeled a by the labeling Lab^S (so it is preserved by the match filter$|_{a,v,\mathsf{Lab}^S}$). The literal is either labeled a by the labeling Lab or not (in which case its label can be b or ab). In the former case the literal is preserved by the match filter$|_{a,v,\mathsf{Lab}}$. In the latter case the literal is preserved by the weakened-labels filter $|\downharpoonleft_v$. To sum it up, all the literals in the conjunction after the implication occur even in the conjunction(s) before the implication thus the implication holds.

The assignment filter $[\pi]$ is applied to reduce the set of literals being considered. It holds that:

$$\neg\langle\Theta\rangle[\pi]|_{a,v,\mathsf{Lab}} \wedge \neg\langle\Theta\rangle|\downharpoonleft_v \Rightarrow \neg\langle\Theta\rangle[\pi]|_{a,v,\mathsf{Lab}^S}$$

This again follows from the fact that the assignment filter $[\pi]$ removes the same literals from both partial interpolants. The implication above is even stronger than the invariant, since it does not require S in the assumptions of the implication.

Hyp-($S_{\overline{\pi}} \cup B_{\overline{\pi}}$)-($A_{\overline{\pi}} \cup S_{\overline{\pi}}$): $\langle\Theta\rangle \in S_{\overline{\pi}}$ so $I = \neg\langle\Theta\rangle[\pi]|_{a,v,\mathsf{Lab}}$ and $I' = \langle\Theta\rangle[\pi]|_{b,v,\mathsf{Lab}^S}$.
First we show that in this case it holds that:

$$\langle\Theta\rangle \Leftrightarrow \langle\Theta\rangle|_{a,v,\mathsf{Lab}} \vee \langle\Theta\rangle|_{b,v,\mathsf{Lab}^S} \vee \langle\Theta\rangle|\downharpoonleft_v$$

The $\Leftarrow$ implication is trivial, since filters only remove literals, so if the right-hand side of the equivalence holds, the unfiltered clauses $\langle\Theta\rangle$ must also hold. The $\Rightarrow$ implication is shown bellow. We consider all the combinations of labels the literal can get by the labeling functions Lab and Lab^S. The literal in the clause $\langle\Theta\rangle$ is either preserved by:

- the match filter $|_{a,v,\mathsf{Lab}}$ if the label is a in both labeling functions, or
- the match filter $|_{b,v,\mathsf{Lab}^S}$ if the label is b in both labeling functions, or
- the weakened-label filter $|\downharpoonleft_v$ if the first alternative (D5.1.1) is used and the weaker label a or alternatively the labels ab or d^+ are set by both labeling functions.

Note that the remaining label changes not covered above (which increase the strength – $a \Rightarrow \{b', ab', d'^+\}$ and $\{ab, d^+\} \Rightarrow b'$) are not possible (since we have shown that $\mathsf{Lab} \preceq \mathsf{Lab}^S$).

Moreover from Lemma 4.11 it follows that:

$$\pi \models \langle\Theta\rangle \Leftrightarrow \langle\Theta\rangle[\pi]|_{a,v,\mathsf{Lab}} \vee \langle\Theta\rangle[\pi]|_{b,v,\mathsf{Lab}^S} \vee \langle\Theta\rangle|\downharpoonleft_v$$

The literal removed by the assignment filter $[\pi]$ evaluates to $\perp$ given the partial variable assignment π since the clause $\langle\Theta\rangle \in S_{\overline{\pi}}$ cannot contain any satisfied literal.

The invariant is show bellow.

$$\begin{array}{rccr}
\pi \models & I \quad \wedge & S & \wedge \neg\langle\Theta\rangle|\downharpoonleft_v \equiv \\
& \equiv \neg\langle\Theta\rangle[\pi]|_{a,v,\mathsf{Lab}} \wedge & S & \wedge \neg\langle\Theta\rangle|\downharpoonleft_v \Rightarrow \\
& \Rightarrow \neg\langle\Theta\rangle[\pi]|_{a,v,\mathsf{Lab}} \wedge & \langle\Theta\rangle & \wedge \neg\langle\Theta\rangle|\downharpoonleft_v \Leftrightarrow \\
& \Leftrightarrow \neg\langle\Theta\rangle[\pi]|_{a,v,\mathsf{Lab}} \wedge & (\langle\Theta\rangle[\pi]|_{a,v,\mathsf{Lab}} \vee \langle\Theta\rangle[\pi]|_{b,v,\mathsf{Lab}^S} & \vee \langle\Theta\rangle|\downharpoonleft_v) \wedge \neg\langle\Theta\rangle|\downharpoonleft_v \Leftrightarrow \\
& \Leftrightarrow \neg\langle\Theta\rangle[\pi]|_{a,v,\mathsf{Lab}} \wedge & \langle\Theta\rangle[\pi]|_{b,v,\mathsf{Lab}^S} & \wedge \neg\langle\Theta\rangle|\downharpoonleft_v \Rightarrow \\
& & \Rightarrow \langle\Theta\rangle[\pi]|_{b,v,\mathsf{Lab}^S} & \equiv I'
\end{array}$$

The first implication follows from the fact that $\langle\Theta\rangle \in S$ and S is a conjunction (a set) of clauses so $S \Rightarrow \langle\Theta\rangle$. The second equivalence is shown above. The third equivalence is a logical consequence. The following pattern is used twice: $\neg A \wedge (A \vee B) \Leftrightarrow (\neg A \wedge A) \vee (\neg A \wedge B) \Leftrightarrow \neg A \wedge B$, where $A \equiv \neg\langle\Theta\rangle[\pi]|_{a,v,\mathsf{Lab}}$ and $A \equiv \neg\langle\Theta\rangle|\downharpoonleft_v$, respectively. The last implication is a trivial logical consequence.

Hyp-A_π-$(A_\pi \cup S_\pi)$, Hyp-$(S_\pi \cup B_\pi)$-B_π, Hyp-$(S_\pi \cup B_\pi)$-$(A_\pi \cup S_\pi)$: All the cases with satisfied clauses. In these cases $I' = \top$ so the invariant holds trivially.

Induction hypothesis. Now the invariant is shown for the inner vertices of the proof where the resolution operation is applied.

First, we focus on the types of resolutions which may occur. From the fact that $\mathsf{Lab} \preceq \mathsf{Lab}^S$ it directly follows that labels of the pivots are the same or weaker in the computation of the interpolant I' than in the interpolant I. Thus the types of the resolution are the same or equal in the case of I' interpolant. Moreover since in both cases we use the same partial variable assignment π, the d^+ labels and thus Res-d^+ are used at the same time in both interpolants.

Our induction hypothesis is stronger than (and thus implies) the induction hypothesis from the proof of the interpolant-strength Theorem 4.9 so it is possible to directly use the resolutions from there.

The proofs for all the weakening resolutions (Res-b-$\{b', ab', a'\}$, Res-ab-$\{ab', a'\}$, Res-a-a' and Res-d^+-d'^+) are in Theorem 4.9. Note that proofs for the resolutions in the theorem are independent of A and B sets – only the labels are considered by the proof. Also note that all the other resolutions have either a stronger resolution in the computation of the I' interpolant (Res-a-$\{b', ab'\}$ and Res-ab-b') or the Res-d^+ resolutions change in the I' computation (Res-$\{a, ab, b\}$-d'^+ and Res-d^+-$\{a', ab', b'\}$) so they violates the observation shown above.

□

Now we formally explain the meaning of the *strongest* adjective from the definition. To generalize Lemma 5.3 for other labeling functions, the following lemma is introduced.

Lemma 5.4. *Let* Lab *be a locality preserving labeling function for* $(A, S \cup B, \pi)$*-refutation* R*. Let* Lab' *be a locality preserving labeling for* $(A \cup S, B, \pi)$*-refutation* R*.*

If $\mathsf{Lab} \preceq \mathsf{Lab}'$ *then also* $\mathsf{Lab}^S \preceq \mathsf{Lab}'$*.*

Proof. We show the lemma by contradiction. For contradiction, assume that $\mathsf{Lab} \preceq \mathsf{Lab}'$ and $\mathsf{Lab}^S \npreceq \mathsf{Lab}'$. That means there exists a vertex $v \in V$ and a literal $l \in cl(v)$ such that $\mathsf{Lab}^S(v, l) \npreceq \mathsf{Lab}'(v, l)$.

The label $\mathsf{Lab}^S(v, l)$ is assigned either by the first (D5.1.1) or second (D5.1.2) alternative. In the former case, the literal l is $(A_{\overline{\pi}} \cup S_{\overline{\pi}})$-local so it must have the label a. But the $\mathsf{Lab}'(v, l)$ is strictly greater than a which contradicts locality of the function Lab'. Stated formally, because the first alternative is used we know that $\mathsf{Var}(l) \notin \mathsf{Var}(\pi)$ and $\mathsf{Var}(l) \in \mathsf{Var}(S_{\overline{\pi}}) \subseteq \mathsf{Var}(A_{\overline{\pi}} \cup S_{\overline{\pi}})$ and $\mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\pi}})$ so all the requirements for locality preserving rule D4.3.2 are preserved. Thus if $\mathsf{Lab}'(v, l) \neq a$ as required in this case, Lab' is not locality preserving labeling function for $(A \cup S, B, \pi)$.

The situation is simple in the latter case if the label is set by the second alternative. Then, Lab' violates the assumptions because $\mathsf{Lab}(v, l) = \mathsf{Lab}^S(v, l) \npreceq \mathsf{Lab}'(v, l)$; it shows that Lab' is not weaker than Lab (formally $\mathsf{Lab} \npreceq \mathsf{Lab}'$). □

The following theorem states the main result for cases of a fixed partial assignment.

Theorem 5.5 (Inductive step). *Let* Lab *and* Lab' *be locality preserving labeling functions for the (A,* $S \cup B$*,* π*)-refutation* R *and (*$A \cup S$*, B,* π*)-refutation* R*, respectively. Let* $\mathsf{LpaItp}(\mathsf{Lab}, R) = I$ *and* $\mathsf{LpaItp}(\mathsf{Lab}', R) = I'$*.*

If $\mathsf{Lab} \preceq \mathsf{Lab}'$ *then* $\pi \models I \wedge S \Rightarrow I'$*.*

Proof. Let I^S be the partial variable interpolant for the strongest successor labeling function Lab^S. From Lemma 5.3 it holds that $\pi \models I \wedge S \Rightarrow I^S$. From Lemma 5.4 we know that $\mathsf{Lab}^S \preceq \mathsf{Lab}'$ and from Theorem 4.12 it follows that $\pi \models I^S \Rightarrow I'$. □

5.1.1 Multiple partial assignments.

Now we generalize Theorem 5.5 to cases where different partial assignments π and π' are used for computation of interpolants I and I', respectively. Then, the desired result is $\pi, \pi' \models I \wedge S \Rightarrow I'$.

Assignments. Having two (different) PVA π and π', the expression (π, π') represents the PVA formed by the union of the π and π' assignments. We say that PVA σ is an *extension* of PVA π, if $\sigma \Rightarrow \pi$ (if the PVAs are viewed as conjunctions of literals). In other words, σ can be created from π by assigning additional variables. In the case of conflicting π and π' (assigning one $\top$ and the other $\bot$ to a particular variable), the inductive-step formula above holds trivially and therefore the case can be omitted from now on.

Definition 5.6. *We say that the variable is* assignable *if it is McMillan-labeled and not $A_{\overline{\pi}}$-local.*

The idea of our approach is to split the problem into the following three simpler steps:

(1)	$(A, S \cup B, \quad \pi \quad) \rightarrow (A, S \cup B, (\pi, \quad \pi' \quad))$	Add assignment π'
(2)	$(A, S \cup B, (\quad \pi, \pi') \quad) \rightarrow (A \cup S, B, (\pi, \quad \pi' \quad))$	PI for fixed assignment
(3)	$(A \cup S, B, (\quad \pi, \pi') \quad) \rightarrow (A \cup S, B, \quad \pi' \quad)$	Remove assignment π

In the first and the last step Theorem 4.12 is used to relate particular interpolants with each other via implications, while in the second step Theorem 5.5 is utilized. To be able to apply this scheme, we have to prove that the hypotheses of the theorems hold – the existence of the locality preserving labeling functions of decreasing strength. The approach taken is similar to the one used for a fixed variable assignment; we define the *extended-assignment labeling* for the first step and the *restricted-assignment labeling* for the third step.

Definition 5.7 (Extended-assignment labeling)**.** *Let* Lab *be a labeling function for a (A, B, π)-refutation R and a partial variable assignment σ be an extension of π. The* extended-assignment labeling $\mathsf{Lab}^{+}_{\pi\rightarrow\sigma}$ *is defined as follows: $\forall v \in V$ and $\forall l \in cl(v)$:* $\mathsf{Lab}^{+}_{\pi\rightarrow\sigma}(v, l) =$

$$= \begin{cases} d^{+} & \textit{if } \pi \not\models l \wedge \sigma \models l & (D5.7.1) \\ a & \textit{if } \mathsf{Var}(l) \textit{ is unassigned by } \sigma \textit{ and } A_{\overline{\sigma}}\textit{-local} & (D5.7.2) \\ b & \textit{if } \mathsf{Var}(l) \textit{ is unassigned by } \sigma, B_{\overline{\sigma}}\textit{-local and not } B_{\overline{\pi}}\textit{-local} & (D5.7.3) \\ a & \textit{if } \mathsf{Var}(l) \textit{ is unassigned by } \sigma, A_{\overline{\sigma}}B_{\overline{\sigma}}\textit{-clean and} & \\ & \qquad \exists \textit{ vertex } v' \textit{ where literal } l \textit{ or } \neg l \textit{ has label } a \textit{ or } ab & (D5.7.4) \\ \mathsf{Lab}(v, l) & \textit{otherwise} & (D5.7.5) \end{cases}$$

Formally:

$$\begin{cases} d^{+} & \textit{if } \pi \not\models l \wedge \sigma \models l & (D5.7.1) \\ a & \textit{if } \mathsf{Var}(l) \notin \mathsf{Var}(\sigma) \wedge \mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\sigma}}) \wedge \mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\sigma}}) & (D5.7.2) \\ b & \textit{if } \mathsf{Var}(l) \notin \mathsf{Var}(\sigma) \wedge \mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\sigma}}) \wedge \mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\sigma}}) \wedge & \\ & \qquad \wedge \mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\pi}}) & (D5.7.3) \\ a & \textit{if } \mathsf{Var}(l) \notin \mathsf{Var}(\sigma) \wedge \mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\sigma}}) \wedge \mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\sigma}}) \wedge & \\ & \qquad \exists v' \in V, l' \in cl(v') \textit{ such that} & \\ & \qquad\qquad \mathsf{Var}(l) = \mathsf{Var}(l') \wedge \mathsf{Lab}(v', l') \in \{a, ab\} & (D5.7.4) \\ \mathsf{Lab}(v, l) & \textit{otherwise} & (D5.7.5) \end{cases}$$

In a similar way as in the strongest successor labeling, if a literal l is not present in the vertex clause, then $\mathsf{Lab}^{+}_{\pi\rightarrow\sigma}(v, l) = \bot$. The idea behind the definition is to create a locality preserving labeling for (A, B, σ)-refutation. Each of the first four alternatives satisfies the corresponding locality preserving constraint (D4.3.1-D4.3.4) for newly-occurring cases due to assignment extension.

The first alternative (D5.7.1) covers the newly satisfied literals. The second alternative (D5.7.2) covers the new $A_{\overline{\sigma}}$-local (as well as the old $A_{\overline{\pi}}$-local) variables, while the third alternative (D5.7.3) covers the new $B_{\overline{\sigma}}$-local variables. The fourth alternative (D5.7.4) covers the case when due to the new assignment the variable becomes $A_{\overline{\sigma}}B_{\overline{\sigma}}$-clean and the label a must be assigned (because the variable is not consistently labeled b).

First we show that we have defined a valid labeling function. The $D4.2.1$ rule is trivial (no label is defined if $l \notin cl(v)$). As for $D4.2.2$, in all the alternatives their conditions do not depend on the vertex being considered, so the rule is used consistently for the given literal in the whole proof. For all but the last alternative the assigned labels are constant. For any label c it holds that $c \sqcup c = \bot \sqcup c = c$ so the property $D4.2.2$ holds trivially. For the last alternative the property $D4.2.2$ holds, because Lab is a valid labeling function.

Lemma 5.8 (Locality of extended-assignment labeling)**.** *Let* Lab *be a* locality *preserving labeling function for a (A, B, π)-refutation R and let a partial variable assignment σ be an extension of π.*

Then the extended-assignment labeling $\mathsf{Lab}^{+}_{\pi\rightarrow\sigma}$ *is a* locality preserving labeling *for the (A, B, σ)-refutation R.*

Proof. Due to the newly assigned variables in σ, the clauses from the unsatisfied clauses set can be removed (formally $A_{\overline{\sigma}} \subseteq A_{\overline{\pi}}$ and $B_{\overline{\sigma}} \subseteq B_{\overline{\pi}}$). We show that all the locality requirements are satisfied.

Satisfied literals – the requirement D4.3.1.

Let $\sigma \models l$; we need to show that $\mathsf{Lab}^{+}_{\pi\to\sigma}(v,l) = d^{+}$. It either holds $\pi \not\models l$ (a newly assigned variable) or $\pi \models l$. In the former case the first alternative (D5.7.1) is applied and the label d^{+} is assigned as needed. In the latter case the last alternative (D5.7.5) is applied and the label d^{+} is assigned due to the locality of the Lab function.

Let $\sigma \not\models l$; we need to show that $\mathsf{Lab}^{+}_{\pi\to\sigma}(v,l) \neq d^{+}$. In such a case the first alternative (D5.7.1) cannot be applied. The second, third and fourth alternatives satisfy it trivially since different labels a or b are assigned. Let the last alternative is used. It holds that $\pi \not\models l$ since σ is an extension of π. So the locality of Lab gives us that the label is not d^{+} as required.

$A_{\overline{\sigma}}$-locality – the requirement D4.3.2.

Let the variable $\mathsf{Var}(l)$ is $A_{\overline{\sigma}}$-local so the labeling function $\mathsf{Lab}^{+}_{\pi\to\sigma}$ must assign the label a to it. In such a case exactly the second alternative (D5.7.2) applies.

$B_{\overline{\sigma}}$-locality – the requirement D4.3.3.

Let the variable $\mathsf{Var}(l)$ is $B_{\overline{\sigma}}$-local so the labeling function must assign the label b to it. It can be either even $B_{\overline{\pi}}$-local (in which case $\mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\pi}})$) or it is not a $B_{\overline{\pi}}$-local variable and then $\mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\pi}})$. In the former case the last alternative (D5.7.5) applies and the locality of the function Lab guaranties that the label b is assigned. Note that the first alternative (D5.7.1) cannot be applied since $B_{\overline{\sigma}}$-local implies $\mathsf{Var}(l) \notin \mathsf{Var}(\pi) \subseteq \mathsf{Var}(\sigma)$. The second (D5.7.2) and fourth (D5.7.4) alternatives cannot be applied because $\mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\sigma}})$. And finally the third alternative cannot be applied since $B_{\overline{\pi}}$-locality is assumed in this case. In the latter case when the variable is newly $B_{\overline{\sigma}}$-local the third alternative (D5.7.3) applies and the assigned label is b.

Satisfied clauses – the requirement D4.3.4.

Let the variable $\mathsf{Var}(l)$ is $A_{\overline{\sigma}}B_{\overline{\sigma}}$-clean. In such a case it must be labeled a or b consistently across the whole proof. If the variable is $A_{\overline{\sigma}}B_{\overline{\sigma}}$-clean then either fourth (D5.7.4) or the last (D5.7.5) alternative applies The fact that $\mathsf{Var}(l) \notin \mathsf{Var}(\sigma)$ (which comes from D4.3.4) denies the first alternative (D5.7.1). $\mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\sigma}})$ blocks the second (D5.7.2) alternative and symmetrically $\mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\sigma}})$ blocks the third (D5.7.3) alternative.

Moreover from the condition $\mathsf{Var}(l) \notin \mathsf{Var}(\sigma)$ it follows that the label d^{+} is not assigned in any vertex clause to (positive and negative literal over) the variable $\mathsf{Var}(l)$. Let the variable has the label b assigned by Lab to all the its occurrences (i.e., consistently). In such a case the last alternative applies and the requirement is satisfied. In all the other cases the fourth alternative applies and the label is consistently set to a in all the vertex clauses containing the variable $\mathsf{Var}(l)$ and the requirement is satisfied again. □

A similar property as for the strongest successor labeling holds for the extended-assignment labeling.

Lemma 5.9. *Let* Lab *and* Lab' *be locality preserving labeling functions for a (A,B,π)-refutation R and a (A,B,σ)-refutation R, respectively. Let a partial variable assignment σ be an extension of π.*

If $\mathsf{Lab} \preceq \mathsf{Lab}'$ *then also* $\mathsf{Lab}^{+}_{\pi\to\sigma} \preceq \mathsf{Lab}'$.

Proof. We show the lemma by contradiction. For contradiction assume that $\mathsf{Lab} \preceq \mathsf{Lab}'$ and $\mathsf{Lab}^{+}_{\pi\to\sigma} \not\preceq \mathsf{Lab}'$. That means there exists a vertex $v \in V$ and a literal $l \in cl(v)$ such that $\mathsf{Lab}^{+}_{\pi\to\sigma}(v,l) \not\preceq \mathsf{Lab}'(v,l)$. It especially means that $\mathsf{Lab}^{+}_{\pi\to\sigma}(v,l) \neq \mathsf{Lab}'(v,l)$.

Let the label $\mathsf{Lab}^{+}_{\pi\to\sigma}(v,l)$ is set by the first alternative (D5.7.1) so $\mathsf{Lab}^{+}_{\pi\to\sigma}(v,l) = d^{+}$. In such a case it holds $\mathsf{Lab}'(v,l) \neq d^{+}$ and Lab' is not locality preserving since it directly violates the rule D4.3.1. Let the label is set by the second alternative (D5.7.2) so $\mathsf{Lab}^{+}_{\pi\to\sigma}(v,l) = a$. In such a case it holds $\mathsf{Lab}'(v,l) \neq a$ and Lab' is not locality preserving since it directly violates the rule D4.3.2. Let the label is set by the third alternative (D5.7.3) so $\mathsf{Lab}^{+}_{\pi\to\sigma}(v,l) = b$. In such a case holds $\mathsf{Lab}'(v,l) \neq b$ and Lab' is not locality preserving since violates the rule D4.3.3.

Let the label is set by the fourth (D5.7.4) alternative so $\mathsf{Lab}^{+}_{\pi\to\sigma}(v,l) = a$. Be a vertex v' and a literal l' witness from the alternative definition such that $\mathsf{Lab}(v',l') \in \{a, ab\}$.

The assumption of the alternative (D5.7.4) directly satisfies the assumption of the rule D4.3.4 so even locality labeling function $\mathsf{Lab}'(v,l)$ must return either a or b. The first case cannot occur since v and l are chosen so that $a = \mathsf{Lab}'(v,l) \neq \mathsf{Lab}^{+}_{\pi\to\sigma}(v,l) = a$. The latter case, so let $\mathsf{Lab}'(v,l) = b$, must apply. It either holds that $\mathsf{Lab}'(v',l') = b$ or not. Let $\mathsf{Lab}'(v',l') = b$ holds, then $\{a,ab\} \ni \mathsf{Lab}(v',l') \not\preceq \mathsf{Lab}'(v',l') = b$ so it holds $\mathsf{Lab} \not\preceq \mathsf{Lab}'$ and the assumptions of the lemma are violated. In the latter case if $\mathsf{Lab}'(v',l') \neq b$ then the v', l' parameters are the witnesses that Lab' is not locality preserving since it violates the locality rule D4.3.4.

Let the label is set by the last alternative (D5.7.5) so $\mathsf{Lab}^{+}_{\pi\to\sigma}(v,l) = \mathsf{Lab}(v,l)$. In such a case the vertex v and the literal l also show that $\mathsf{Lab} \not\preceq \mathsf{Lab}'$ so the assumptions of the lemma are violated. □

The strongest successor labeling is weaker than the original labeling function. However, this might not be the case of the extended-assignment labeling where the alternatives D5.7.1 and D5.7.3 may increase the strength. Thus additional requirements are introduced to ensure this property.

Lemma 5.10. *Let* Lab *be a locality preserving labeling function for* (A, B, π)*-refutation* R*. Let a partial variable assignment* σ *be an extension of* π*. If the clause sets* $A_{\overline{\pi}}$ *and* $A_{\overline{\sigma}}$ *are equal and all newly* σ*-assigned variables (not assigned by* π*) are assignable in* Lab *then:*

$$\mathsf{Lab} \preceq \mathsf{Lab}^{+}_{\pi\to\sigma}$$

The requirements of Lemma 5.10 can be still relaxed; we decided to articulate them in this way, since they do not limit the applicability of the lemma yet they are simple enough.

Proof sketch. From the requirement $A_{\overline{\pi}} = A_{\overline{\sigma}}$ it follows that $B_{\overline{\sigma}}$-local variables are also $B_{\overline{\pi}}$-local, so the D5.7.3 alternative cannot be used. The newly σ-assigned variables (to which the first alternative (D5.7.1) applies) are not $A_{\overline{\pi}}$-local and for all the other possible cases ($B_{\overline{\pi}}$-local, $A_{\overline{\pi}}B_{\overline{\pi}}$-shared, or $A_{\overline{\pi}}B_{\overline{\pi}}$-clean) the locality and the McMillan labeling property imply that they are consistently labeled b in the original labeling Lab. Thus, the (D5.7.1) alternative assigns a weaker label d^+ in $\mathsf{Lab}^{+}_{\pi\to\sigma}$. All remaining alternatives cannot increase the strength of the label thus $\mathsf{Lab} \preceq \mathsf{Lab}^{+}_{\pi\to\sigma}$. □

Proof. To proof that $\mathsf{Lab} \preceq \mathsf{Lab}^{+}_{\pi\to\sigma}$ it is necessary to show that $\forall v \in V$ and $\forall l \in cl(v)$ it holds that $\mathsf{Lab}(v, l) \preceq \mathsf{Lab}^{+}_{\pi\to\sigma}(v, l)$. The label $\mathsf{Lab}^{+}_{\pi\to\sigma}(v, l)$ must be defined by one of the alternatives (D5.7.1-5) from the definition of the extended-assignment labeling.

Let the first alternative (D5.7.1) is used to define the label so $\mathsf{Lab}^{+}_{\pi\to\sigma}(v, l) = d^+$. In the following we show that $\mathsf{Lab}(v, l) = b$ and $\mathsf{Lab}^{+}_{\pi\to\sigma}(v, l)$ assigns a weaker label as needed. First, the variable $\mathsf{Var}(l)$ is not assigned by π. For contradiction, if $\mathsf{Var}(l)$ were assigned by π, it would hold that $\pi \models \neg l$. The other polarity $\pi \models l$ violates the assumptions of the D5.7.1 alternative. Because σ is an extension of π, it would hold $\sigma \models \neg l$. However the assumption of the D5.7.1 alternative requires $\sigma \models l$. In such a case it would hold that the partial variable assignment σ assigns to the variable $\mathsf{Var}(l)$ both $\top$ and $\bot$ which is not possible. So the variable $\mathsf{Var}(l)$ is assigned by σ and not by π; it is newly-assigned and thus $\mathsf{Var}(l)$ is assignable.

The proof of the D5.7.1 alternative splits into four cases according to the type of the unassigned variable $\mathsf{Var}(l)$, which can be $A_{\overline{\pi}}$-local, $B_{\overline{\pi}}$-local, $A_{\overline{\pi}}B_{\overline{\pi}}$-shared or $A_{\overline{\pi}}B_{\overline{\pi}}$-clean. Let $\mathsf{Var}(l)$ is:

(L5.10.1-1) $A_{\overline{\pi}}$-local, so $\mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\pi}})$ and $\mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\pi}})$.
This case cannot occur, because $\mathsf{Var}(l)$ is assignable and thus not $A_{\overline{\pi}}$-local.

(L5.10.1-2) $B_{\overline{\pi}}$-local, so $\mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\pi}})$ and $\mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\pi}})$.
From the locality requirement D4.3.3 it follows that $\mathsf{Lab}(v, l) = b$ as we need.

(L5.10.1-3) $A_{\overline{\pi}}B_{\overline{\pi}}$-shared, so $\mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\pi}})$ and $\mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\pi}})$.
Because the variable $\mathsf{Var}(l)$ is assignable and thus McMillan-labeled, it follows that $\mathsf{Lab}(v, l) = b$ as we need.

(L5.10.1-4) $A_{\overline{\pi}}B_{\overline{\pi}}$-clean, so $\mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\pi}})$ and $\mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\pi}})$.
Because the variable $\mathsf{Var}(l)$ is assignable and thus McMillan-labeled, it follows that $\mathsf{Lab}(v, l) = b$ as we need.

Let the second alternative (D5.7.2) is used so $\mathsf{Lab}^{+}_{\pi\to\sigma}(v, l) = a$. The requirement holds trivially in this case, since the label a is the weakest one according to the strength ordering $\preceq$. (The label $\bot$ is not possible since $l \in cl(v)$.)

The third alternative (D5.7.3) cannot be used. We show that all $B_{\overline{\sigma}}$-local variables are also $B_{\overline{\pi}}$-local. Let the variable $\mathsf{Var}(l)$ is $B_{\overline{\sigma}}$-local, formally $\mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\sigma}})$ and $\mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\sigma}})$. From the assumption $A_{\overline{\sigma}} = A_{\overline{\pi}}$ it follows that $\mathsf{Var}(A_{\overline{\sigma}}) = \mathsf{Var}(A_{\overline{\pi}})$. Because σ is an extension of π it follows that $B_{\overline{\sigma}} \subseteq B_{\overline{\pi}}$ so $\mathsf{Var}(B_{\overline{\sigma}}) \subseteq \mathsf{Var}(B_{\overline{\pi}})$. The above inclusions show the $B_{\overline{\pi}}$-locality of the variable $\mathsf{Var}(l)$ (formally $\mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\sigma}}) = \mathsf{Var}(A_{\overline{\pi}})$ and $\mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\sigma}}) \subseteq \mathsf{Var}(B_{\overline{\pi}})$).

Let the fourth (D5.7.4) alternative is used so $\mathsf{Lab}^{+}_{\pi\to\sigma}(v, l) = a$. The requirement holds trivially in this case, since the label a is the weakest one according to the strength ordering $\preceq$. (The label $\bot$ is not possible since $l \in cl(v)$.)

Let the last alternative (D5.7.5) is used then the requirement holds trivially. (It holds that $\mathsf{Lab}^{+}_{\pi\to\sigma}(v, l) = \mathsf{Lab}(v, l)$ and for any label c holds $c \preceq c$.)

□

Now we consider the case when assignment of some variables is removed.

Definition 5.11 (Restricted-assignment labeling). *Let* Lab *be a labeling function for a* $(A,\ B,\ \pi)$*-refutation* R*. Let* σ *be a partial variable assignment such that* π *is an extension of* σ*. The* restricted-assignment labeling $\mathsf{Lab}^-_{\pi\to\sigma}$ *is defined as follows:*$\forall v \in V$ *and* $\forall l \in cl(v)$*:* $\mathsf{Lab}^-_{\pi\to\sigma}(v,l) =$

$$
= \begin{cases}
ab & \textit{if } \pi \models l \textit{ and } \mathsf{Var}(l) \textit{ is unassigned by } \sigma \textit{ and } A_{\overline{\sigma}}B_{\overline{\sigma}}\textit{-shared} & (D5.11.1)\\
a & \textit{if } \mathsf{Var}(l) \textit{ is assigned by } \pi\textit{, unassigned by } \sigma\textit{, } A_{\overline{\sigma}}B_{\overline{\sigma}}\textit{-clean and} & \\
 & \qquad \exists \textit{ vertex } v' \textit{ where literal } l \textit{ or } \neg l \textit{ has label } a, ab \textit{ or } d^+ & (D5.11.2)\\
a & \textit{if } \mathsf{Var}(l) \textit{ is unassigned by } \sigma \textit{ and } A_{\overline{\sigma}}\textit{-local} & (D5.11.3)\\
b & \textit{if } \mathsf{Var}(l) \textit{ is unassigned by } \sigma \textit{ and } B_{\overline{\sigma}}\textit{-local} & (D5.11.4)\\
\mathsf{Lab}(v,l) & \textit{otherwise} & (D5.11.5)
\end{cases}
$$

Formally:

$$
= \begin{cases}
ab & \textit{if } \pi \models l \wedge \mathsf{Var}(l) \notin \mathsf{Var}(\sigma) \wedge & \\
 & \qquad \wedge \mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\sigma}}) \wedge \mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\sigma}}) & (D5.11.1)\\
a & \textit{if } \mathsf{Var}(l) \in \mathsf{Var}(\pi) \wedge \mathsf{Var}(l) \notin \mathsf{Var}(\sigma) \wedge & \\
 & \qquad \wedge \mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\sigma}}) \wedge \mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\sigma}}) & \\
 & \qquad\qquad \wedge \exists v' \in V, l' \in cl(v') \textit{ such that} & \\
 & \qquad\qquad\qquad \mathsf{Var}(l) = \mathsf{Var}(l') \wedge \mathsf{Lab}(v',l') \in \{a, ab, d^+\} & (D5.11.2)\\
a & \textit{if } \mathsf{Var}(l) \notin \mathsf{Var}(\sigma) \wedge \mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\sigma}}) \wedge \mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\sigma}}) & (D5.11.3)\\
b & \textit{if } \mathsf{Var}(l) \notin \mathsf{Var}(\sigma) \wedge \mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\sigma}}) \wedge \mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\sigma}}) & (D5.11.4)\\
\mathsf{Lab}(v,l) & \textit{otherwise} & (D5.11.5)
\end{cases}
$$

In a similar way as in the extended-assignment labeling, if a literal l is not present in the vertex clause, then $\mathsf{Lab}^-_{\pi\to\sigma}(v,l) = \perp$. The idea behind the definition is to create a locality preserving labeling for an (A,B,σ)-refutation.

The first two alternatives (D5.11.1-2) assign labels to the variables whose assignment is removed. The third and fourth rules cover new $A_{\overline{\sigma}}$-local and $B_{\overline{\sigma}}$-local variables.

First we show that we have defined a valid labeling function. The $D4.2.1$ rule is trivial (no label is defined if $l \notin cl(v)$). As for $D4.2.2$ rule, in all previous cases the alternative used to set the label does not depend on the vertex being considered. For all but the last alternative the assigned labels are constant. For a label c it holds that $c \sqcup c = \perp \sqcup c = c$ so the rule holds trivially. For the last alternative the property holds, because Lab is a valid labeling function.

Lemma 5.12 (Locality of restricted-assignment labeling). *Let* Lab *be a* locality *preserving labeling function for a* (A,B,π)*-refutation* R *and* π *be an extension of a partial variable assignment* σ*.*

Then the restricted-assignment labeling $\mathsf{Lab}^-_{\pi\to\sigma}$ *is a* locality *preserving labeling for the* (A,B,σ)*-refutation* R*.*

Proof. Due to removal of variables from the assignment, some satisfied clauses can become unsatisfied (formally $A_{\overline{\pi}} \subseteq A_{\overline{\sigma}}$ and $B_{\overline{\pi}} \subseteq B_{\overline{\sigma}}$). We show that all the locality requirements are satisfied.

Satisfied literals – the requirement D4.3.1.

Assume that $\sigma \models l$. Then only the last alternative (D5.11.5) can be used since $\mathsf{Var}(l)$ is assigned by σ. Moreover since the assignment π is an extension of σ, it holds that $\pi \models l$. So the locality of Lab gives us that $\mathsf{Lab}^-_{\pi\to\sigma}(v,l) = \mathsf{Lab}(v,l) = d^+$ as needed.

Let $\sigma \not\models l$ then it we need to show that $\mathsf{Lab}^-_{\pi\to\sigma}(v,l) \neq d^+$. It either holds that $\sigma \models \neg l$ (i.e. the literal l is falsified) or $\mathsf{Var}(l)$ is unassigned by σ. In the former case the last alternative (D5.11.5) is used, since all other alternatives require $\mathsf{Var}(l)$ to be unassigned by σ. Because σ is restriction of π, from the fact that $\sigma \models \neg l$ it follows that $\pi \models \neg l$, thus $\pi \not\models l$ (partial variable assignment assigns to $\mathsf{Var}(l)$ only a single value – $\top$ or $\perp$). From the locality of Lab we know that it returns d^+ if and only if $\pi \models l$ so in this case $\mathsf{Lab}^-_{\pi\to\sigma}(v,l) \neq d^+$.

Now focus on the latter case, i.e. $\mathsf{Var}(l)$ is unassigned by σ. The proof splits into four cases according to the type of the unassigned variable $\mathsf{Var}(l)$. Let $\mathsf{Var}(l)$ is:

(L5.12.1-1) $A_{\overline{\sigma}}$-local, so $\mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\sigma}})$ and $\mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\sigma}})$.

In this case the D5.11.3 alternative is used so a label different from d^+ is assigned as needed.

(L5.12.1-2) $B_{\overline{\sigma}}$-local, so $\mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\sigma}})$ and $\mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\sigma}})$.

In this case the D5.11.4 alternative is used so a label different from d^+ is assigned as needed.

(L5.12.1-3) $A_{\overline{\sigma}}B_{\overline{\sigma}}$-shared, so $\mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\sigma}})$ and $\mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\sigma}})$.
In this case it either holds that $\pi \models l$ or $\pi \not\models l$. In the former case the first alternative (D5.11.1) is used and the label ab is assigned. In the latter case the last alternative (D5.11.5) is used. From the locality of the Lab and the fact that $\pi \not\models l$ it follows that $\mathsf{Lab}(v,l) \neq d^+$ as needed.

(L5.12.1-4) $A_{\overline{\sigma}}B_{\overline{\sigma}}$-clean, so $\mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\sigma}})$ and $\mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\sigma}})$.
In this case either the second alternative (D5.11.2) or the last alternative (D5.11.5) is used. In the former case a label different from d^+ is assigned as needed. In the latter case the alternative D5.11.5 is used to set the label. Since the second alternative is not used it either holds that $\mathsf{Var}(l)$ is unassigned by π ($\mathsf{Var}(l) \notin \mathsf{Var}(\pi)$) or all instances of the variable $\mathsf{Var}(l)$ are labeled consistently by b ($\forall v' \in V, l' \in cl(v')$ such that $\mathsf{Var}(l) = \mathsf{Var}(l') \wedge \mathsf{Lab}(v',l') \notin \{a, ab, d^+\}$). Let $\mathsf{Var}(l)$ is unassigned by π ($\mathsf{Var}(l) \notin \mathsf{Var}(\pi)$), it follows that $\pi \not\models l$ so the locality of Lab gives us that $\mathsf{Lab}(v,l) \neq d^+$ as needed. In the latter case, if the variable $\mathsf{Var}(l)$ is labeled b consistently by Lab, the property holds trivially.

$A_{\overline{\sigma}}$-locality – the requirement D4.3.2.
Assume that the variable l is $A_{\overline{\sigma}}$-local. The locality preserving labeling function must assign the label a here. In such a case the third alternative (D5.11.3) applies.

$B_{\overline{\sigma}}$-locality – the requirement D4.3.3.
Assume that the variable l is $B_{\overline{\sigma}}$-local-local. The locality preserving labeling function must assign the label b. In such a case the fourth alternative (D5.11.4) applies.

Satisfied clauses – the requirement D4.3.4.
Assume that the variable $\mathsf{Var}(l)$ is $A_{\overline{\sigma}}B_{\overline{\sigma}}$-clean. The locality preserving labeling function must consistently assign the label a or b to $\mathsf{Var}(l)$. If the variable is $A_{\overline{\sigma}}B_{\overline{\sigma}}$-clean, None of the first (D5.11.1), the third (D5.11.3), and the fourth (D5.11.4) alternative can be used to set its label. Formally, if the first (D5.11.1) or the third (D5.11.3) alternative is used, the variable $\mathsf{Var}(l)$ is not $A_{\overline{\sigma}}B_{\overline{\sigma}}$-clean because $\mathsf{Var}(l) \in A_{\overline{\sigma}}$. If the fourth (D5.11.4) alternative is used, the variable $\mathsf{Var}(l)$ is not $A_{\overline{\sigma}}B_{\overline{\sigma}}$-clean because $\mathsf{Var}(l) \in B_{\overline{\sigma}}$.

Assume that the second alternative (D5.11.2) is used to set the label of the variable $\mathsf{Var}(l)$. Then the requirement is satisfied since the permitted label a is assigned and the alternative is used for all the occurrences of the variable in the proof.

Assume that the last alternative (D5.11.5) is used to set the label. The variable $\mathsf{Var}(l)$ is either unassigned by π or assigned by π. In the former case the variable is also $A_{\overline{\pi}}B_{\overline{\pi}}$-clean, because $A_{\overline{\pi}} \subseteq A_{\overline{\sigma}}$ and $B_{\overline{\pi}} \subseteq B_{\overline{\sigma}}$. So the locality of the labeling function Lab gives us that the variable $\mathsf{Var}(l)$ is consistently labeled by a or b as needed. In the latter case where $\mathsf{Var}(l)$ is assigned by π, it follows that all the occurrences of the variable $\mathsf{Var}(l)$ are labeled b because the second alternative (D5.11.2) is not used. So even in this case the label b is assigned consistently as needed and the requirement D4.3.4 is satisfied. □

A similar property as for the strongest successor labeling and the extended-assignment labeling holds also for the restricted-assignment labeling.

Lemma 5.13. *Let Lab and Lab' be locality preserving labeling functions for a (A, B, π)-refutation R and a (A, B, σ)-refutation R, respectively. Let π be an extension of a partial variable assignment σ.*
If $\mathsf{Lab} \preceq \mathsf{Lab}'$ then also $\mathsf{Lab}^-_{\pi\to\sigma} \preceq \mathsf{Lab}'$.

Proof. We show the lemma by contradiction. For contradiction assume that $\mathsf{Lab}^-_{\pi\to\sigma} \not\preceq \mathsf{Lab}'$. Then there exists a vertex v and a literal l such that $\mathsf{Lab}^-_{\pi\to\sigma}(v,l) \not\preceq \mathsf{Lab}'(v,l)$. It especially means that $\mathsf{Lab}^-_{\pi\to\sigma}(v,l) \neq \mathsf{Lab}'(v,l)$.

Let the label be set by the first alternative (D5.11.1). Then it holds that $\mathsf{Lab}^+_{\pi\to\sigma}(v,l) = ab$. In such a case it must hold that $\mathsf{Lab}'(v,l) = b$ (since only b is stronger than ab). The assumption of the alternative gives us that $\pi \models l$, so from locality of Lab it follows that $\mathsf{Lab}(v,l) = d^+$. The parameters v and l show that $\mathsf{Lab} \not\preceq \mathsf{Lab}'$ – violation of the assumptions of the lemma.

Let the label be set by the second alternative (D5.11.2). Then it holds that $\mathsf{Lab}^+_{\pi\to\sigma}(v,l) = a$. In such a case it must hold that $\mathsf{Lab}'(v,l) \neq a$. The variable $\mathsf{Var}(l)$ is $A_{\overline{\sigma}}B_{\overline{\sigma}}$-clean so from the requirement D4.3.4 it follows that any locality preserving function for (A, B, σ) must consistently assign either the label a or label b to the variable $\mathsf{Var}(l)$. Since Lab' is locality preserving, the only option is that $\mathsf{Lab}'(v,l) = b$. However, the assumption of the second alternative (D5.11.2) gives us that there exists a vertex v' and a literal l' ($v' \in V, l' \in cl(v')$) such that $\mathsf{Var}(l) = \mathsf{Var}(l')$ and $\mathsf{Lab}(v',l') \in \{a, ab, d^+\}$. It either holds that $\mathsf{Lab}'(v',l') = b$ or $\mathsf{Lab}'(v',l') \neq b$. In the former case the parameters v' and l' are the witnesses that $\mathsf{Lab} \not\preceq \mathsf{Lab}'$. In the latter case the parameters v' and l' are the witnesses that Lab' is not a locality preserving labeling, since it violates the locality rule $D4.3.4$, because the $A_{\overline{\sigma}}B_{\overline{\sigma}}$-clean variable $\mathsf{Var}(l)$ is not labeled consistently by Lab'.

Let the label be set by the third alternative (D5.11.3) so $\mathsf{Lab}^{-}_{\pi\to\sigma}(v,l) = a$. In such a case it holds that $\mathsf{Lab}'(v,l) \neq a$ and Lab' is not locality preserving since it directly violates the rule D4.3.2.

Let the label be set by the fourth alternative (D5.11.4) so $\mathsf{Lab}^{-}_{\pi\to\sigma}(v,l) = b$. And since b is the strongest label (according $\preceq$), $\mathsf{Lab}'(v,l)$ cannot be stronger so it cannot violate the assumption. Note that it must even hold that $\mathsf{Lab}'(v,l) = b$ otherwise Lab' is not locality preserving since the rule D4.3.3 is violated.

Let the label be set by the last alternative (D5.11.5) so $\mathsf{Lab}^{-}_{\pi\to\sigma}(v,l) = \mathsf{Lab}(v,l)$. In such a case the vertex v and the literal l also show that $\mathsf{Lab} \npreceq \mathsf{Lab}'$ so the assumptions of the lemma are violated. □

The restricted-assignment labeling might be stronger than the original labeling function; this is caused only by the D5.11.4 alternative. Additional requirements must ensure that the alternative is used only if the original label is b.

Lemma 5.14. *Let* Lab *be a locality preserving labeling function for* (A,B,π)*-refutation* R*. Let* π *be an extension of a partial variable assignment* σ*. If the clause sets* $B_{\overline{\pi}}$ *and* $B_{\overline{\sigma}}$ *are equal and all newly* σ*-unassigned variables (assigned by* π *and not by* σ*) are not* $B_{\overline{\sigma}}$*-local then:*

$$\mathsf{Lab} \preceq \mathsf{Lab}^{-}_{\pi\to\sigma}$$

Similarly to Lemma 5.10, the requirements of Lemma 5.14 are not the most general ones.

Proof sketch. Let the variable $\mathsf{Var}(l)$ is $B_{\overline{\sigma}}$-local. We show that it is also $B_{\overline{\pi}}$-local and so the locality constraint D4.3.3 guarantees that $\mathsf{Var}(l)$ is labeled b by Lab.

If the variable $\mathsf{Var}(l)$ were assigned by π, it would violate the requirement that the newly unassigned variables are not $B_{\overline{\sigma}}$-local. Thus the variable $\mathsf{Var}(l)$ is unassigned by π. From the assumption $B_{\overline{\pi}} = B_{\overline{\sigma}}$ and the fact that π is an extension of σ it follows that $B_{\overline{\sigma}}$-local variable $\mathsf{Var}(l)$ is also $B_{\overline{\pi}}$-local.

Note that the first alternative D5.11.1 cannot increase the strength because of the fact that $\pi \models l$ implies that Lab assigns label d^{+}. □

Proof. We want to show that $\forall v \in V$ and $\forall l \in cl(v)$ it holds $\mathsf{Lab}(v,l) \preceq \mathsf{Lab}^{-}_{\pi\to\sigma}(v,l)$. The label $\mathsf{Lab}^{-}_{\pi\to\sigma}(v,l)$ must be defined by one of the alternatives (D5.11.1-5) from the definition of the restricted-assignment labeling.

Let the first alternative (D5.11.1) be used so $\mathsf{Lab}^{-}_{\pi\to\sigma}(v,l) = ab$. From the assumption $\pi \models l$ of the alternative and the locality requirement D4.3.1 it follows that locality preserving $\mathsf{Lab}(v,l) = d^{+}$. It holds that $\mathsf{Lab}(v,l) = d^{+} \preceq ab = \mathsf{Lab}^{-}_{\pi\to\sigma}(v,l)$ as it is needed to show the goal.

Let the second (D5.11.2) or the third (D5.11.3) alternative be used so $\mathsf{Lab}^{-}_{\pi\to\sigma}(v,l) = a$. The requirement holds trivially in this case, since the label a is the weakest one according to the strength ordering $\preceq$. Note that the label $\bot$ is not applicable because $l \in cl(v)$.

Let the fourth alternative (D5.11.4) be used so $\mathsf{Lab}^{-}_{\pi\to\sigma}(v,l) = b$. We show that $\mathsf{Lab}(v,l) = b$ so the goal is satisfied. The variable $\mathsf{Var}(l)$ can be either assigned or unassigned by π. If the variable $\mathsf{Var}(l)$ were assigned by π, it would violate the requirement of the lemma that the newly unassigned variables are not $B_{\overline{\sigma}}$-local. Let the variable $\mathsf{Var}(l)$ be unassigned by π. We know that the variable $\mathsf{Var}(l)$ is $B_{\overline{\sigma}}$-local, formally $\mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\sigma}})$ and $\mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\sigma}})$. From the fact that $B_{\overline{\pi}} = B_{\overline{\sigma}}$ it follows that $\mathsf{Var}(l) \in \mathsf{Var}(B_{\pi}) = \mathsf{Var}(B_{\sigma})$. From the fact that π is extension of σ it follows that $\mathsf{Var}(A_{\overline{\pi}}) \subseteq \mathsf{Var}(A_{\overline{\sigma}})$, so $\mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{\pi}}) \subseteq \mathsf{Var}(A_{\overline{\sigma}})$ and the variable $\mathsf{Var}(l)$ is also $B_{\overline{\pi}}$-local. Thus the locality preserving function Lab assigns the label b to the variable as it is required.

Let the last alternative (D5.11.5) be used then the requirement holds trivially. (It holds that $\mathsf{Lab}^{-}_{\pi\to\sigma}(v,l) = \mathsf{Lab}(v,l)$ and for any label c it holds that $c \preceq c$.) □

Now, all the labeling functions are connected together to get from $(A, S\cup B, \pi)$ to $(A\cup S, B, \pi')$. The following lemma states the prerequisites and properties of the labeling function chain.

Lemma 5.15 (Chaining)**.** *Let* Lab *and* Lab' *be locality preserving labeling functions for an* $(A, S\cup B, \pi)$*-refutation* R *and an* $(A\cup S, B, \pi')$*-refutation* R*, respectively. Let assume a partial variable assignment* (π,π') *exists and let* $B_{\overline{\pi'}} \subseteq B_{\overline{\pi}}$*. Let* $A_{\overline{\pi}} \subseteq A_{\overline{\pi'}}$ *and all the variables assigned by* π' *but not by* π *be assignable in* Lab*. Let the variables assigned by* π' *and not by* π *be not* $B_{\overline{\pi'}}$*-local.*

Let $\mathsf{Lab}^{+}_{\pi\to(\pi,\pi')}$ *be an extended-assignment labeling for* Lab*. Let* $\mathsf{Lab}^{S}_{(\pi,\pi')}$ *be the strongest successor labeling for* $\mathsf{Lab}^{+}_{\pi\to(\pi,\pi')}$ *induced by* S*. Let* $\mathsf{Lab}^{-}_{(\pi,\pi')\to\pi'}$ *be a restricted-assignment labeling for* $\mathsf{Lab}^{S}_{(\pi,\pi')}$*.*

If $\mathsf{Lab} \preceq \mathsf{Lab}'$ *then the following holds:*

$$\mathsf{Lab} \preceq \mathsf{Lab}^{+}_{\pi\to(\pi,\pi')} \preceq \mathsf{Lab}^{S}_{(\pi,\pi')} \preceq \mathsf{Lab}^{-}_{(\pi,\pi')\to\pi'} \preceq \mathsf{Lab}'$$

The most important result of the lemma is that the chain of the labeling functions is always in between Lab and Lab' according to the strength ordering. This allows to apply the theorems about the interpolants strength.

Proof sketch. First, we show that the defined labeling functions form a strength-decreasing chain $\mathsf{Lab} \preceq \mathsf{Lab}^{+}_{\pi\to(\pi,\pi')} \preceq \mathsf{Lab}^{S}_{(\pi,\pi')} \preceq \mathsf{Lab}^{-}_{(\pi,\pi')\to\pi'}$ using Lemmas 5.10, 5.14. Later, the chain is extended by Lab'. For contradiction, let us assume $\mathsf{Lab}^{-}_{(\pi,\pi')\to\pi'} \not\preceq \mathsf{Lab}'$; then going backwards along the chain, we show that either Lab' is not locality preserving or $\mathsf{Lab} \not\preceq \mathsf{Lab}'$. The proof exploits the fact that all the alternatives in extended/restricted-assignment and the strongest labeling are introduced to satisfy the locality constraints and assigns the strongest label possible. □

Proof. First, the assumptions of Lemma 5.10 and 5.14 are shown.

Assumptions of Lemma 5.10: From the fact that (π,π') is an extension of π, it follows that $A_{\overline{(\pi,\pi')}} \subseteq A_{\overline{\pi}}$. Formally a clause $c \in A_{\overline{(\pi,\pi')}}$ is satisfied by neither π nor π'; thus it holds that $c \in A_{\overline{\pi}}$. Now we show that $A_{\overline{\pi}} \subseteq A_{\overline{(\pi,\pi')}}$. Let $c \in A_{\overline{\pi}}$ (i.e. c is unsatisfied by π). Then because of the assumption $A_{\overline{\pi}} \subseteq A_{\overline{\pi'}}$ of Lemma 5.15 it holds that $c \in A_{\overline{\pi'}}$ (i.e. c is unsatisfied by π'). So we have shown that c is unsatisfied by both π and π' thus $c \in A_{\overline{(\pi,\pi')}}$. We have proved that $A_{\overline{(\pi,\pi')}} = A_{\overline{\pi}}$.

The second assumption about the assignability of the newly assigned variables follows from the fact that the newly (π,π')-assigned variables (i.e. variables in $\mathsf{Var}((\pi,\pi')) \setminus \mathsf{Var}(\pi)$) are exactly the newly π'-assigned variables (variables in $\mathsf{Var}(\pi') \setminus \mathsf{Var}(\pi)$). Formally it holds:

$$\mathsf{Var}((\pi,\pi')) = \mathsf{Var}(\pi) \cup \mathsf{Var}(\pi')$$

$$\begin{aligned} &\mathsf{Var}((\pi,\pi')) \setminus \mathsf{Var}(\pi) = \\ &= (\mathsf{Var}(\pi) \setminus \mathsf{Var}(\pi)) \cup (\mathsf{Var}(\pi') \setminus \mathsf{Var}(\pi)) = \\ &= \emptyset \cup (\mathsf{Var}(\pi') \setminus \mathsf{Var}(\pi)) \end{aligned}$$

Assumptions of Lemma 5.14: The proof is symmetric to the previous paragraph. From the fact that (π,π') is an extension of π', it follows that $B_{\overline{(\pi,\pi')}} \subseteq B_{\overline{\pi'}}$. (Formally a clause $c \in B_{\overline{(\pi,\pi')}}$ is satisfied by neither π nor π'. So the clause c is not satisfied by π' thus it holds $c \in B_{\overline{\pi'}}$.) Now we show that $B_{\overline{\pi'}} \subseteq B_{\overline{(\pi,\pi')}}$. Assume a clause $c \in B_{\overline{\pi'}}$ (i.e. c is unsatisfied by π') then because of the assumption $B_{\overline{\pi'}} \subseteq B_{\overline{\pi}}$ of Lemma 5.15, it holds that $c \in B_{\overline{\pi}}$ (i.e. c is unsatisfied by π). So we have shown that c is unsatisfied by both π and π' thus $c \in B_{\overline{(\pi,\pi')}}$. We have proved that $B_{\overline{(\pi,\pi')}} = B_{\overline{\pi'}}$ The second assumption about the unassigned variables follows from the fact that the newly (π,π')-unassigned variables (i.e. variables in $\mathsf{Var}((\pi,\pi')) \setminus \mathsf{Var}(\pi')$) are exactly the newly π'-unassigned variables (variables in $\mathsf{Var}(\pi) \setminus \mathsf{Var}(\pi')$). Formally it holds:

$$\mathsf{Var}((\pi,\pi')) = \mathsf{Var}(\pi) \cup \mathsf{Var}(\pi')$$

$$\begin{aligned} \mathsf{Var}((\pi,\pi')) \setminus \mathsf{Var}(\pi') =& (\mathsf{Var}(\pi) \setminus \mathsf{Var}(\pi')) \cup (\mathsf{Var}(\pi') \setminus \mathsf{Var}(\pi')) = \\ =& (\mathsf{Var}(\pi) \setminus \mathsf{Var}(\pi')) \cup \emptyset \end{aligned}$$

So Lemma 5.10 and 5.14 can be used to create a strength-decreasing chain of labeling functions:

$$\mathsf{Lab} \preceq \mathsf{Lab}^{+}_{\pi\to(\pi,\pi')} \preceq \mathsf{Lab}^{S}_{(\pi,\pi')} \preceq \mathsf{Lab}^{-}_{(\pi,\pi')\to\pi'}$$

Now in the second part of the proof, we extend the chain with Lab'. The proof is done by contradiction – we assume that $\mathsf{Lab}^{-}_{(\pi,\pi')\to\pi'} \not\preceq \mathsf{Lab}'$. In the following we push the contradiction backwards along the chain. In each backward step, all the alternatives from the definition of the labeling function are considered. The following cases (contradictions of the assumptions of Lemma 5.15 may occur:

(L5.15.1) shows that Lab' is not weaker than Lab, or

(L5.15.2) shows that Lab' is not locality preserving, or

(L5.15.3) shows that $B_{\overline{\pi}} \not\subseteq B_{\overline{\pi'}}$, or

(L5.15.4) pushes the contradiction back (using the last "*otherwise*" alternative)

Also note that the only assumption used in this part of the proof is that $B_{\overline{\pi}} \subseteq B_{\overline{\pi'}}$.

Initial step. It either holds that $\mathsf{Lab}^-_{(\pi,\pi')\to\pi'} \preceq \mathsf{Lab}'$ and we are done or the opposite holds. Assume it holds that $\mathsf{Lab}^-_{(\pi,\pi')\to\pi'} \not\preceq \mathsf{Lab}'$. It means that there exists a vertex $v \in V$ and a literal $l \in cl(v)$ such that $\mathsf{Lab}^-_{(\pi,\pi')\to\pi'}(v,l) \not\preceq \mathsf{Lab}'(v,l)$. It especially means that $\mathsf{Lab}^-_{(\pi,\pi')\to\pi'}(v,l) \neq \mathsf{Lab}'(v,l)$.

The label $\mathsf{Lab}^-_{(\pi,\pi')\to\pi'}(v,l)$ must be defined by one of the D5.11.1-5 alternatives.

(L5.15-D5.11.1) Let the first alternative (D5.11.1) is used to define the label so it holds that $\mathsf{Lab}^-_{(\pi,\pi')\to\pi'}(v,l) = ab$. It must hold that $\mathsf{Lab}'(v,l) = b$, since only the label b is strictly stronger than the label ab. The alternative assumptions ($\pi, \pi' \models l$ and $\mathsf{Var}(l)$ is unassigned by π') imply that $\pi \models l$ thus for the locality preserving Lab it must hold that $\mathsf{Lab}(v,l) = d^+$. So we have shown that $\mathsf{Lab} \not\preceq \mathsf{Lab}'$ (contradiction type L5.15.1).

(L5.15-D5.11.2) Let the second alternative (D5.11.2) is used to define the label so it holds that $\mathsf{Lab}^-_{(\pi,\pi')\to\pi'}(v,l) = a$. It holds that $\mathsf{Lab}'(v,l) \neq a$. Since the variable is AB-clean in $(A \cup S, B, \pi')$, it must hold that $\mathsf{Lab}'(v,l) = b$, otherwise Lab' is not locality preserving (contradiction type L5.15.2). The assumptions of the alternative give us that there exist v' and l' such that $\mathsf{Lab}^S_{(\pi,\pi')}(v',l') \in \{a, ab, d^+\}$. Moreover it also holds that $\mathsf{Lab}'(v',l') = b$ (since it consistently assigns the same label b to $\mathsf{Var}(l) = \mathsf{Var}(l')$). The assumptions of the alternative ($\mathsf{Var}(l) = \mathsf{Var}(l')$ is assigned by (π,π') and unassigned by π') also imply that $\mathsf{Var}(l')$ is assigned by π.

Now we focus on the $\mathsf{Lab}(v',l')$. It either holds that $\pi \models l'$ or $\pi \models \neg l'$. In the former case it must hold that $\mathsf{Lab}(v',l') = d^+$ (since Lab is locality preserving) so we have shown that $\mathsf{Lab} \not\preceq \mathsf{Lab}'$ (a contradiction type L5.15.1), since:

$$\mathsf{Lab}(v',l') = d^+ \not\preceq b = \mathsf{Lab}'(v,l)$$

In the latter case it holds that $\mathsf{Lab}(v',l') = \mathsf{Lab}^+_{\pi\to(\pi,\pi')}(v',l') = \mathsf{Lab}^S_{(\pi,\pi')}(v',l')$. Since $\mathsf{Var}(l')$ is assigned by π, the alternatives D5.7.2, D5.7.3, D5.7.4 and D5.1.1 cannot apply; also the alternative D5.7.1 cannot apply (if $\pi \models \neg l$ then because (π,π') is PVA it cannot hold $\pi,\pi' \models l'$). Thus the only remaining alternatives are D5.7.5 and D5.1.2, which guarantee the equality above. And again as in the former case, we have shown contradiction (type L5.15.1) since:

$$\mathsf{Lab}(v',l') = \mathsf{Lab}^+_{\pi\to(\pi,\pi')}(v',l') = \mathsf{Lab}^S_{(\pi,\pi')}(v',l') \in \{a, ab, d^+\} \not\preceq \mathsf{Lab}'(v',l') = b$$

(L5.15-D5.11.3) Let the third alternative (D5.11.3) is used to define the label so it holds that $\mathsf{Lab}^-_{(\pi,\pi')\to\pi'}(v,l) = a$. It must hold that $\mathsf{Lab}'(v,l) \neq a$. However the variable $\mathsf{Var}(l)$ is A-local in $(A \cup S, B, \pi')$ thus we have shown a contradicting fact that Lab' is not locality preserving (L5.15.2).

(L5.15-D5.11.4) Let the fourth alternative (D5.11.4) is used to define the label so it holds that $\mathsf{Lab}^-_{(\pi,\pi')\to\pi'}(v,l) = b$. It must hold that $\mathsf{Lab}'(v,l) \neq b$. However the variable $\mathsf{Var}(l)$ is B-local in $(A \cup S, B, \pi')$ thus we have shown contradicting the fact that Lab' is not locality preserving (L5.15.2).

(L5.15-D5.11.5) Let the last alternative (D5.11.5) is used to define the label so it holds that $\mathsf{Lab}^-_{(\pi,\pi')\to\pi'}(v,l) = \mathsf{Lab}^S_{(\pi,\pi')}(v,l)$. In this case we push the contradiction backwards (L5.15.4) and we show that $\mathsf{Lab}^S_{(\pi,\pi')}(v,l) \not\preceq \mathsf{Lab}'(v,l)$.

Second step. To sum up the situation, the last alternative (D5.11.5) in the definition of the restricted-assignment labeling is used which yields:

$$\mathsf{Lab}^S_{(\pi,\pi')}(v,l) = \mathsf{Lab}^-_{(\pi,\pi')\to\pi'}(v,l) \not\preceq \mathsf{Lab}'(v,l)$$

The label $\mathsf{Lab}^S_{(\pi,\pi')}(v,l)$ must be defined by one of the D5.1.1-2 alternatives.

(L5.15-D5.1.1) Let the first alternative (D5.1.1) is used to define the label so the following holds: $\mathsf{Lab}^S_{(\pi,\pi')}(v,l) = \mathsf{Lab}^-_{(\pi,\pi')\to\pi'}(v,l) = a$. The assumptions of the alternative give us that the variable $\mathsf{Var}(l)$ is A-local in $(A \cup S, B, (\pi,\pi'))$, formally:

$$\mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{(\pi,\pi')}} \cup S_{\overline{(\pi,\pi')}}) \qquad \mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{(\pi,\pi')}})$$

Since the alternative D5.11.5 in the restricted-assignment labeling is used (and not D5.11.3) the $\mathsf{Var}(l)$ is not A-local in $(A \cup S, B, \pi')$, formally it cannot hold:

$$\mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\pi'}} \cup S_{\overline{\pi'}}) \qquad \mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\pi'}})$$

Bellow, we show that if it were $B_{\overline{\pi}} \subseteq B_{\overline{\pi'}}$ then the variable $\mathsf{Var}(l)$ would be A-local in $(A \cup S, B, \pi')$ (so the alternative D5.11.5 would not be used). Thus, to use the D5.11.5 alternative in the restricted-assignment labeling, it must hold $B_{\overline{\pi}} \not\subseteq B_{\overline{\pi'}}$ (L5.15.3) which is a contradiction of the assumptions of the chaining lemma.

(Due to removal of variables) it holds that $\mathsf{Var}(A_{\overline{(\pi,\pi')}}) \subseteq \mathsf{Var}(A_{\overline{\pi'}})$, for any set of clauses A. Thus it holds:

$$\mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{(\pi,\pi')}} \cup S_{\overline{(\pi,\pi')}}) \Rightarrow \mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\pi'}} \cup S_{\overline{\pi'}})$$

As we have shown above (assumptions of Lemma 5.14), if it holds that $B_{\overline{\pi}} \subseteq B_{\overline{\pi'}}$ then $B_{\overline{(\pi,\pi')}} = B_{\overline{\pi'}}$ so $\mathsf{Var}(B_{\overline{(\pi,\pi')}}) = \mathsf{Var}(B_{\overline{\pi'}})$. This gives us:

$$\mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{(\pi,\pi')}}) \Rightarrow \mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{\pi'}}) \tag{3}$$

We have shown (the contradicting fact) that the variable $\mathsf{Var}(l)$ is (even) A-local in $(A \cup S, B, \pi')$. To be able to use the alternative D5.11.5 in the following restricted-assignment labeling, it must hold that $B_{\overline{\pi}} \not\subseteq B_{\overline{\pi'}}$ (L5.15.3) which is a contradiction of the assumptions of the chaining lemma.

(L5.15-D5.1.2) Let the second alternative (D5.1.2) is used to define the label so the following holds: $\mathsf{Lab}^{S}_{(\pi,\pi')}(v,l) = \mathsf{Lab}^{+}_{\pi\to(\pi,\pi')}(v,l)$. In this case we push the contradiction backwards (L5.15.4) and we show that $\mathsf{Lab}^{+}_{\pi\to(\pi,\pi')}(v,l) \not\preceq \mathsf{Lab}'(v,l)$.

Third step. To sum up the situation, the last alternative in the definition of the strongest-successor and restricted-assignment labeling is used:

$$\mathsf{Lab}^{+}_{\pi\to(\pi,\pi')}(v,l) = \mathsf{Lab}^{S}_{(\pi,\pi')}(v,l) = \mathsf{Lab}^{-}_{(\pi,\pi')\to\pi'}(v,l) \not\preceq \mathsf{Lab}'(v,l)$$

The label $\mathsf{Lab}^{+}_{\pi\to(\pi,\pi')}(v,l)$ must be defined by one of the D5.7.1-5 alternatives.

(L5.15-D5.7.1) Let the first alternative (D5.7.1) is used to define the label so it holds that $\mathsf{Lab}^{+}_{\pi\to(\pi,\pi')}(v,l) = d^{+}$. It must hold that $\mathsf{Lab}'(v,l) = b$, since only the label b is strictly stronger than the label d^{+}. The assumptions of the alternative ($\pi \not\models l$ and $\pi,\pi' \models l$) imply that $\pi' \models l$. Thus we have shown the contradicting fact that Lab' is not locality preserving (L5.15.2), since the locality constrain D4.3.1 requires that $\mathsf{Lab}'(v,l) = d^{+}$.

(L5.15-D5.7.2) Let the second alternative (D5.7.2) is used to define the label so it holds that $\mathsf{Lab}^{+}_{\pi\to(\pi,\pi')}(v,l) = a$ and the variable $\mathsf{Var}(l)$ is A-local in $(A \cup S, B, (\pi,\pi'))$. Here, exactly the same reasoning about A-local variables as in the L5.15-D5.1.1 case is applied.

If it were $B_{\overline{\pi}} \subseteq B_{\overline{\pi'}}$ then the variable $\mathsf{Var}(l)$ would be A-local in $(A \cup S, B, \pi')$ (so the alternative D5.11.5 would not be used). Thus, to use the D5.11.5 alternative in the following restricted-assignment labeling, it must hold that $B_{\overline{\pi}} \not\subseteq B_{\overline{\pi'}}$ (L5.15.3) which is a contradiction of the assumptions of the chaining lemma.

(L5.15-D5.7.3) Let the third alternative (D5.7.3) is used to define the label so it holds that $\mathsf{Lab}^{+}_{\pi\to(\pi,\pi')}(v,l) = b$. The strongest possible label is assigned so the labeling function $\mathsf{Lab}'(v,l)$ cannot have any (strictly) stronger label.

(L5.15-D5.7.4) Let the fourth alternative (D5.7.4) is used to define the label so it holds that $\mathsf{Lab}^{+}_{\pi\to(\pi,\pi')}(v,l) = a$, the variable $\mathsf{Var}(l)$ is unassigned by π and π', the variable is AB-clean in $(A \cup S, B, (\pi,\pi'))$. Formally, it holds:

$$\mathsf{Var}(l) \notin \mathsf{Var}(A_{\overline{(\pi,\pi')}} \cup S_{\overline{(\pi,\pi')}}) \qquad \mathsf{Var}(l) \notin \mathsf{Var}(B_{\overline{(\pi,\pi')}})$$

The variable $\mathsf{Var}(l)$ unassigned by π' can be either: A-local, B-local, AB-shared or AB-clean in $(A \cup S, B, \pi')$. Since the alternative D5.11.5 in the restricted-assignment labeling is used, the variable cannot be A-local, B-local (otherwise the alternatives D5.11.3 or D5.11.4 would have been used). If the variable $\mathsf{Var}(l)$ were AB-clean even in $(A \cup S, B, \pi')$, then the alternative D5.11.2 would have been used. (In the strongest-successor labeling the second alternative is used and the label a is preserved and the v, l satisfy the assumptions of the alternative D5.11.2). The only remaining case is that the variable $\mathsf{Var}(l)$ is AB-shared in $(A \cup S, B, \pi')$, formally it must hold:

$$\mathsf{Var}(l) \in \mathsf{Var}(A_{\overline{\pi}'} \cup S_{\overline{\pi}'}) \qquad\qquad \mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\pi}'})$$

We show that if it holds $B_{\overline{\pi}} \subseteq B_{\overline{\pi}'}$ then the variable $\mathsf{Var}(l)$ is AB-shared in $(A \cup S, B, \pi')$ (so the alternative D5.11.5 is not used). Thus, to use the D5.11.5 alternative in the restricted-assignment labeling, it must hold that $B_{\overline{\pi}} \not\subseteq B_{\overline{\pi}'}$ (L5.15.3) which is a contradiction of the assumptions of the chaining lemma. Assume it holds that $B_{\overline{\pi}} \subseteq B_{\overline{\pi}'}$. Then (as it is shown above in the assumptions of Lemma 5.14) $B_{\overline{(\pi,\pi')}} = B_{\overline{\pi}'}$ and $\mathsf{Var}(B_{\overline{(\pi,\pi')}}) = \mathsf{Var}(B_{\overline{\pi}'})$. So as we have shown in the equation 3, it cannot be $\mathsf{Var}(l) \in \mathsf{Var}(B_{\overline{\pi}'})$ so the variable cannot be AB-shared.

(L5.15-D5.7.5) Let the last alternative (D5.7.5) is used to define the label so it holds that $\mathsf{Lab}^{+}_{\pi\to(\pi,\pi')}(v,l) = \mathsf{Lab}(v,l)$. In this case we push the contradiction backwards enough to show that $\mathsf{Lab} \not\preceq \mathsf{Lab}'$ (L5.15.1) which violates the assumptions of the chaining lemma. It holds:

$$\mathsf{Lab}(v,l) = \mathsf{Lab}^{+}_{\pi\to(\pi,\pi')}(v,l) = \mathsf{Lab}^{S}_{(\pi,\pi')}(v,l) = \mathsf{Lab}^{-}_{(\pi,\pi')\to\pi'}(v,l) \not\preceq \mathsf{Lab}'(v,l)$$

So we have analysed all the cases and have shown that if the assumptions L5.15.1-3 hold, then $\mathsf{Lab}^{-}_{(\pi,\pi')\to\pi'} \preceq \mathsf{Lab}'$ and the lemma is proved. □

The following theorem states the main result for the general inductive step, where different PVAs are used.

Theorem 5.16 (Inductive step). *Let* Lab *be a locality preserving labeling function for* $(A, S \cup B, \pi)$*-refutation* R *and let the partial variable assignment interpolant* $I = \mathsf{Lpaltp}(\mathsf{Lab}, (A, S \cup B, \pi))$*. Let* Lab' *be a locality preserving labeling function for* $(A \cup S, B, \pi')$*-refutation* R *and let the PVAI interpolant* $I' = \mathsf{Lpaltp}(\mathsf{Lab}', (A \cup S, B, \pi'))$*. Let* $A_{\overline{\pi}} \subseteq A_{\overline{\pi}'}$*,* $B_{\overline{\pi}'} \subseteq B_{\overline{\pi}}$ *and let the variables assigned by* π' *and not by* π *are assignable in* Lab *and the variables assigned by* π *and not by* π' *are not* $B_{\overline{\pi}'}$*-local.*

If $\mathsf{Lab} \preceq \mathsf{Lab}'$ *then it holds* $\pi, \pi' \models I \wedge S \Rightarrow I'$*.*

In terms of our motivation example, the theorem relates interpolants for subsequent nodes in the abstract reachability graph. The theorem provides us with the main result required for well-labeledness [2, 8]. Assume we have interpolants I_2 for the node 2 and I_3 for the node 3. This means $I_2 \wedge \mu_2 \Rightarrow I_3$. Having this for all nodes, a path interpolant for any path in ARG can be formed. Note that all the assumptions of the theorem except for the assignability and relation if labeling functions are automatically fulfilled by the used Cond encoding.

Proof. The expression (π, π') either represents a valid assignment or not (in the case of a conflict). In the latter case the assignments π and π' are conflicting, i.e., there exists a literal l being assigned $\top$ by π and $\bot$ by π', or vice versa. Then, (π, π') contains a contradiction, thus $\pi, \pi' \models I \wedge S \Rightarrow I'$ holds trivially.

For the former case, assume that (π, π') is a valid assignment. Using Lemma 5.15 we obtain:

$$\mathsf{Lab} \preceq \mathsf{Lab}^{+}_{\pi\to(\pi,\pi')} \preceq \mathsf{Lab}^{S}_{(\pi,\pi')} \preceq \mathsf{Lab}^{-}_{(\pi,\pi')\to\pi'} \preceq \mathsf{Lab}'$$

Moreover $\mathsf{Lab}^{+}_{\pi\to(\pi,\pi')}$ is locality preserving for $(A, S \cup B, (\pi, \pi'))$ (lemma 5.8), $\mathsf{Lab}^{S}_{(\pi,\pi')}$ is locality preserving for $(A \cup S, B, (\pi, \pi'))$ (lemma 5.2) and $\mathsf{Lab}^{-}_{(\pi,\pi')\to\pi'}$ is locality preserving for $(A \cup S, B, \pi')$ (lemma 5.12).

First, Theorem 4.12 is applied using the labeling function $\mathsf{Lab}^{+}_{\pi\to(\pi,\pi')}$ to obtain the interpolant I_1 for $(A, S \cup B, (\pi, \pi'))$ such that $\pi, \pi' \models I \Rightarrow I_1$. Then, Theorem 5.5 is applied using the strongest successor labeling $\mathsf{Lab}^{S}_{(\pi,\pi')}$ to get the interpolant I_2 for $(A \cup S, B, (\pi, \pi'))$ such that $\pi, \pi' \models I_1 \wedge S \Rightarrow I_2$. Then, Theorem 4.12 is applied again, using $\mathsf{Lab}^{-}_{(\pi,\pi')\to\pi'}$ to obtain the interpolant I_3 for $(A \cup S, B, \pi')$ such that $\pi, \pi' \models I_2 \Rightarrow I_3$. Finally, Theorem 4.12 with a fixed assignment π' and Lab' is applied to weaken the interpolant I_3 to form I' for $(A \cup S, B, \pi')$, so it holds that $\pi, \pi' \models I_3 \Rightarrow I'$.

To sum it up, we have proved that:

$$\pi, \pi' \models I \wedge S \overset{T4.12}{\Rightarrow} I_1 \wedge S \overset{T5.5}{\Rightarrow} I_2 \overset{T4.12}{\Rightarrow} I_3 \overset{T4.12}{\Rightarrow} I'$$

□

6 Conclusion

In this paper, we introduced Partial Variable Assignment Interpolants, which (in contrast to Craig Interpolants) permit to specialize the interpolants to sub-problems specified in the form of variable assignments. We showed how the concept of PVAIs finds application in abstract reachability graphs and DAG interpolation. We also developed the new framework of Labeled Partial Assignment Interpolation Systems, which can be used to compute PVAI's for propositional logic, and showed its properties.

To the best of our knowledge, the only strongly related work in this area is [2], based on the linearization of a DAG into a single path. The authors implemented the proposed technique inside the UFO tool for linear integer arithmetic; the computed interpolants are quantified formulas in general.

PVAIs do not yield DAG interpolants directly, but only simple post-processing steps are required. In the future, we will show how quantifier-free DAG interpolants can be derived from PVAIs, in contrast to the method presented in [2]. Further, we plan to extend the ideas of LPAIS and to introduce a PVAI interpolation system for linear integer arithmetic, a theory particularly relevant to program verification.

Acknowledgment. Special thanks go to Ondřej Šerý for his valuable contribution.

References

[1] Aws Albarghouthi, Arie Gurfinkel, and Marsha Chechik. Craig Interpretation. In *SAS '12*, volume 7460 of *LNCS*, pages 300–316, 2012.

[2] Aws Albarghouthi, Arie Gurfinkel, and Marsha Chechik. From Under-Approximations to Over-Approximations and Back. In *TACAS '12*, volume 7214 of *LNCS*, pages 157–172, 2012.

[3] Aws Albarghouthi, Yi Li, Arie Gurfinkel, and Marsha Chechik. Ufo: A Framework for Abstraction- and Interpolation-Based Software Verification. In *CAV '12*, volume 7358 of *LNCS*, pages 672–678, 2012.

[4] W. Craig. Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory. *J. of Symbolic Logic*, pages 269–285, 1957.

[5] Vijay D'Silva, Daniel Kroening, Mitra Purandare, and Georg Weissenbacher. Interpolant strength. In *VMCAI'10*, volume 5944 of *LNCS*, pages 129–145, 2010.

[6] Arie Gurfinkel, Simone Fulvio Rollini, and Natasha Sharygina. Interpolation Properties and SAT-Based Model Checking. In *ATVA '13*, volume 8172 of *LNCS*, pages 255–271, 2013.

[7] Kenneth L. McMillan. Interpolation and SAT-Based Model Checking. In *CAV '03*, volume 2725 of *LNCS*, pages 1–13, 2003.

[8] Kenneth L. McMillan. Lazy Abstraction with Interpolants. In *CAV '06*, volume 4144 of *LNCS*, pages 123–136, 2006.

[9] Pavel Pudlák. Lower Bounds for Resolution and Cutting Plane Proofs and Monotone Computations. *Journal of Symbolic Logic*, 62(3):981–998, 1997.

[10] S.F. Rollini, L. Alt, G. Fedyukovich, A. Hyv ärinen, and N. Sharygina. PeRIPLO: A Framework for Producing Effective Interpolants in SAT-Based Software Verification. In *LPAR*, 2013.

[11] Simone Fulvio Rollini, Ondrej Sery, and Natasha Sharygina. Leveraging Interpolant Strength in Model Checking. In *CAV '12*, volume 7358 of *LNCS*, pages 193–209, 2012.

[12] Yakir Vizel and Orna Grumberg. Interpolation-sequence based Model Checking. In *FMCAD '09*, pages 1–8. IEEE, 2009.