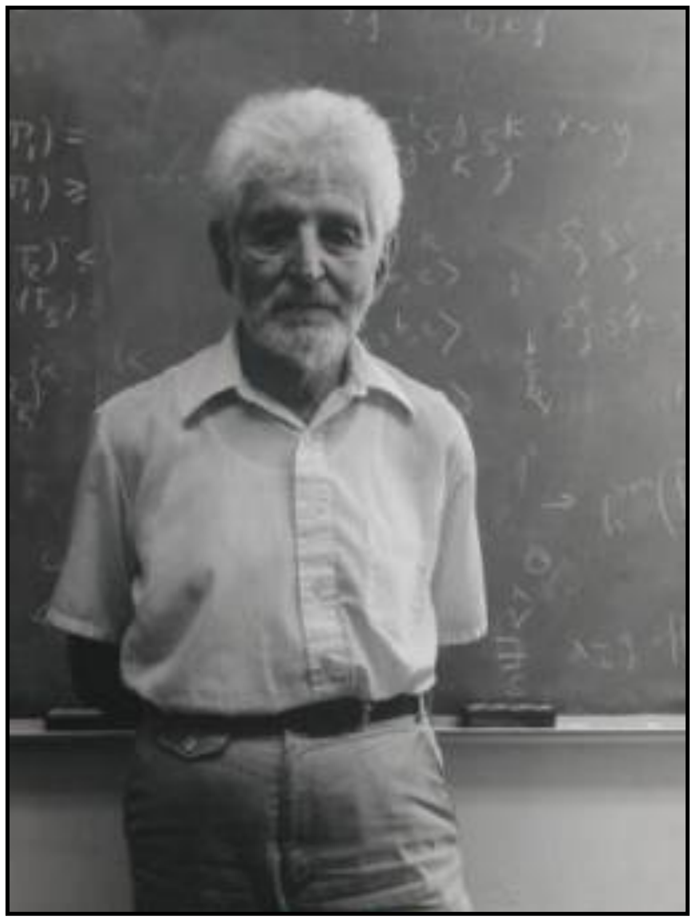
Interpolation: Theory and Applications

Vijay D'Silva
Google Inc., San Francisco

Logic Colloquium, U.C. Berkeley
2016

Interpolation Lemma (1957)



William Craig in 1988

http://sophos.berkeley.edu/interpolations/

THE JOURNAL OF SYMBOLIC LOGIC
VOLUME 22, Number 3, Sept. 1957

LINEAR REASONING.
A NEW FORM OF THE HERBRAND-GENTZEN THEOREM.

WILLIAM CRAIG

1. Introduction. In Herbrand's Theorem [2] or Gentzen's Extended Hauptsatz [1], a certain relationship is asserted to hold between the structures of A and A′, whenever A *implies* A′ (i.e., A $\supset$ A′ is valid) and moreover A is a conjunction and A′ an alternation of first-order formulas in prenex normal form. Unfortunately, the relationship is described in a roundabout way, by relating A and A′ to a quantifier-free tautology. One purpose

THE JOURNAL OF SYMBOLIC LOGIC
Volume 22, Number 3, Sept, 1957

THREE USES OF THE HERBRAND-GENTZEN THEOREM IN RELATING MODEL THEORY AND PROOF THEORY

WILLIAM CRAIG

1. Introduction. One task of metamathematics is to relate suggestive but nonelementary modeltheoretic concepts to more elementary prooftheoretic concepts, thereby opening up modeltheoretic problems to prooftheoretic methods of attack. Herbrand's Theorem (see [8] or also [9],

Interpolants

An interpolant I for a pair of formulae A and B, where the validity of A implies the validity of B, is a formula satisfying that: (i) A implies I, (ii) I implies B, and (iii) the *vocabulary condition* that the non-logical symbols in I occur in both A and B.

A logic has the interpolation property if every such A and B has an interpolant.

$$P \vee (Q \wedge R) \qquad\qquad P \vee Q \qquad\qquad S \implies (\neg Q \implies P)$$

Theorem. (Craig, 1957) First-order logic has the interpolation property.

“In terms of reasoning, this is not at all surprising. If A involves apples and oranges, and B involves apples and bananas and A implies B, then A ought to imply a statement that involves only apples and B ought to follow from a statement that involves only apples. The oranges should not help and the bananas should not hurt.

So what is the mystery then? The Craig statement is trickier to prove than one might think. One has to have the same statement about apples for A and B! ”

-- Alessandra Carbone, Bulletin of the AMS, April ’97

IBM

International Business Machines Corporation
2050 Rt 52
Hopewell Junction, NY 12533
845-892-5262

October 7, 2008

Dear Andreas,

I would like to congratulate Cadence Research Labs on their 15th Anniversary. In these 15 years, Cadence Research Labs has worked at several frontiers of Electronic Design Automation. They focus on hard problems that when solved significantly push the state of the art forward. They found novel solutions to system, synthesis and formal verification problems.

Formal verification is the process of exhaustively validating that a logic entity behaves correctly. In contrast to testing-based approaches, which may expose flaws though generally cannot yield a proof of correctness, the exhaustiveness of formal verification ensures that no flaw will be left unexposed. Formal verification is thus a critical technology in many domains, being essential to safety-critical applications and to enable increased quality and reduced development costs of hardware and software systems. The benefits of formal verification come at a substantial "cost": its exhaustiveness implies that it generally requires computational resources which grow exponentially with respect to the size of the entity being analyzed. Cadence Research Labs has had a fundamental role in the research and development of leading-edge formal verification technologies, which have been critical to increasing the scalability and applicability of formal verification techniques to an industrially relevant level.

CRL made important contributions in satisfiability checking technologies and model checking algorithms. Satisfiability checking is arguably one of the most fundamental algorithms in computer-aided design, with pervasive application domains including verification. Members of Cadence Research labs are world-recognized experts in the field of high-performance satisfiability solvers, and collectively have developed a set of solvers including MiniSAT, BerkMin, and Forklift which have won numerous competitions, been downloaded and used in thousands of applications, and have integrated novel tricks and ideas which have become the basis of countless other solvers.

Model checking algorithms are widely used for verifying hardware and software models. CRL has pioneered numerous fundamental ideas and algorithms to this field, including "interpolation" as a satisfiability-based proof method which is often dramatically faster and more scalable than prior proof techniques. CBL researchers invented numerous novel methods to automatically reduce the domain of a verification problem through "abstracting" it based upon unsatisfiability proofs. These techniques have substantially increased the scalability of formal verification of complex hardware designs.

CRL researchers have not only used logic optimizations to speed up formal verification algorithms, but are now also applying them to sequential optimization. Sequential synthesis has long been a holy grail in logic optimization. A large part of the design space remains untapped unless one can reliably and effectively optimize and verify in the sequential domain. Recent progress from CRL shows that there is some promise we can tap into this some time in the not too distant future.

Leon

Leon Stok
Director,
Electronic Design Automation
IBM Corporation

International Business Machines Corporation
2050 Rt 52
Hopewell Junction, NY 12533
845-892-5262

October 7, 2008

Dear Andreas,

I would like to congratulate Cadence Research Labs on their 15th Anniversary. In these 15 years, Cadence Research Labs has worked at several frontiers of Electronic Design Automation. They focus on hard problems that when solved significantly push the state of the art forward. They found novel solutions to system, synthesis and formal verification problems.

Formal verification is the process of exhaustively validating that a logic entity behaves correctly. In contrast to testing-based approaches, which may expose flaws though generally cannot yield a proof of correctness, the exhaustiveness of formal verification ensures that no flaw will be left unexposed. Formal verification is thus a critical technology in many domains, being essential to safety-critical applications and

Model checking algorithms are widely used for verifying hardware and software models. CRL has pioneered numerous fundamental ideas and algorithms to this field, including "interpolation" as a satisfiability-based proof method which is often dramatically faster and more scalable than prior proof techniques. CBL researchers invented numerous novel methods to automatically reduce the domain of a verification problem through "abstracting" it based upon unsatisfiability proofs. These techniques have substantially increased the scalability of formal verification of complex hardware designs.

Model checking algorithms are widely used for verifying hardware and software models. CRL has pioneered numerous fundamental ideas and algorithms to this field, including "interpolation" as a satisfiability-based proof method which is often dramatically faster and more scalable than prior proof techniques. CBL researchers invented numerous novel methods to automatically reduce the domain of a verification problem through "abstracting" it based upon unsatisfiability proofs. These techniques have substantially increased the scalability of formal verification of complex hardware designs.

CRL researchers have not only used logic optimizations to speed up formal verification algorithms, but are now also applying them to sequential optimization. Sequential synthesis has long been a holy grail in logic optimization. A large part of the design space remains untapped unless one can reliably and effectively optimize and verify in the sequential domain. Recent progress from CRL shows that there is some promise we can tap into this some time in the not too distant future.

Leon

Leon Stok
Director,
Electronic Design Automation
IBM Corporation

Interpolation Within Logic

- Simpler proofs of known properties: Beth definability, Robinson's theorem.
- Interpolant structure: Lyndon Interpolation theorems (1959).
- Preservation under homomorphisms (connections to finite-model theory).

- Many-sorted and Infinitary logics: Feferman '68, '74, Lopez-Escobar '65, Barwise '69, Stern '75, Otto '00.
- Model theoretic characterizations: See Makowsky '85 for a survey.
- Amalgamation: See Czelakowski and Pigozzi '95.

- Guarded fragment: Hoogland, Marx, Otto '00.
- Modal and fixed point logics: Maksimova '79, '91, Ten Cate '05.
- Uniform interpolation: Pitt '92, Visser '96, d'Agostino, Hollenberg '00.

Interpolation and Complexity Theory

1957	1960	1970	1980	1990	2000	2010

1971	1971, Cook. The Complexity of Theorem Proving Procedures
1982	Mundici, NP and Craig's Interpolation Theorem (pub. 1984)
1983	Mundici, A Lower bound for the complexity of Craig's Interpolants in Sentential Logic

Theorem. (Mundici, 1982) At least one of the following is true.

1. P $=$ NP.
2. NP $\neq$ coNP.
3. For F and G in propositional logic, such that $F \implies G$, an interpolant is not computable in time polynomial in the size of F and G.

Interpolation and (Proof) Complexity Theory

1957	1960	1970	1980	1990	2000	2010

1997	Jan Krajíček, Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic.
	Pudlák, Lower Bounds for Resolution and Cutting Plane Proofs and Monotone Computations

A proof system $\vdash$ has feasible interpolation if, whenever there is a short refutation of $A \wedge B$, the interpolant is computable in polynomial time in the size of the proof.

Lemma If there is a resolution refutation of size n for a formula $A \wedge B$, there is an interpolant of circuit size $3n$ that is computable in time n.

Interpolants in Automated Reasoning

1957	1960	1970	1980	1990	2000	2010

1995	Huang, Constructing Craig Interpolation Formulas. (OTTER)
2001	Amir, McIlraith, Partition-Based Logical Reasoning.
2003	McMillan, Interpolation and SAT-Based Model Checking.
2004	Henziger, Jhala,Majumdar,McMillan, Abstractions from Proofs
2005	McMillan, An Interpolating Theorem Prover

1	A Brief History of Interpolation
2	Analysis with Interpolants
3	Labelled Interpolation Systems
4	Current and Future Directions

A Fundamental Problem in Program Verification

```
   int x = i;
int y = j;
while (foo()) {
// Code that does not
// modify x,y,i,j.
    x = y + 1;
    y = x + 1;
}
if (i = j && x <= 10)
 assert(y <= 10);
```

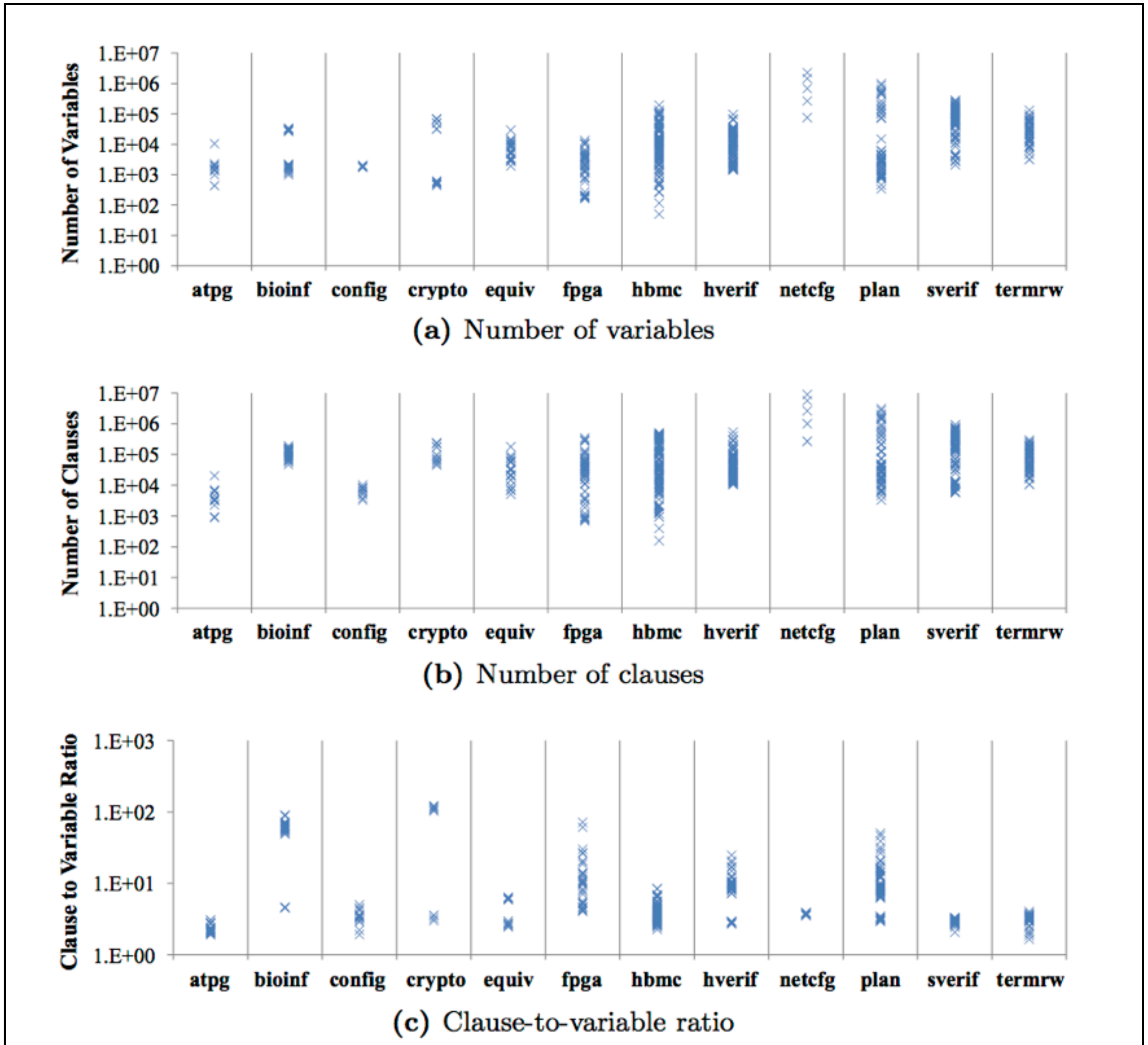
- The assertion checking problem.
- More generally, a safety property, of a discrete, state transition system can be reduced to reachability.
- Manual proof would use Hoare logic and invariants.

Bounded Execution as a Formula

```
   int x = i;
int y = j;
while (foo()) {
// Code that does not
// modify x,y,i,j.
    x = y + 1;
    y = x + 1;
}
if (i = j && x <= 10)
 assert(y <= 10);
```

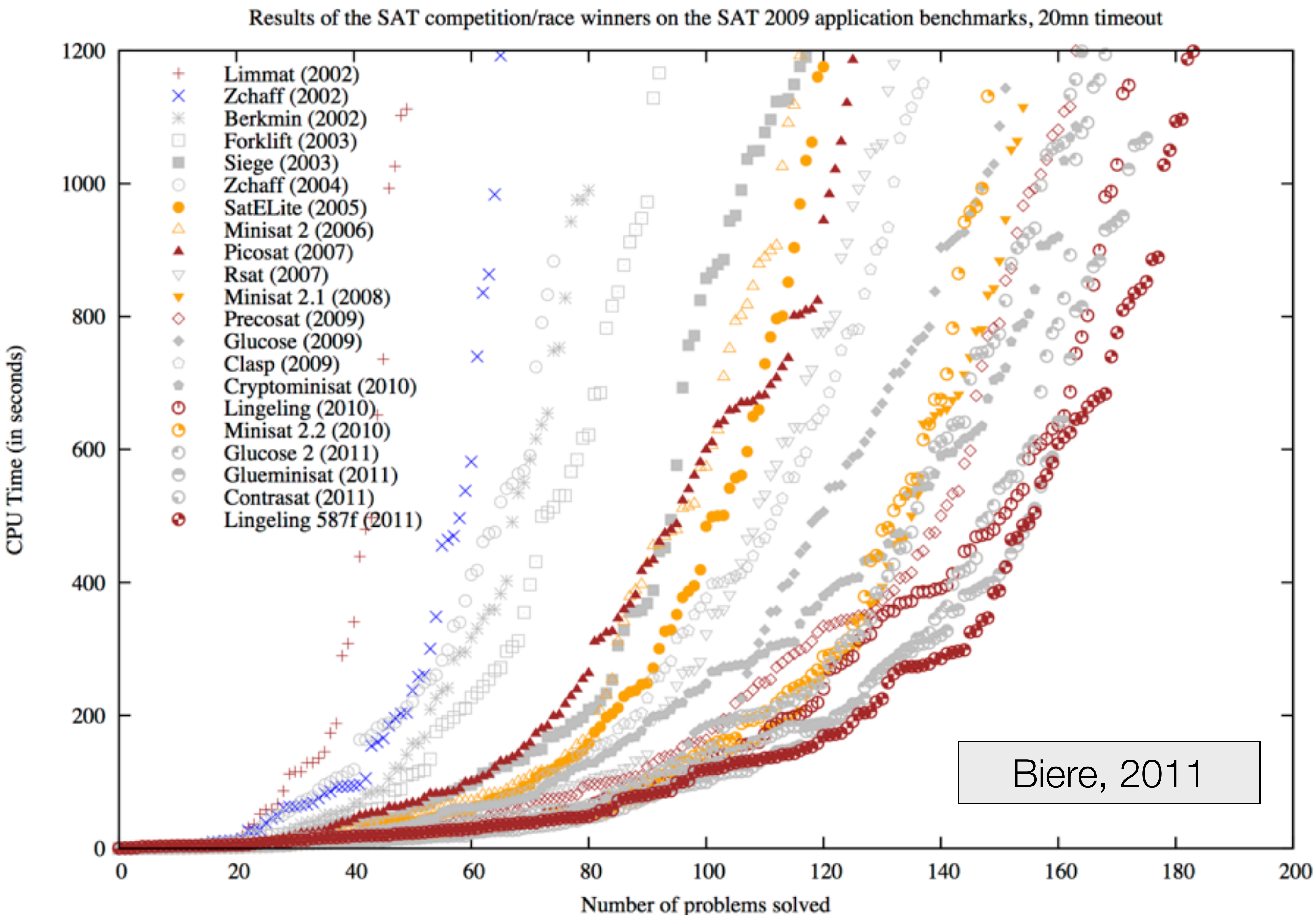
```
x0 = i and
y0 = j and
x1 = y0 + 1 and
y1 = x0 + 1 and
x2 = y1 + 1 and
y2 = x1 + 1 and

x3 = y2 + 1 and
y3 = x2 + 1 and
(i = j and x3 <= 10)
  implies (y3 > 10)
```

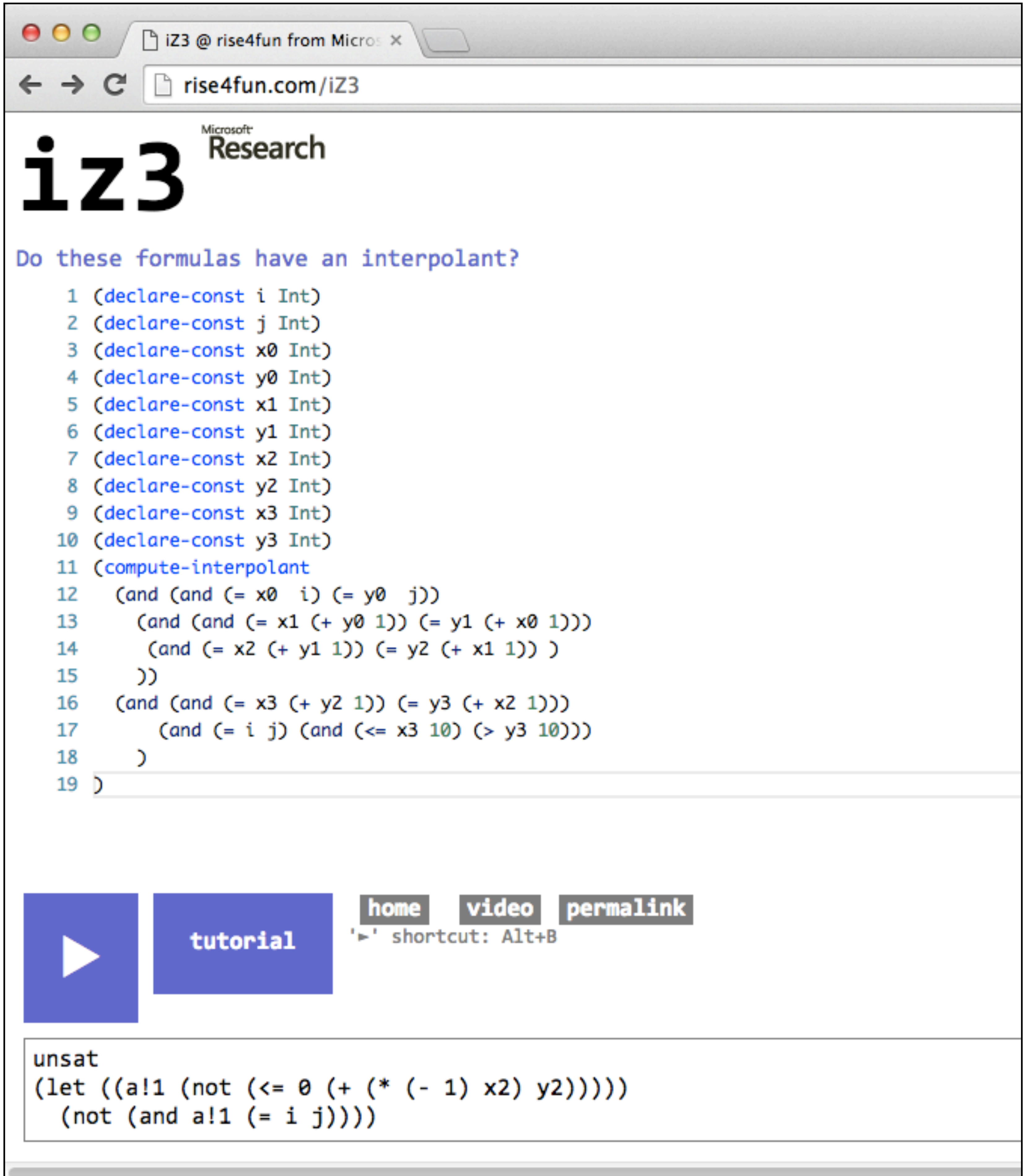
Empirical Progress in SAT Solving



Katebi, Sakallah, Marques-Silva, 2011

Empirical Progress in SAT Solving




iZ3 @ rise4fun from Micros
rise4fun.com/iZ3
iz3
Microsoft Research
Do these formulas have an interpolant?
(declare-const i Int)
(declare-const j Int)
(declare-const x0 Int)
(declare-const y0 Int)
(declare-const x1 Int)
(declare-const y1 Int)
(declare-const x2 Int)
(declare-const y2 Int)
(declare-const x3 Int)
(declare-const y3 Int)
(compute-interpolant
(and (and (= x0 i) (= y0 j))
(and (and (= x1 (+ y0 1)) (= y1 (+ x0 1)))
(and (= x2 (+ y1 1)) (= y2 (+ x1 1)))
))
(and (and (= x3 (+ y2 1)) (= y3 (+ x2 1)))
(and (= i j) (and (<= x3 10) (> y3 10)))
)
)
tutorial
home
video
permalink
'►' shortcut: Alt+B
unsat
(let ((a!1 (not (<= 0 (+ (* (- 1) x2) y2)))))
(not (and a!1 (= i j))))

Interpolants from Bounded Executions

```
x0 = i and
y0 = j and
x1 = y0 + 1 and
y1 = x0 + 1 and
x2 = y1 + 1 and
y2 = x1 + 1 and
```

A

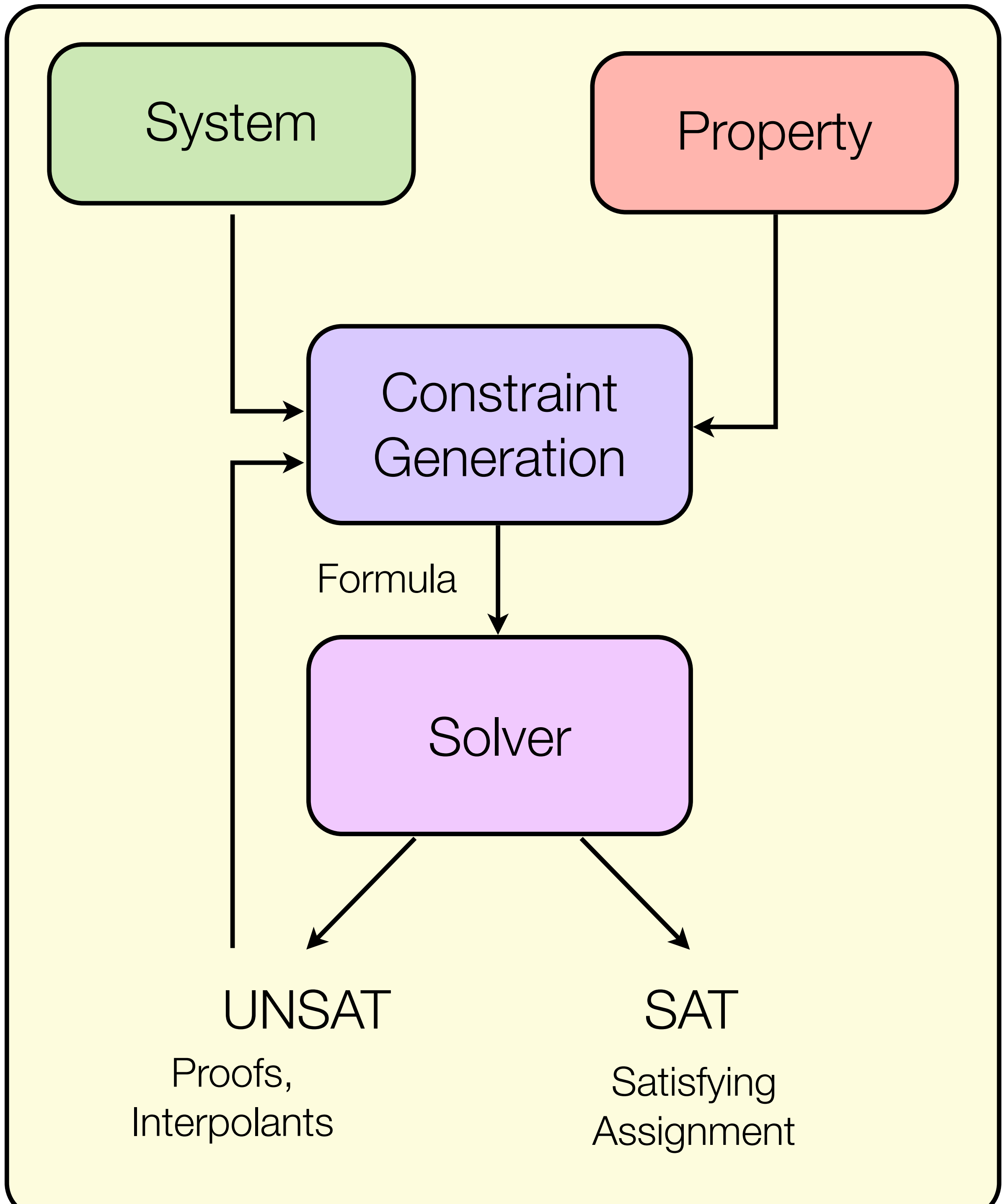
```
x3 = y2 + 1 and
y3 = x2 + 1 and
(i = j and x3 <= 10)
  implies (y3 > 10)
```

B

$$i = j \implies x_2 \leq y_2$$

- Interpolant is with respect to a theory.
- Computed from a proof produced by solver for the theory.
- After renaming, we have an invariant.
- Invariant generation typically involves a series of quantifier elimination steps, or fixed point computation.

Analysis of a System with Interpolants



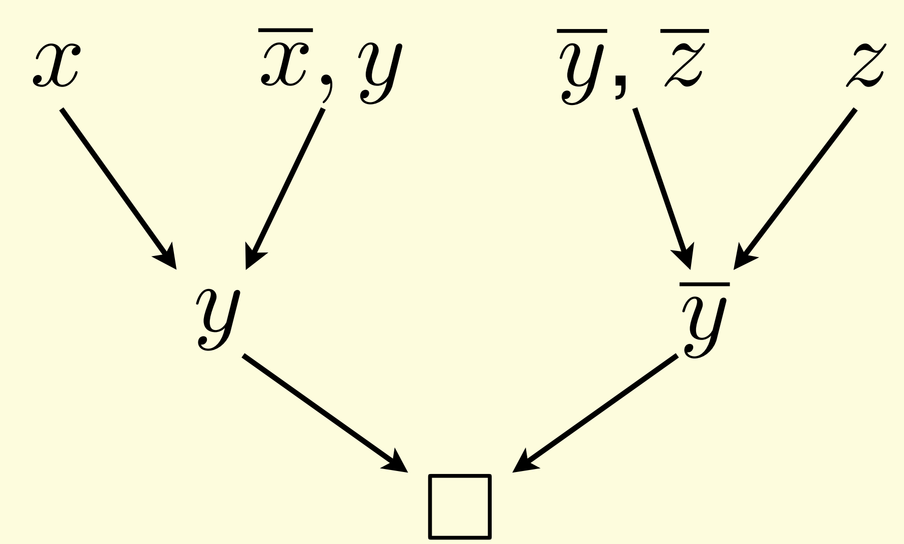
- A poor person's quantifier elimination.
- Analysis algorithms involve repeated calls to a solver and repeated computation of invariants.
- Solvers: Efficient in practice contrary to theoretical expectations.
- Proof generation: Arose from theory to explain practice.
- Efficient interpolation: First studied in theory, applied in practice, leading to more theory.

1	A Brief History of Interpolation
2	Analysis with Interpolants
3	Labelled Interpolation Systems
4	Current and Future Directions

Terminology

Var	Boolean variables: $a_1, a_2, a_3, \ldots$
Literal	Variable or its negation: $a, \overline{a}, \neg a$
Clause	Disjunction or set of literals: $\{a_1, a_2, a_5\}$
CNF Formula	Conjunction or set of clauses: $\{\{a\}, \{\overline{a}, b\}\}$

$$\frac{C \vee x \qquad \overline{x} \vee D}{C \vee D} \quad \text{[Resolution]}$$



Interpolating Proof Rules

A-Hyp $\dfrac{}{C \quad [\{\ell \in C \mid var(\ell) \in B\}]}$ $[C \in A]$

B-Hyp $\dfrac{}{C \quad [\top]}$ $(C \in B)$

A-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$ $(x \in var(A) \setminus var(B))$

B-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$ $(x \in var(B))$

McMillan, 2003

Interpolating Proof Rules

Split rules based on vocabulary

A-Hyp $\quad \dfrac{}{C \quad [\{\ell \in C \mid var(\ell) \in B\}]} \quad [C \in A]$

B-Hyp $\quad \dfrac{}{C \quad [\top]} \quad (C \in B)$

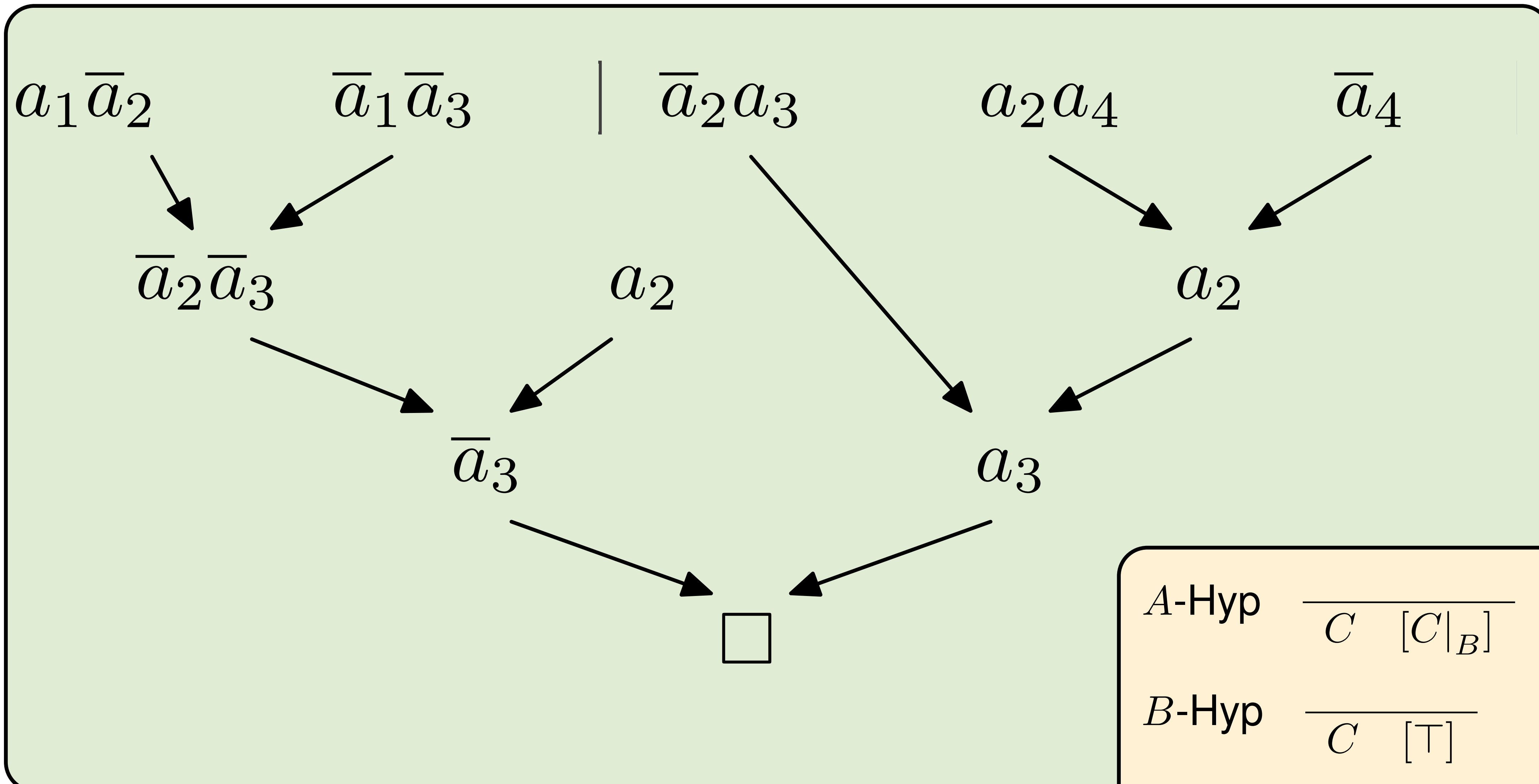
A-Res $\quad \dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]} \quad (x \in var(A) \setminus var(B))$

B-Res $\quad \dfrac{C \vee x\ \boxed{[I_1]} \qquad \overline{x} \vee D\ \boxed{[I_2]}}{C \vee D \quad \boxed{[I_1 \wedge I_2]}} \quad (x \in var(B))$

McMillan, 2003

Annotate formulae with *Partial Interpolants*

Applying Interpolating Proof Rules



$$A = (a_1 \vee \overline{a}_2) \wedge (\overline{a}_1 \vee \overline{a}_3) \wedge a_2$$

$$B = (\overline{a}_2 \vee a_3) \wedge (a_2 \vee a_4) \wedge \overline{a}_4$$

$$I =$$

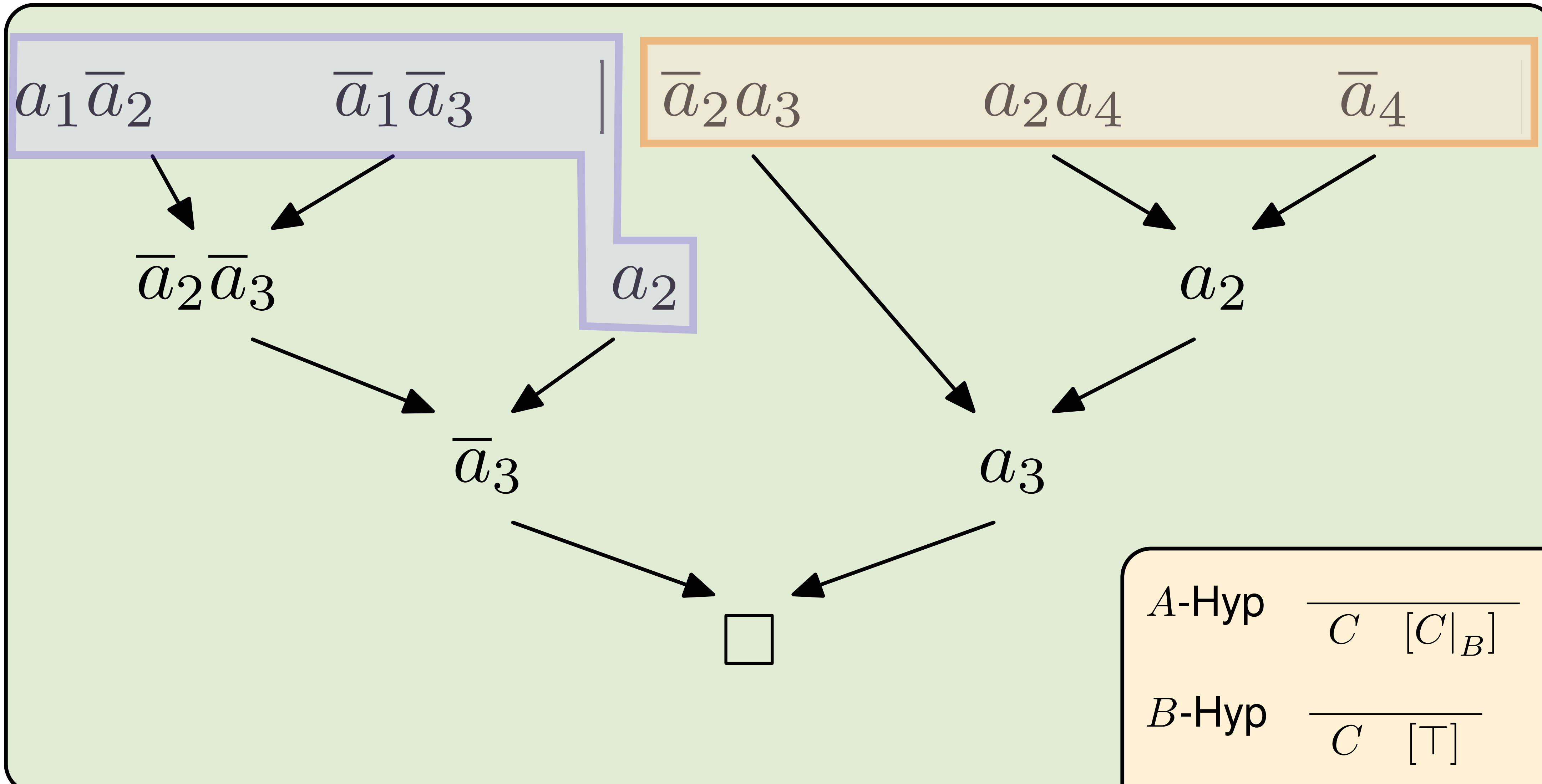
A-Hyp $\dfrac{}{C \quad [C|_B]}$

B-Hyp $\dfrac{}{C \quad [\top]}$

A-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$

B-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$

Applying Interpolating Proof Rules



A-Hyp $\quad \dfrac{}{C \quad [C|_B]}$

B-Hyp $\quad \dfrac{}{C \quad [\top]}$

A-Res $\quad \dfrac{C \vee x \; [I_1] \qquad \overline{x} \vee D \; [I_2]}{C \vee D \quad [I_1 \vee I_2]}$

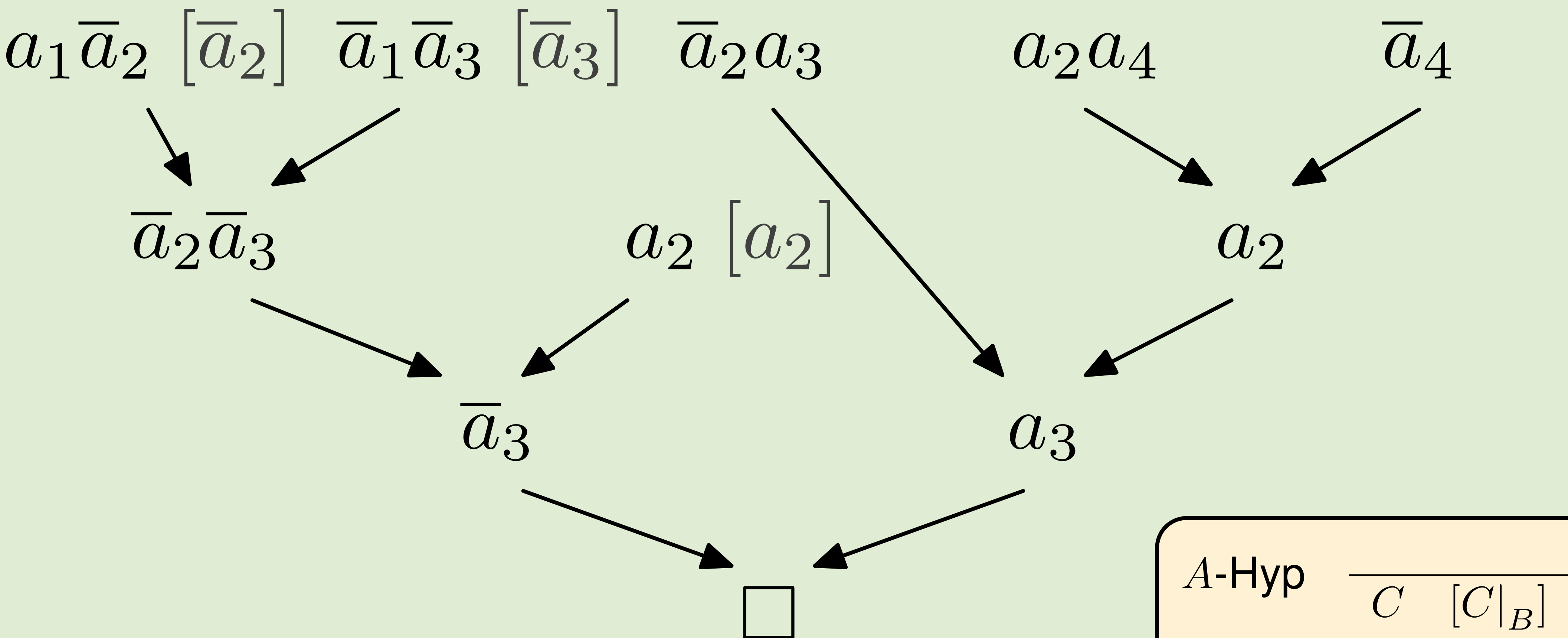
B-Res $\quad \dfrac{C \vee x \; [I_1] \qquad \overline{x} \vee D \; [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$

$A = (a_1 \vee \overline{a}_2) \wedge (\overline{a}_1 \vee \overline{a}_3) \wedge a_2$

$B = (\overline{a}_2 \vee a_3) \wedge (a_2 \vee a_4) \wedge \overline{a}_4$

$I =$

Applying Interpolating Proof Rules



A-Hyp $\dfrac{}{C \quad [C|_B]}$

B-Hyp $\dfrac{}{C \quad [\top]}$

A-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$

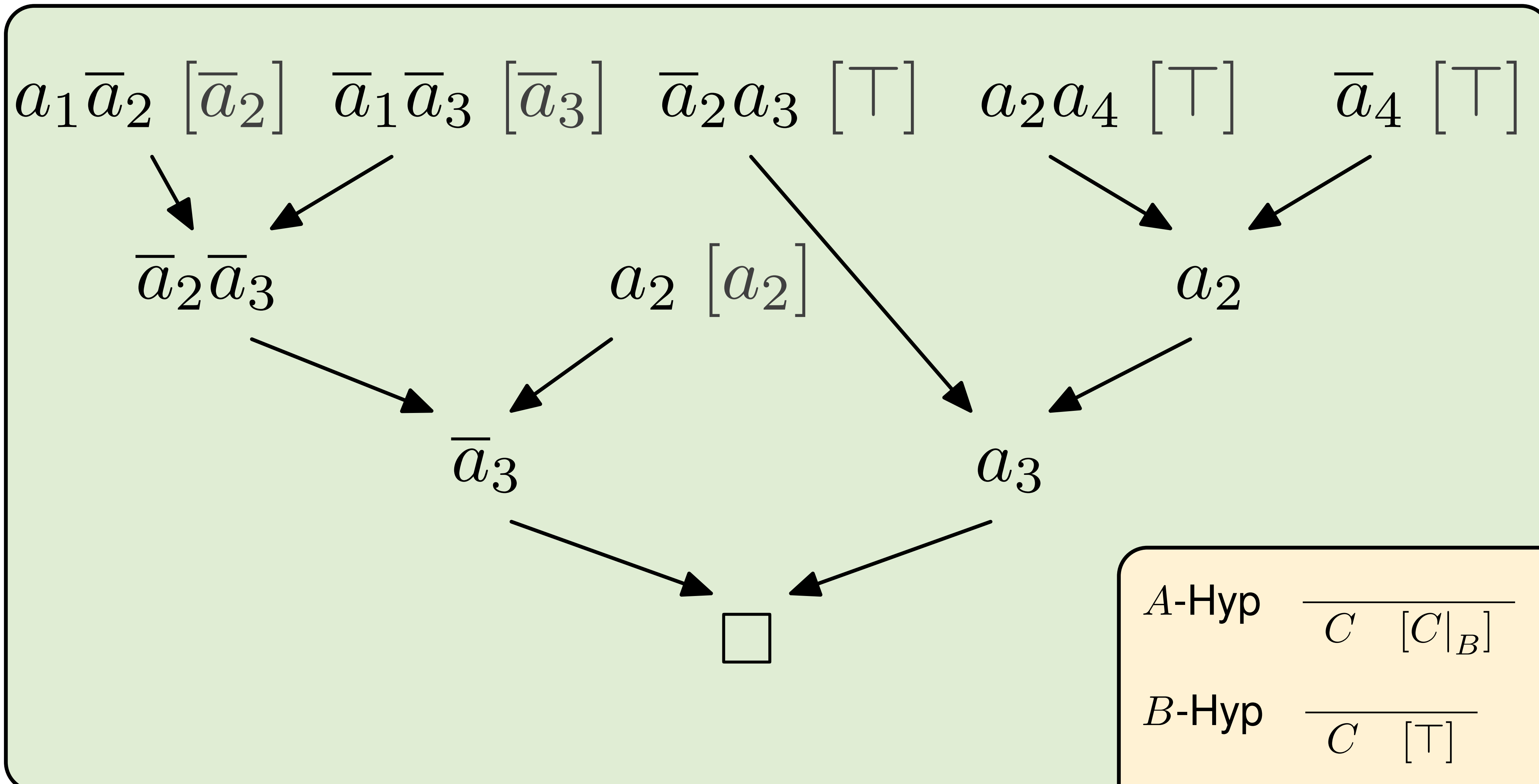
B-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$

$$A = (a_1 \vee \overline{a}_2) \wedge (\overline{a}_1 \vee \overline{a}_3) \wedge a_2$$

$$B = (\overline{a}_2 \vee a_3) \wedge (a_2 \vee a_4) \wedge \overline{a}_4$$

$$I =$$

Applying Interpolating Proof Rules



$$A\text{-Hyp}\quad \frac{}{C \quad [C|_B]}$$

$$B\text{-Hyp}\quad \frac{}{C \quad [\top]}$$

$$A\text{-Res}\quad \frac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$$

$$B\text{-Res}\quad \frac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$$

$$A = (a_1 \vee \overline{a}_2) \wedge (\overline{a}_1 \vee \overline{a}_3) \wedge a_2$$

$$B = (\overline{a}_2 \vee a_3) \wedge (a_2 \vee a_4) \wedge \overline{a}_4$$

$$I =$$

Applying Interpolating Proof Rules

$$
\begin{tikzcd}[column sep=small]
a_1\overline{a}_2\ [\overline{a}_2] \arrow[d] & \overline{a}_1\overline{a}_3\ [\overline{a}_3] \arrow[dl] & & \overline{a}_2 a_3\ [\top] \arrow[dd] & a_2 a_4\ [\top] \arrow[d] & \overline{a}_4\ [\top] \arrow[dl] \\
\overline{a}_2\overline{a}_3\ [\overline{a}_2 \vee \overline{a}_3] \arrow[dr] & & a_2\ [a_2] \arrow[dl] & & a_2\ [\top] \arrow[dl] & \\
& \overline{a}_3 \arrow[dr] & & a_3 \arrow[dl] & & \\
& & \square & & &
\end{tikzcd}
$$

$$A = (a_1 \vee \overline{a}_2) \wedge (\overline{a}_1 \vee \overline{a}_3) \wedge a_2$$

$$B = (\overline{a}_2 \vee a_3) \wedge (a_2 \vee a_4) \wedge \overline{a}_4$$

$$I =$$

A-Hyp $\dfrac{}{C \quad [C|_B]}$

B-Hyp $\dfrac{}{C \quad [\top]}$

A-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$

B-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$

Applying Interpolating Proof Rules

$$
\begin{tikzcd}[column sep=tiny]
a_1\overline{a}_2\ [\overline{a}_2] \arrow[dr] & & \overline{a}_1\overline{a}_3\ [\overline{a}_3] \arrow[dl] & & \overline{a}_2 a_3\ [\top] \arrow[ddrr] & & a_2 a_4\ [\top] \arrow[dr] & & \overline{a}_4\ [\top] \arrow[dl] \\
& \overline{a}_2\overline{a}_3\ [\overline{a}_2 \vee \overline{a}_3] \arrow[dr] & & a_2\ [a_2] \arrow[dl] & & & & a_2\ [\top] \arrow[dl] & \\
& & \overline{a}_3\ [\overline{a}_3 \wedge a_2] \arrow[drr] & & & & a_3\ [\top] \arrow[dll] & & \\
& & & & \square\ [\overline{a}_3 \wedge a_2] & & & &
\end{tikzcd}
$$

A-Hyp $\quad \dfrac{}{C \quad [C|_B]}$

B-Hyp $\quad \dfrac{}{C \quad [\top]}$

A-Res $\quad \dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$

B-Res $\quad \dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$

$$A = (a_1 \vee \overline{a}_2) \wedge (\overline{a}_1 \vee \overline{a}_3) \wedge a_2$$

$$B = (\overline{a}_2 \vee a_3) \wedge (a_2 \vee a_4) \wedge \overline{a}_4$$

$$I = \overline{a}_3 \wedge a_2$$

A Symmetric Construction

A-Hyp $\dfrac{}{C \quad [\bot]}$ $[C \in A]$ B-Hyp $\dfrac{}{C \quad [\top]}$ $(C \in B)$

A-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$ $(x \in var(A) \setminus var(B))$

AB-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [(x \vee I_1) \wedge (\overline{x} \vee I_2)]}$ $(x \in var(B) \cap var(A))$

B-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$ $[x \in var(B) \setminus var(A)]$

Huang 1995, Krajíček;Pudlák 1997

An Interpolant from the Symmetric Construction

$$
\begin{tikzcd}[column sep=small]
a_1\overline{a}_2\ [\bot] \arrow[dr] & & \overline{a}_1\overline{a}_3\ [\bot] \arrow[dl] & & \overline{a}_2 a_3\ [\top] \arrow[ddr] & & a_2 a_4\ [\top] \arrow[dr] & & \overline{a}_4\ [\top] \arrow[dl] \\
& \overline{a}_2\overline{a}_3\ [\bot] \arrow[dr] & & a_2\ [\bot] \arrow[dl] & & & & a_2\ [\top] \arrow[dll] & \\
& & \overline{a}_3\ [\bot] \arrow[drr] & & & a_3\ [\top] \arrow[dl] & & & \\
& & & & \square\ [\overline{a}_3] & & & &
\end{tikzcd}
$$

$$A = (a_1 \vee \overline{a}_2) \wedge (\overline{a}_1 \vee \overline{a}_3) \wedge a_2$$

$$B = (\overline{a}_2 \vee a_3) \wedge (a_2 \vee a_4) \wedge \overline{a}_4$$

$$I = \overline{a}_3$$

A-Hyp $\dfrac{}{C \quad [\bot]}$ B-Hyp $\dfrac{}{C \quad [\top]}$

A-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$

AB-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [(x \vee I_1) \wedge (\overline{x} \vee I_2)]}$

B-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$

What other constructions are there?

A-Hyp $\dfrac{}{C \quad [\bot]}$ B-Hyp $\dfrac{}{C \quad [\top]}$

A-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$

AB-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [(x \vee I_1) \wedge (\overline{x} \vee I_2)]}$

B-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$

A-Hyp $\dfrac{}{C \quad [C|_B]}$

B-Hyp $\dfrac{}{C \quad [\top]}$

A-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$

B-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$

?

What other constructions are there?

A-Hyp $\dfrac{}{C \quad [\bot]}$ B-Hyp $\dfrac{}{C \quad [\top]}$

A-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$

AB-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [(x \vee I_1) \wedge (\overline{x} \vee I_2)]}$

B-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$

A-Hyp $\dfrac{}{C \quad [C|_B]}$

B-Hyp $\dfrac{}{C \quad [\top]}$

A-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$

B-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$

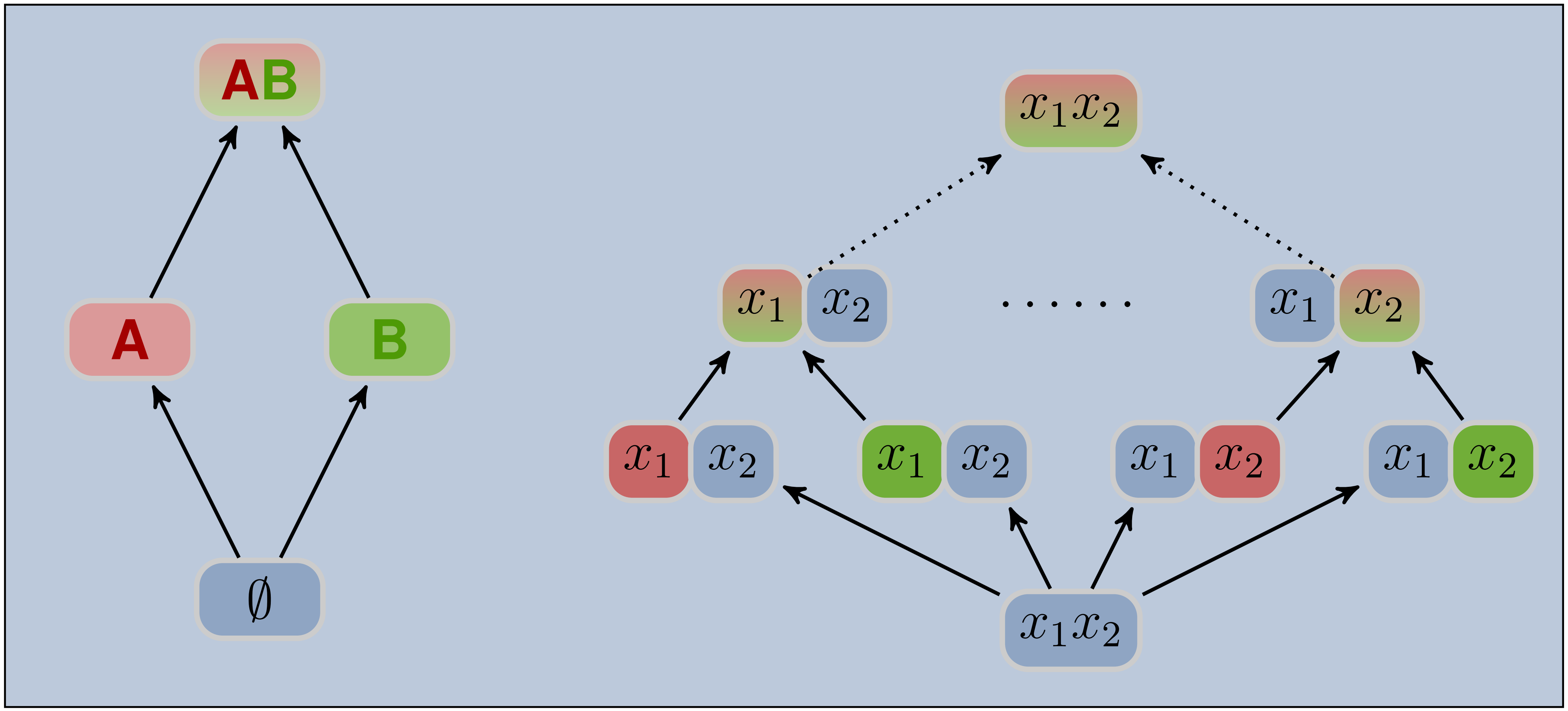
A-Hyp $\dfrac{}{C \quad [\bot]}$

B-Hyp $\dfrac{}{C \quad [\neg C|_A]}$

A-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$

B-Res $\dfrac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$

Labelled Formulae



Colours : $\mathcal{S} \stackrel{\text{def}}{=} \{\emptyset, \textbf{A}, \textbf{B}, \textbf{AB}\}$

Coloured clauses: $C \rightarrow \mathcal{S}$, a lattice under point-wise order.

Coloured CNF: Set of coloured clauses.

Deduction and Interpolation with Labels

Let $\sigma(x)$ be the colour of a literal x. $C|_{\mathbf{A}} = \{x \in C \mid \sigma(x) \sqsubseteq \mathbf{A}\}$

$$A\text{-Hyp} \quad \frac{}{C\ [C|_{\mathbf{B}}]} \quad C \in A \qquad\qquad B\text{-Hyp} \quad \frac{}{C\ [C|_{\mathbf{A}}]} \quad C \in B$$

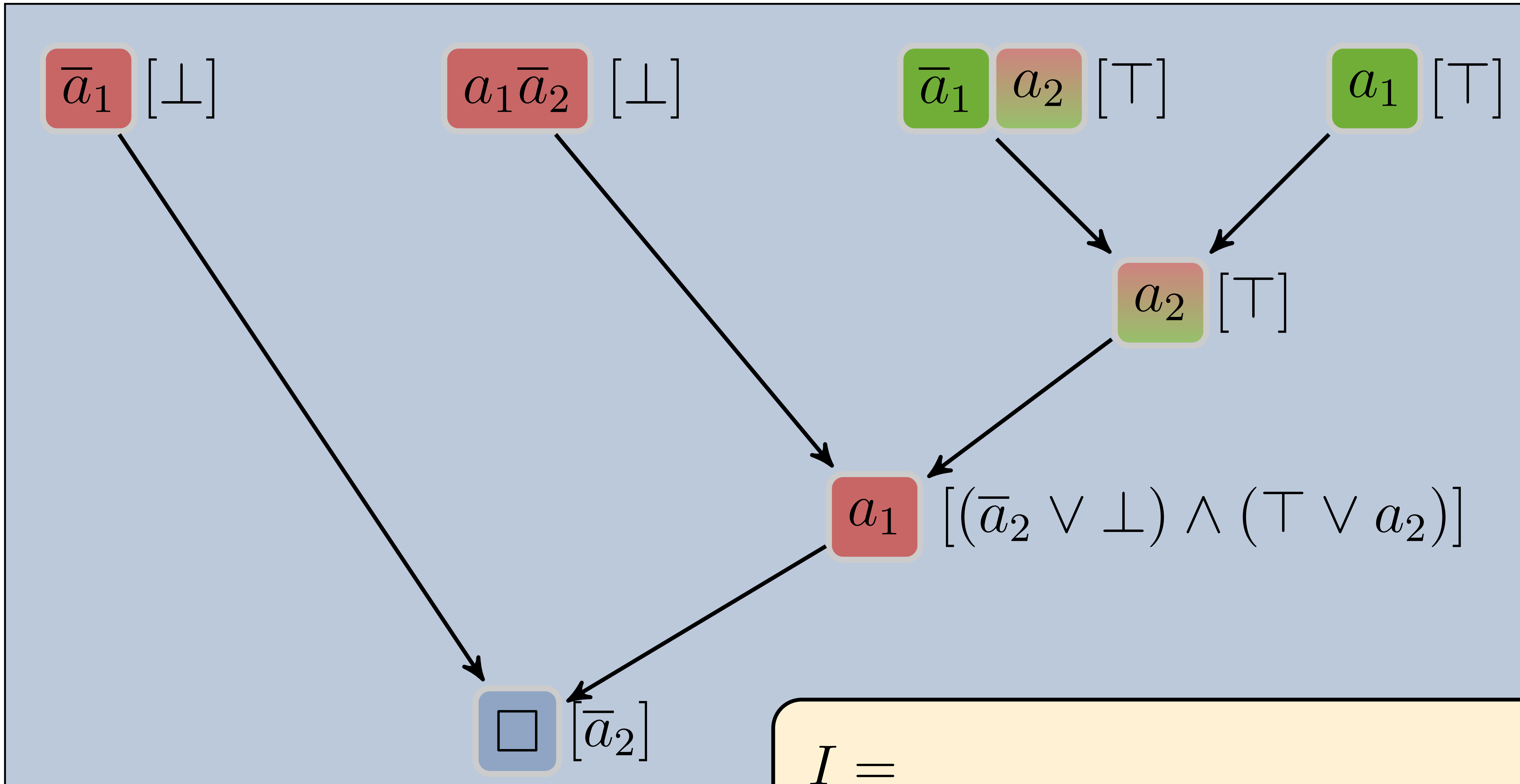
$$A\text{-Res} \quad \frac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]} \quad (\sigma(x) \sqcup \sigma(\overline{x}) = \mathbf{A})$$

$$AB\text{-Res} \quad \frac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [(x \vee I_1) \wedge (I_2 \vee \overline{x})]} \quad (\sigma(x) \sqcup \sigma(\overline{x}) = \mathbf{AB})$$

$$B\text{-Res} \quad \frac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]} \quad (\sigma(x) \sqcup \sigma(\overline{x}) = \mathbf{B})$$

D'Silva, Kroening, Purandare, Weissenbacher, 2010

Applying the Labelled Interpolation System



$$I = \begin{cases} I_1 \vee I_2 \text{ if } \sigma(x) \sqcup \sigma(\overline{x}) = \textbf{A} \\ (x \vee I_1) \wedge (I_2 \vee \overline{x}) \text{ if } \sigma(x) \sqcup \sigma(\overline{x}) = \textbf{AB} \\ I_1 \wedge I_2 \text{ if } \sigma(x) \sqcup \sigma(\overline{x}) = \textbf{B} \end{cases}$$

$$A = \overline{a}_1 \wedge (a_1 \vee \overline{a}_2)$$

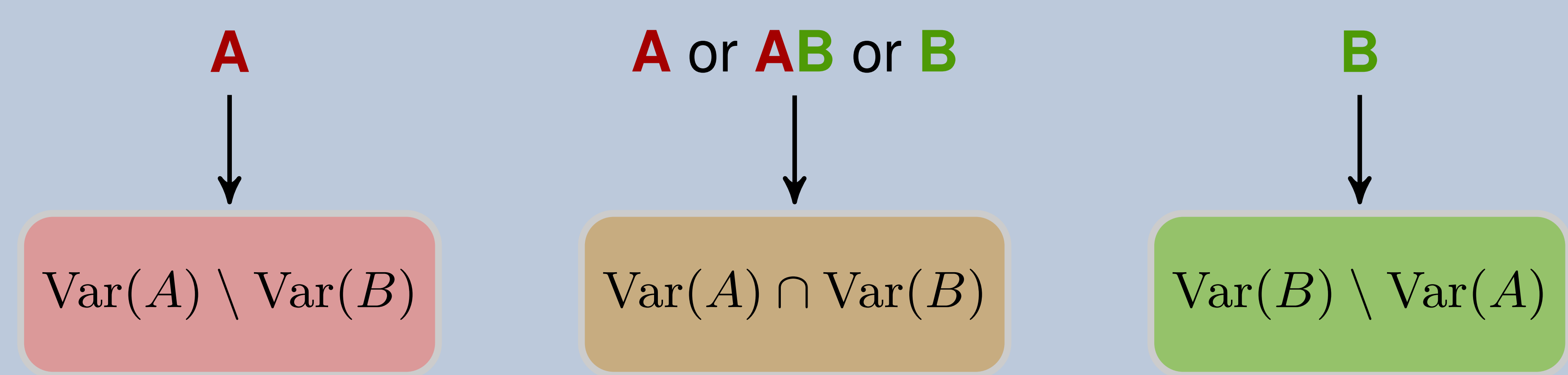
$$B = a_1 \wedge (\overline{a}_1 \vee a_2)$$

$$I = \overline{a}_2$$

Correctness

A colouring of $A \wedge B$ is *locality preserving* if

- Every literal in the formula has a non-empty colour,
- every literal occurring only in A is coloured **A**, and
- every literal occurring only in B is coloured **B**.



Theorem. If $A \wedge B$ is unsatisfiable and has a locality preserving colouring, $\Box\ [I]$ is derivable and I an interpolant for A and B.

Proof adapts an invariant from: *A Combination Method for Generating Interpolants*, Yorsh and Musuvathi, Conference on Automated Deduction, 2005.

It's all in the colour

$\mathrm{Var}(A) \setminus \mathrm{Var}(B)$

$\mathrm{Var}(A) \cap \mathrm{Var}(B)$

$\mathrm{Var}(B) \setminus \mathrm{Var}(A)$

A-Hyp $$\frac{}{C \quad [C|_B]}$$

B-Hyp $$\frac{}{C \quad [\top]}$$

A-Res $$\frac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$$

B-Res $$\frac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$$

A-Hyp $$\frac{}{C \quad [\bot]}$$ B-Hyp $$\frac{}{C \quad [\top]}$$

A-Res $$\frac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$$

AB-Res $$\frac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [(x \vee I_1) \wedge (\overline{x} \vee I_2)]}$$

B-Res $$\frac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$$

$\mathrm{Var}(A) \setminus \mathrm{Var}(B)$

$\mathrm{Var}(A) \cap \mathrm{Var}(B)$

$\mathrm{Var}(B) \setminus \mathrm{Var}(A)$

$\mathrm{Var}(A) \setminus \mathrm{Var}(B)$

$\mathrm{Var}(A) \cap \mathrm{Var}(B)$

$\mathrm{Var}(B) \setminus \mathrm{Var}(A)$

A-Hyp $$\frac{}{C \quad [\bot]}$$

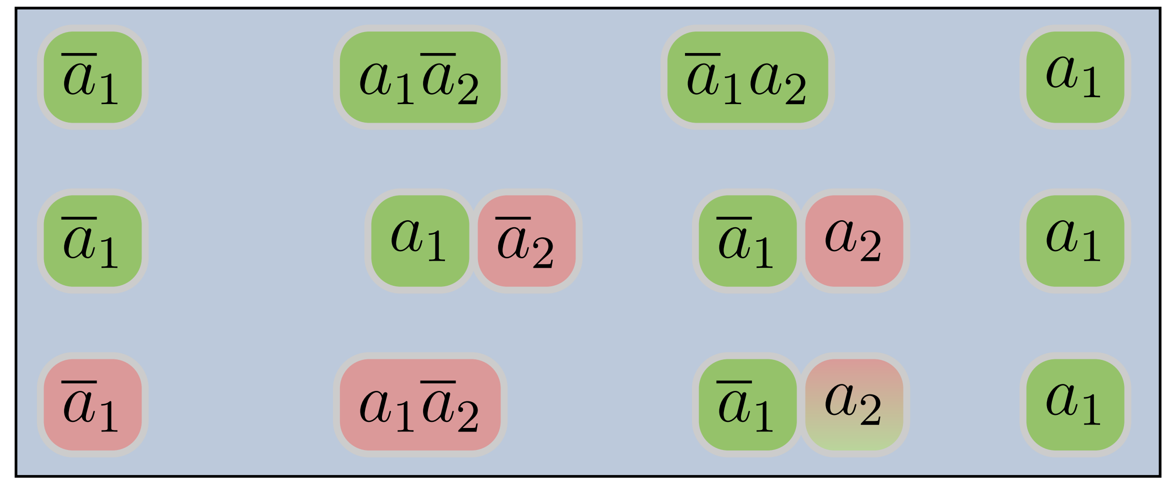
B-Hyp $$\frac{}{C \quad [\neg C|_A]}$$

A-Res $$\frac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \vee I_2]}$$

B-Res $$\frac{C \vee x\ [I_1] \qquad \overline{x} \vee D\ [I_2]}{C \vee D \quad [I_1 \wedge I_2]}$$

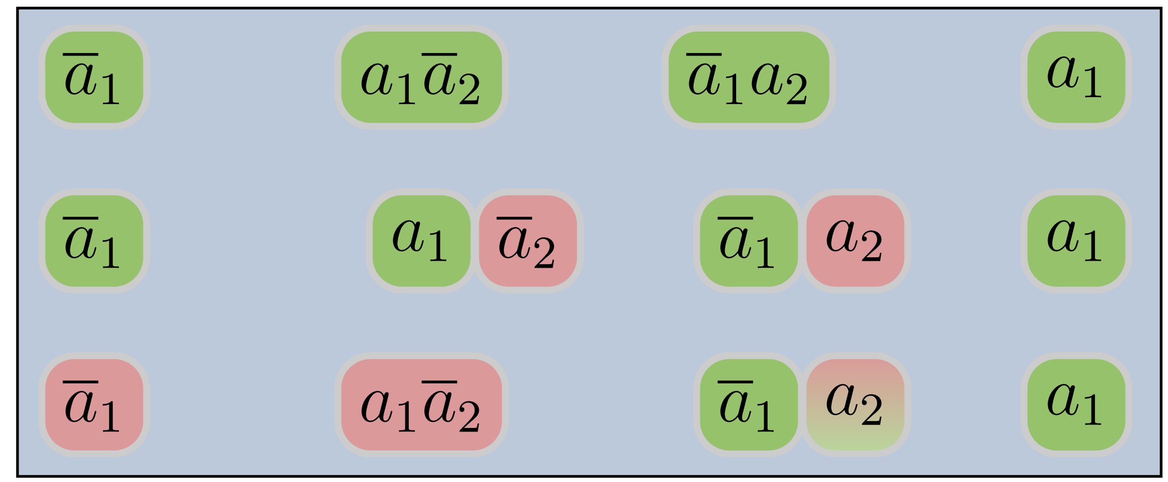
But why those constructions?

A colouring is *partitioning* if every instance of a variable has the same colour.

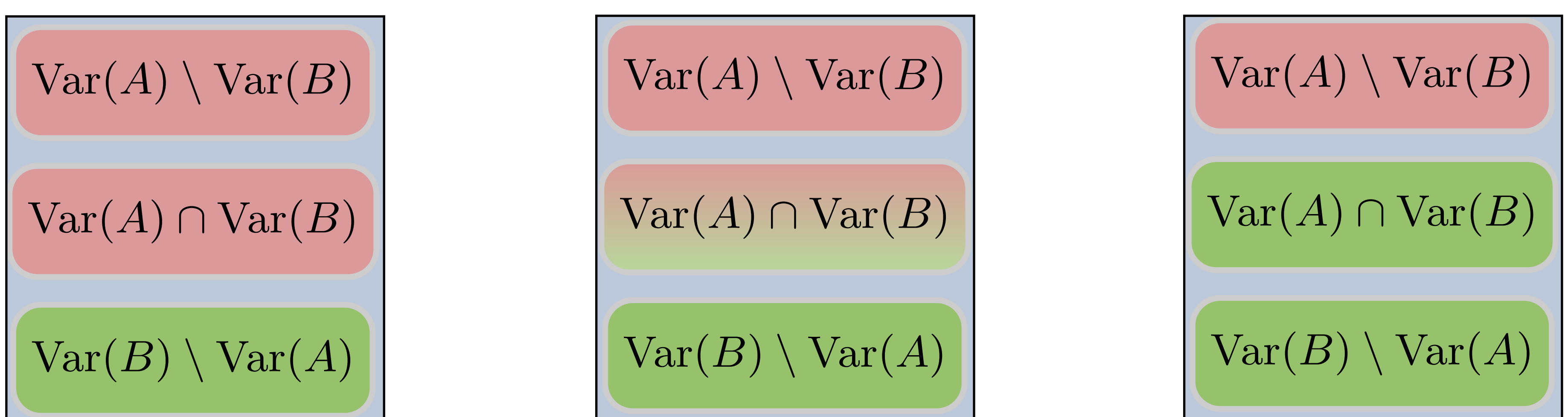


But why those constructions?

A colouring is *partitioning* if every instance of a variable has the same colour.



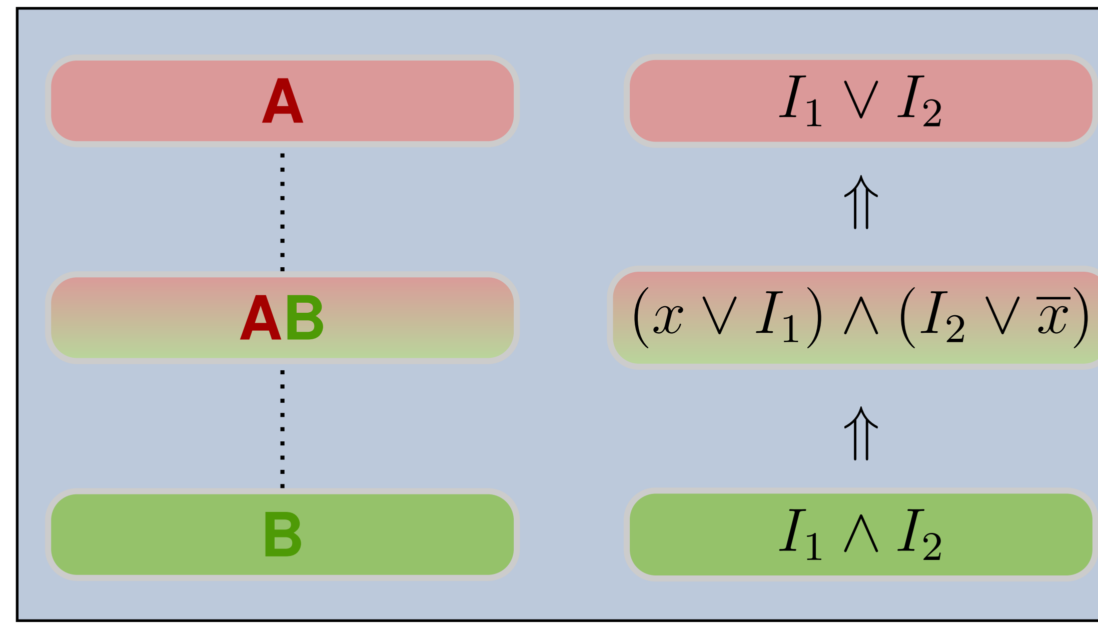
Theorem. There is a unique, coarsest partition that admits exactly three, locality preserving colourings.



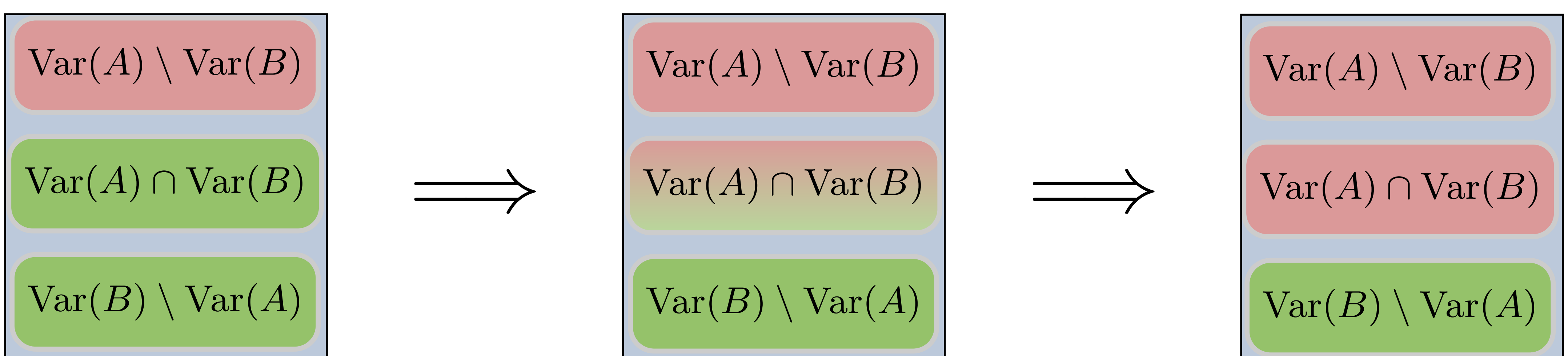
Interpolant Strength

The *strength order* is **B** $\sqsubseteq$ **AB** $\sqsubseteq$ **A**.

Coloured clauses and CNF are ordered pointwise by the strength order.



Theorem. The set of locality-preserving colourings forms a complete lattice with respect to the strength order.

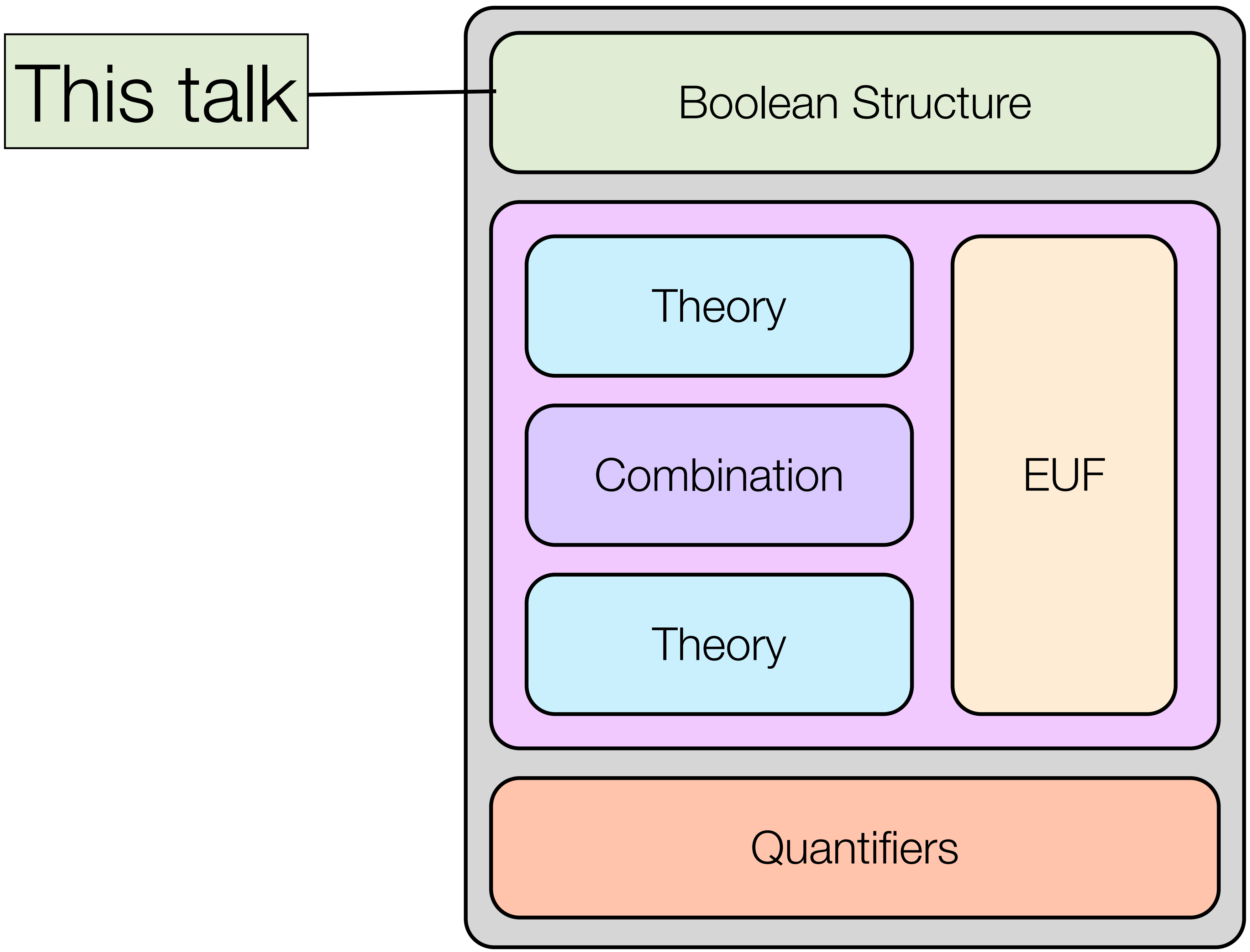


Additional Analysis

- Colourings can be ordered by variable occurrence, which correlates loosely with interpolant size.
- There is a dual operation on the lattice of colours, which lifts pointwise so that every interpolation construction has a dual.
- Sharygina et al. proved results on labelled interpolation applied in the context of reachability analysis.
- Jhala and McMillan, 2006 and Albarghouthi and McMillan, 2013 study additional restrictions on the vocabulary condition.

1	A Brief History of Interpolation
2	Analysis with Interpolants
3	Labelled Interpolation Systems
4	Current and Future Directions

Architecture of a Modern Solver



Equality Proofs

$$\begin{tikzcd}
f(u,y) = z \arrow[dddr] & & u = x \arrow[dr] & & v = y \arrow[dl] & & f(x,v) \neq z \arrow[ddl] \\
& & & \boxed{f(u,y) = f(x,v)} \arrow[drr] & & & \\
& & & & & f(u,y) \neq z \arrow[dllll] & \\
& \Box & & & & &
\end{tikzcd}$$

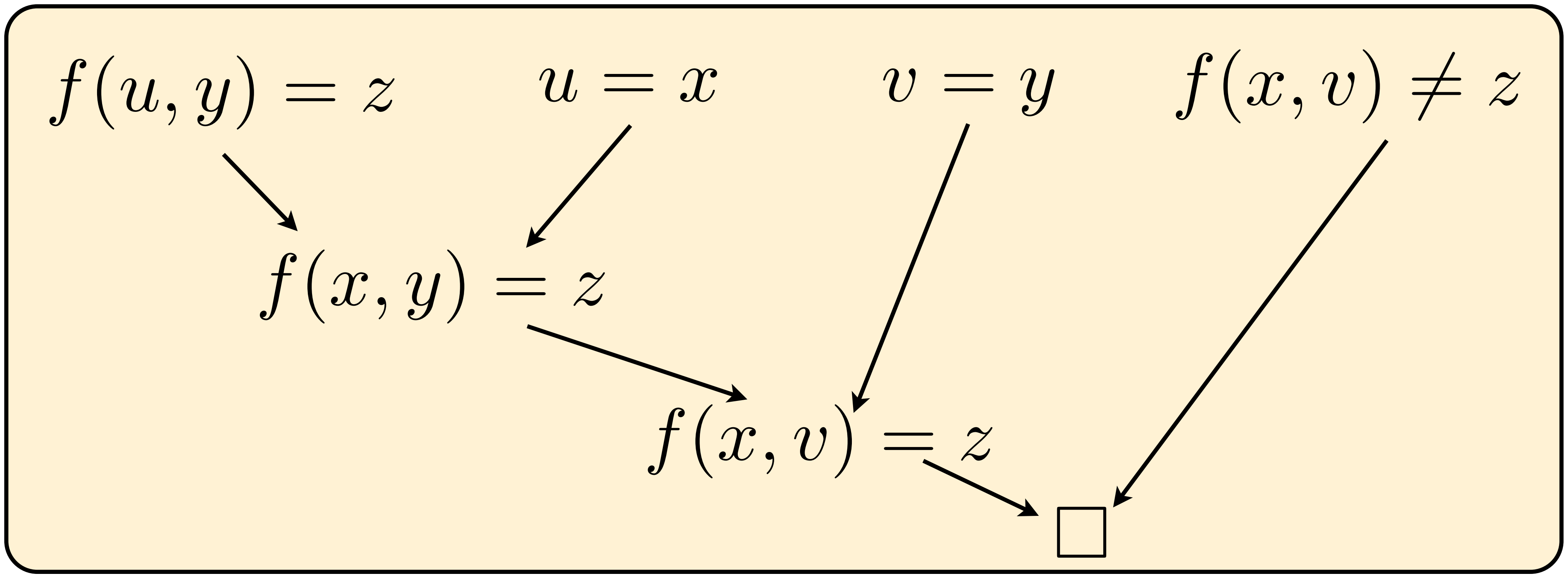
$$A = u = x \wedge f(u,y) = z$$

$$B = v = y \wedge f(x,v) \neq z$$

$$I = f(x,y) = z$$

- Deduced *literals* may not be in A or in B
- New *terms* may use non-shared symbols
- Interpolant may be over terms not in the proof

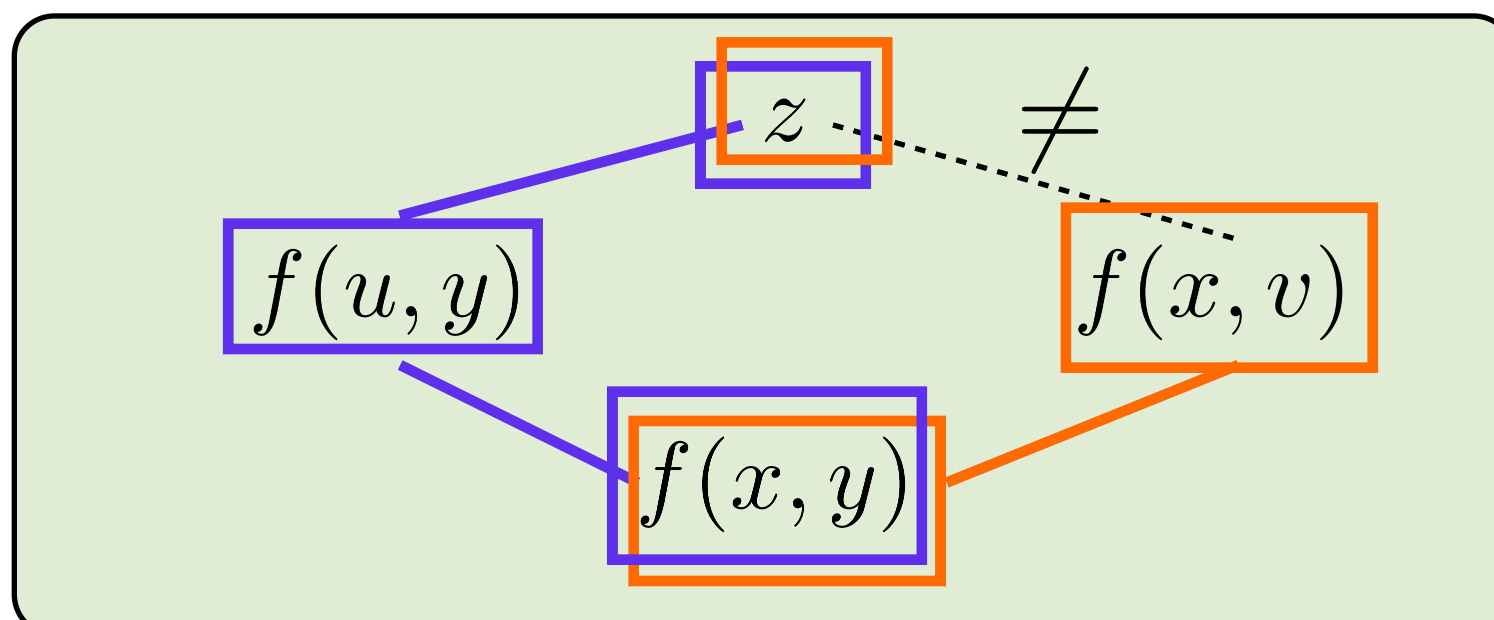
Coloured Congruence Graphs

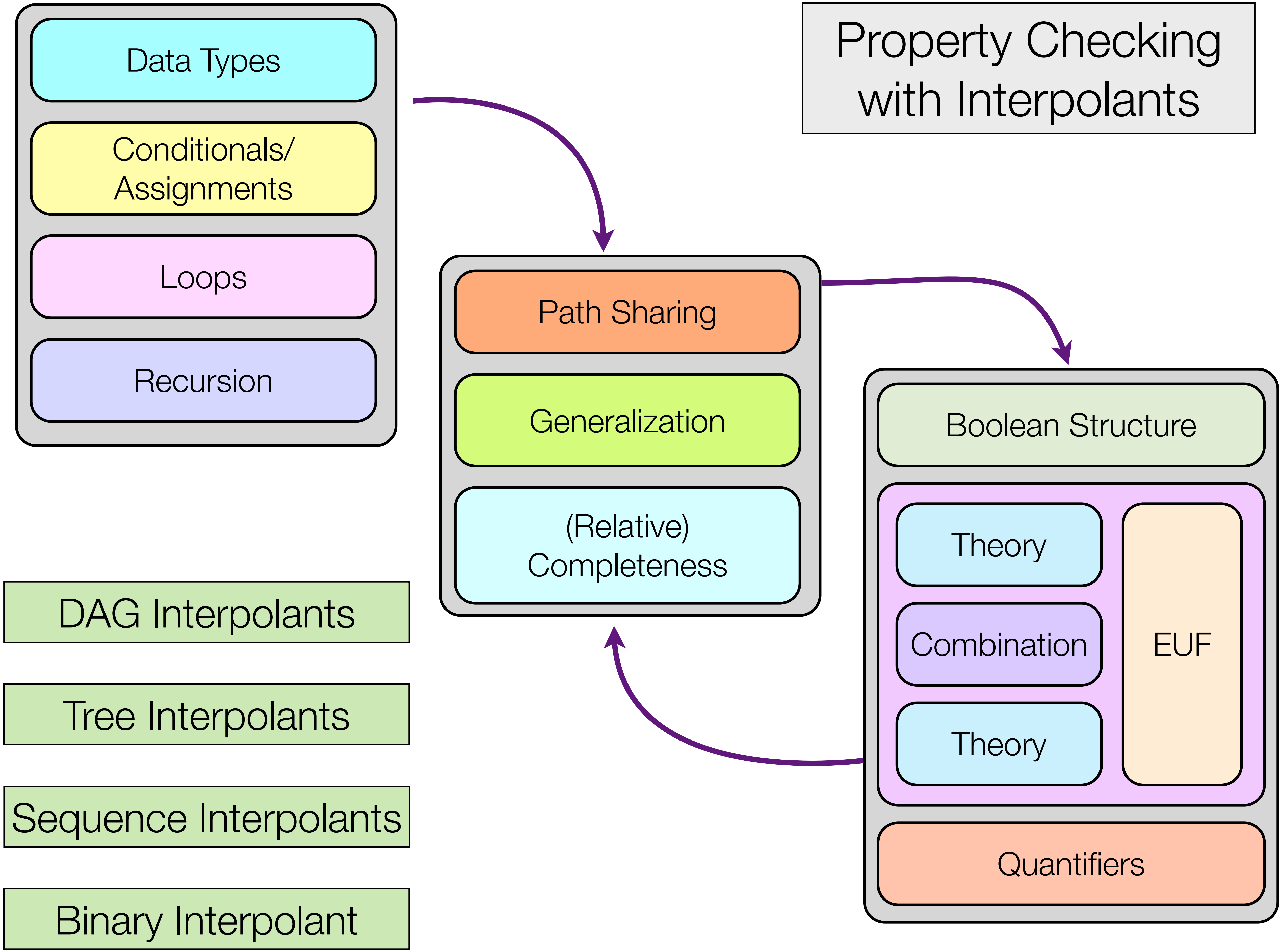


$$\boxed{A = u = x \wedge f(u,y) = z}$$

$$\boxed{B = v = y \wedge f(x,v) \neq z}$$

$$I = f(x,y) = z$$




Property Checking with Interpolants
Data Types
Conditionals/ Assignments
Loops
Recursion
Path Sharing
Generalization
(Relative) Completeness
Boolean Structure
Theory
Combination
Theory
EUF
Quantifiers
DAG Interpolants
Tree Interpolants
Sequence Interpolants
Binary Interpolant

Propositional Interpolants

1995	Huang, Constructing Craig Interpolation Formulas. (OTTER)
1997	Jan Krajíček, Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic.
1997	Pudlák, Lower Bounds for Resolution and Cutting Plane Proofs and Monotone Computations
2003	McMillan, Interpolation and SAT-Based Model Checking.
2006	Yorsh, Musuvathi, A Combination Method for Generating Interpolants.
2009	Biere, Bounded Model Checking (in Handbook of Satisfiability).
2010	D. Kroening, Purandare, Weissenbacher. Interpolant Strength.

Equality Interpolants

1996	Fitting, First-Order Logic and Automated Theorem Proving
2005	McMillan, An Interpolating Theorem Prover
2006	Yorsh, Musuvathi, A Combination Method for Generating Interpolants.
2009	Fuchs, Goel, Grundy, Krstic, Tinelli, Ground Interpolation for the Theory of Equality.
2014	Bonacina, Johansson, Interpolation Systems for Ground Proofs in Automated Reasoning

Interpolation in Theories

2005	McMillan. Interpolating Theorem Prover	LA(Q)
2006	Kapur, Majumdar, Zarba, Interpolation for Data Structures	Datatype theories
2007	Rybalchenko, Sofronie-Stokkermans, Constraint Solving for Interpolation	LA(Q)
2008	Cimatti, Griggio, Sebastiani, Efficient Interpolant Generation in Satisfiability Modulo Theories	LA(Q), DL(Q), UTVPI
2008	Jain, Clarke, Grumberg, Efficient Craig Interpolation for Linear Diophantine (dis)Equations and Linear Modular Equations	LDE, LME
2009	Cimatti, Griggio, Sebastiani, Interpolant Generation for UTVPI	UTVPI
2011	Griggio, Effective Word-Level Interpolation for Software Verification	Bit-Vectors

Interpolation in Theory Combinations

2005	McMillan. Interpolating Theorem Prover	LA(Q) over EUF over Bool
2005	Yorsh and Musuvathi, A Combination Method for Generating Interpolants	Nelson-Oppen
2009	Cimatti, Griggio, Sebastiani, Efficient Generation of Craig Interpolants in Satisfiability Modulo Theories	Delayed Theory Combination
2009	Goel, Krstic, Tinelli, Ground Interpolation for Combined Theories	Proof transformation
2012	Kovacs, Voronkov, Playing in the Gray Area of Proofs	Proof Transformation