TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

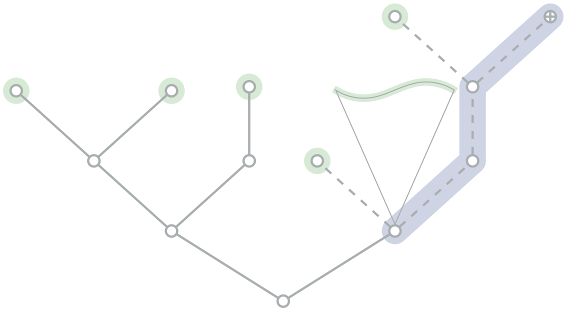
logics Logical Methods in Computer Science

Microsoft

Interpolants from SAT solving certificates

Adrián Rebola-Pardo
Martin Matak
Georg Weissenbacher

TU Wien

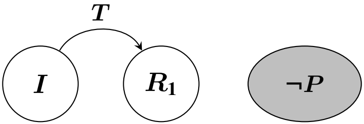


Helmut Veith Workshop
Obertauern, Austria

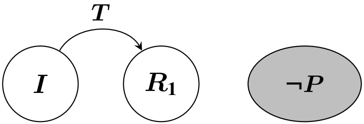
Janary 31, 2018

Supported by FWF W1255-N23 and Microsoft Research through its PhD Programme

Interpolation-based Image Approximation

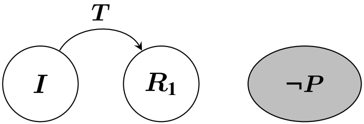


Interpolation-based Image Approximation



$$I(V) \wedge T(V, V') \quad \neg P(V')$$

Interpolation-based Image Approximation

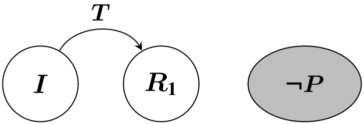


$$I(V) \wedge T(V, V') \quad \neg P(V')$$

- **Image computation amounts to quantifier elimination:**

$$\exists V \,.\, I(V) \wedge T(V, V')$$

Interpolation-based Image Approximation



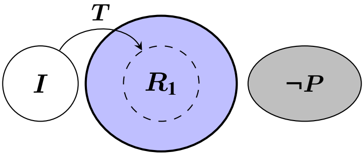
$$I(V) \wedge T(V, V') \quad \neg P(V')$$

- **Image computation amounts to quantifier elimination:**

$$\exists V \, . \, I(V) \wedge T(V, V')$$

- **Can we safely approximate the post-image of T?**

Interpolation-based Image Approximation



$$I(V) \wedge T(V, V') \quad \neg P(V')$$

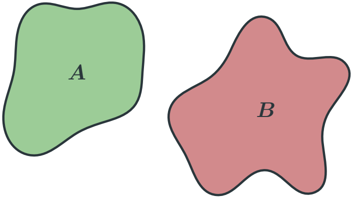
- Image computation amounts to quantifier elimination:

$$\exists V \,.\, I(V) \wedge T(V, V')$$

- Can we safely approximate the post-image of T?

Propositional interpolants

Let A, B be propositional formulae such that $A \wedge B$ is unsatisfiable

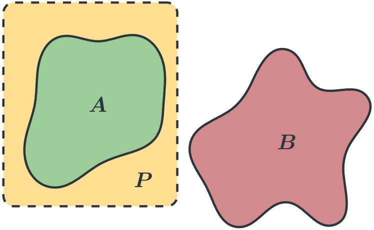


Propositional interpolants

Let A, B be propositional formulae such that $A \wedge B$ is unsatisfiable

Interpolants an (A, B)-interpolant is a propositional formula P such that:

- $A \vDash P$
- $P \wedge B$ is unsatisfiable
- P contains only shared variables between A and B

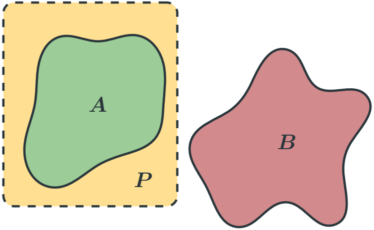


Propositional interpolants

Let A, B be propositional formulae such that $A \wedge B$ is unsatisfiable

Interpolants an (A, B)-interpolant is a propositional formula P such that:

- $A \models P$
- $P \wedge B$ is unsatisfiable
- P contains only shared variables between A and B



Interpolants are essential tools in formal methods and software verification:

- (Un)bounded model checking [McMillan '03]
- Boolean synthesis [Jiang et al. '09]
- Fault localization [Ermis et al. '12]
- Hardware verification [Keng Veneris '09]

Interpolation in practice

Interpolant generation

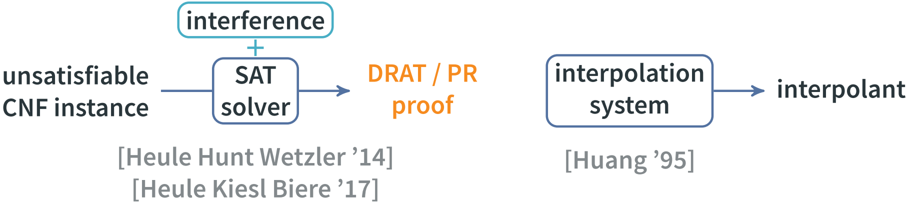
The good old times...

Interpolant generation

The good old times are gone

Interpolant generation

The good old times are gone



Properties of DRAT / PR proofs

- ✔ Shorter and easier to generate or check than resolution proofs
- ✔ Allow to express satisfiability-preserving techniques

Interpolant generation

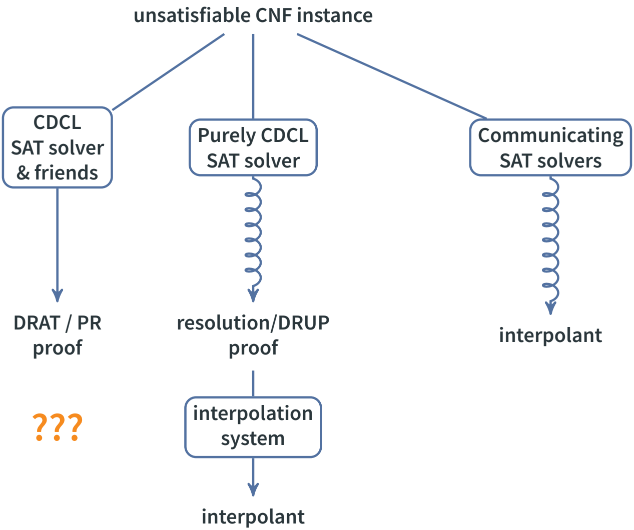
The good old times are gone

Properties of DRAT / PR proofs

- ✔ Shorter and easier to generate or check than resolution proofs
- ✔ Allow to express satisfiability-preserving techniques
- ✘ We do not know how to generate interpolants from DRAT / PR proofs

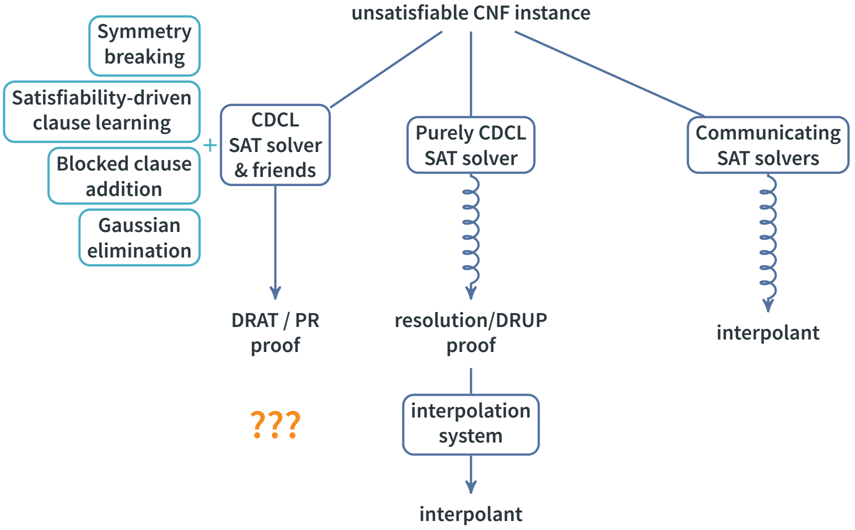
Interpolant generation from proofs

Three approaches



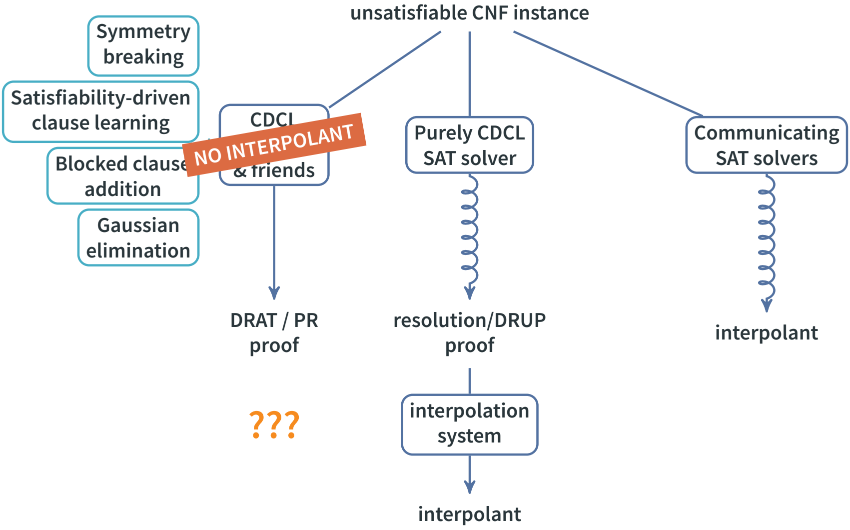
Interpolant generation from proofs

Three approaches



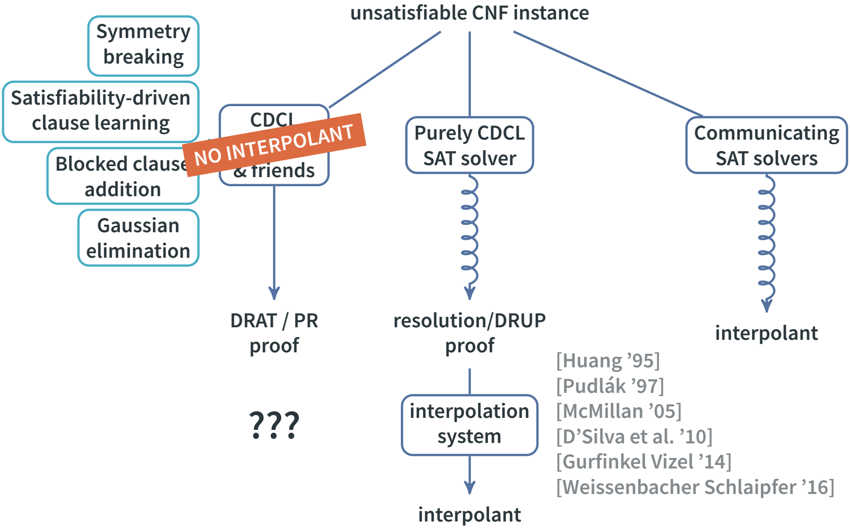
Interpolant generation from proofs

Three approaches



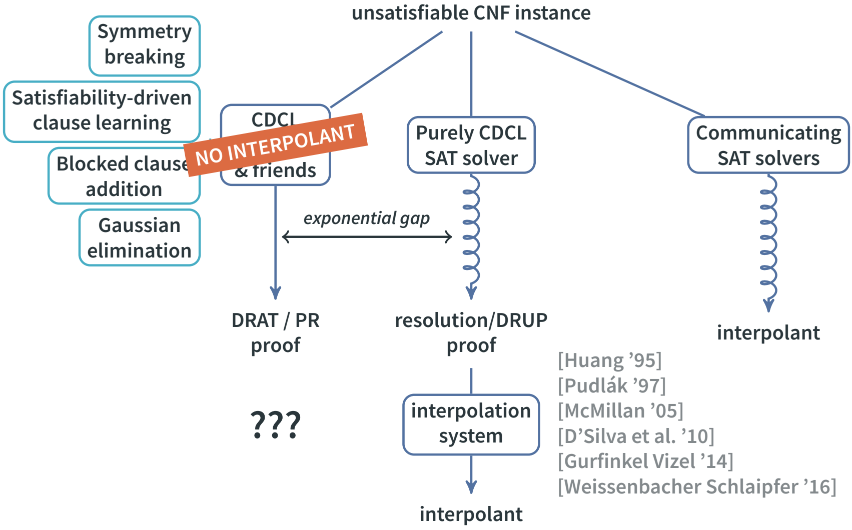
Interpolant generation from proofs

Three approaches



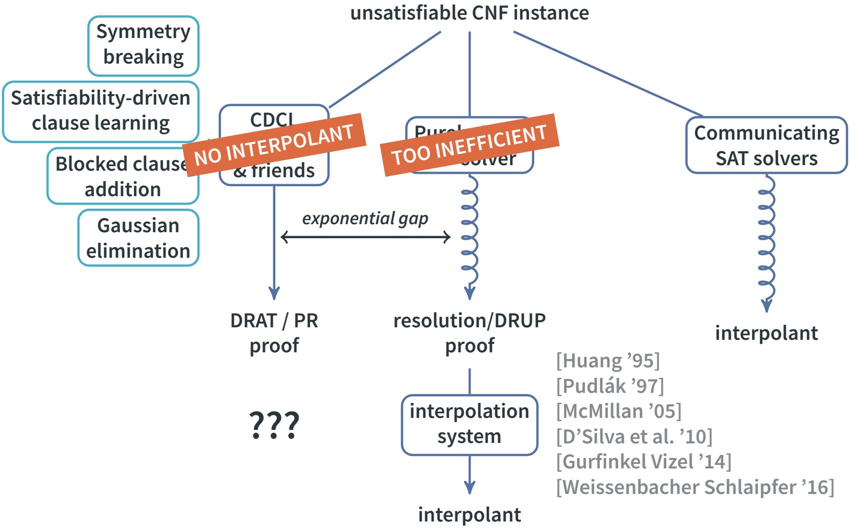
Interpolant generation from proofs

Three approaches



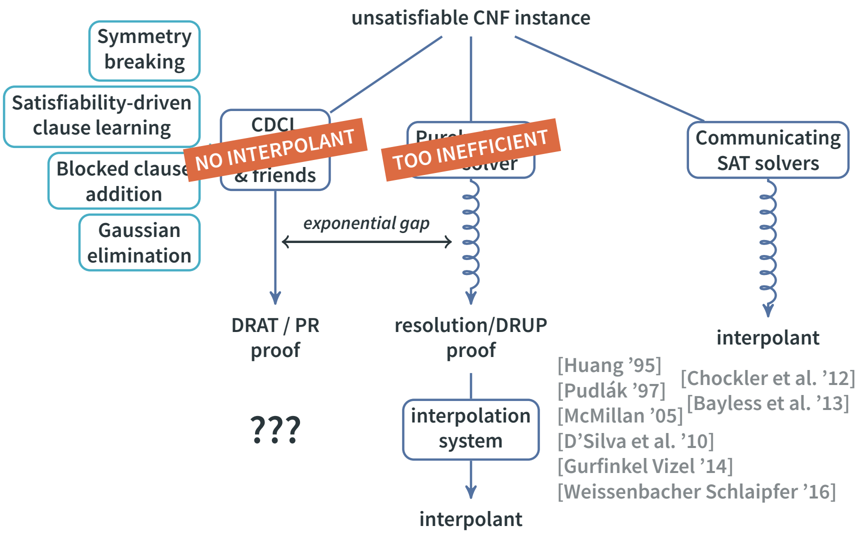
Interpolant generation from proofs

Three approaches



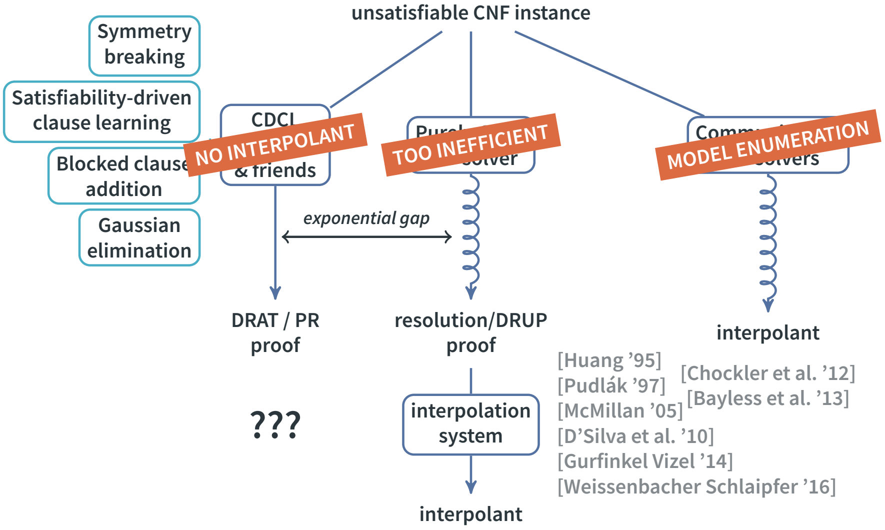
Interpolant generation from proofs

Three approaches



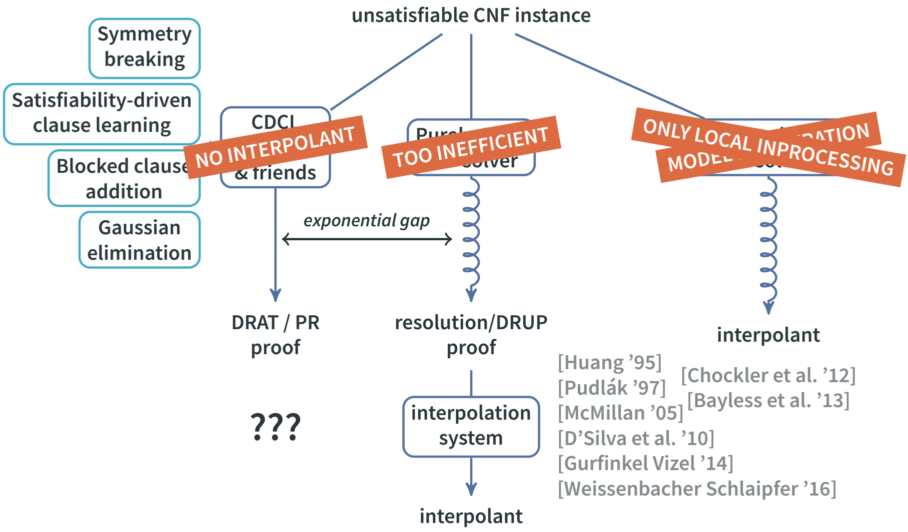
Interpolant generation from proofs

Three approaches



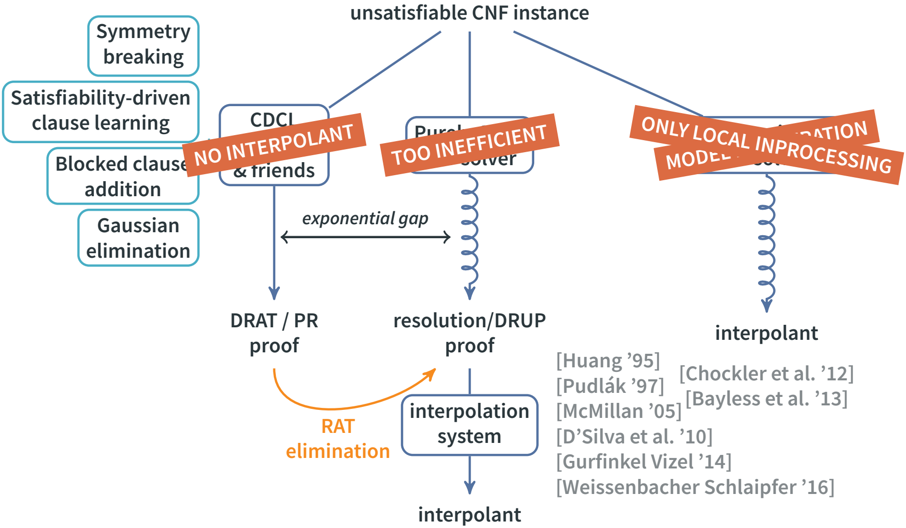
Interpolant generation from proofs

Three approaches



Interpolant generation from proofs

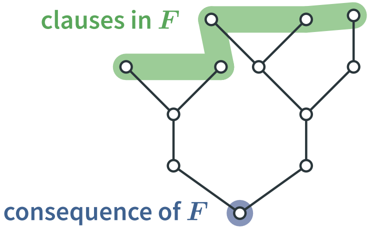
Three approaches



Proof systems for SAT solvers

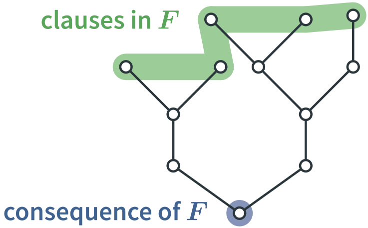
DRUP proofs

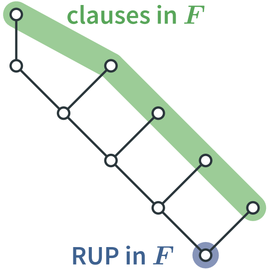
Reverse Unit Propagation (RUP)



DRUP proofs

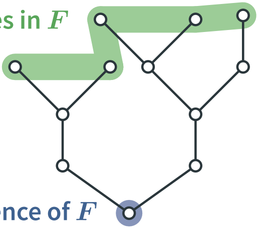
Reverse Unit Propagation (RUP)

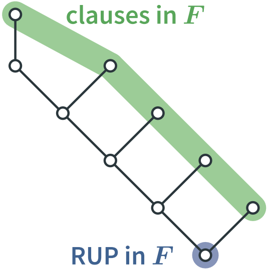




DRUP proofs

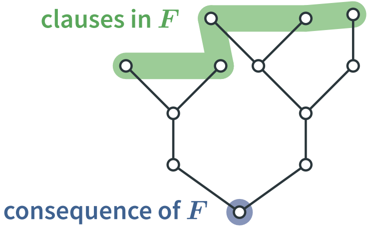
Reverse Unit Propagation (RUP)

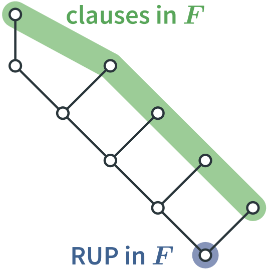




DRUP proof system RUP introduction + arbitrary clause deletion

Reverse Unit Propagation (RUP)





DRUP proof system RUP introduction + arbitrary clause deletion

- Essentially as powerful as resolution [Beame et al. ’04]
- Interpolants can be easily generated [Gurfinkel Vizel ’14]

Resolution asymmetric tautologies

A clause C is a resolution asymmetric tautology (RAT) in a CNF formula F upon a literal l if every resolvent $C \otimes D$ upon l, where $D \in F$, is a RUP in F

Resolution asymmetric tautologies

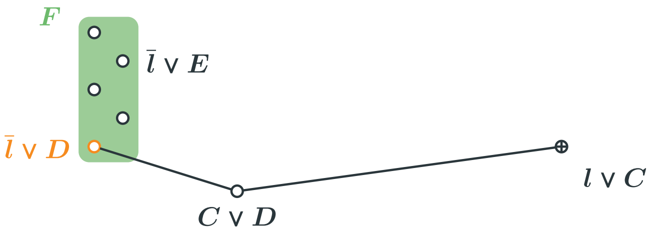
A clause C is a resolution asymmetric tautology (RAT) in a CNF formula F upon a literal l if every resolvent $C \otimes D$ upon l, where $D \in F$, is a RUP in F

Resolution asymmetric tautologies

A clause C is a resolution asymmetric tautology (RAT) in a CNF formula F upon a literal l if every resolvent $C \otimes D$ upon l, where $D \in F$, is a RUP in F

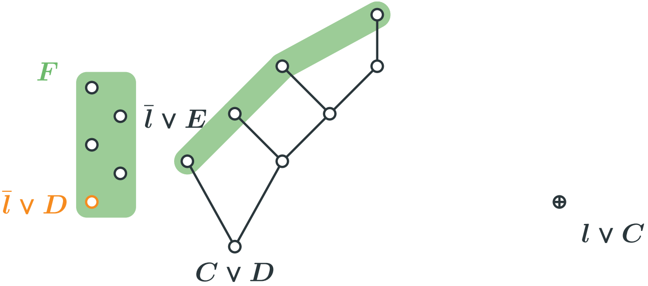
Resolution asymmetric tautologies

A clause C is a resolution asymmetric tautology (RAT) in a CNF formula F upon a literal l if every resolvent $C \otimes D$ upon l, where $D \in F$, is a RUP in F



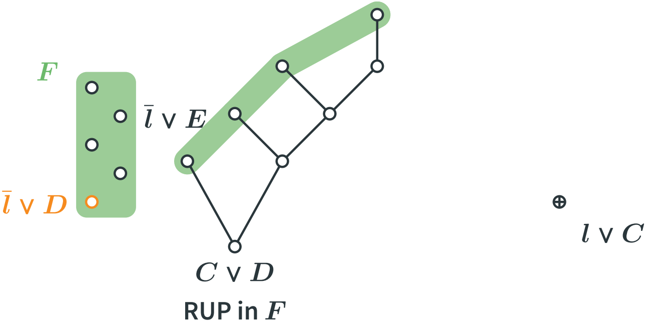
Resolution asymmetric tautologies

A clause C is a resolution asymmetric tautology (RAT) in a CNF formula F upon a literal l if every resolvent $C \otimes D$ upon l, where $D \in F$, is a RUP in F



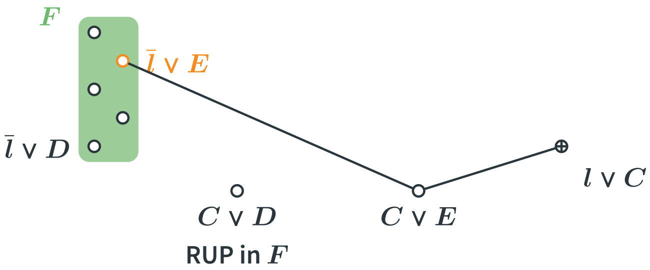
Resolution asymmetric tautologies

A clause C is a resolution asymmetric tautology (RAT) in a CNF formula F upon a literal l if every resolvent $C \otimes D$ upon l, where $D \in F$, is a RUP in F



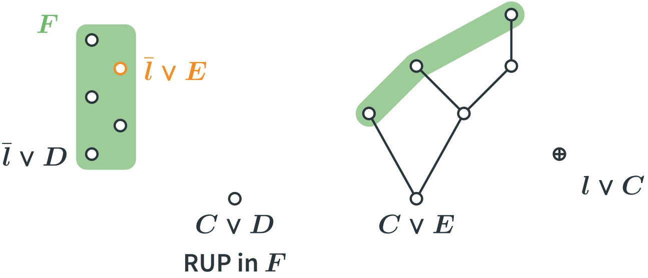
Resolution asymmetric tautologies

A clause C is a resolution asymmetric tautology (RAT) in a CNF formula F upon a literal l if every resolvent $C \otimes D$ upon l, where $D \in F$, is a RUP in F



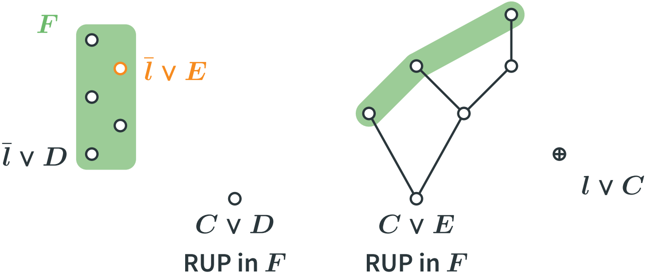
Resolution asymmetric tautologies

A clause C is a resolution asymmetric tautology (RAT) in a CNF formula F upon a literal l if every resolvent $C \otimes D$ upon l, where $D \in F$, is a RUP in F



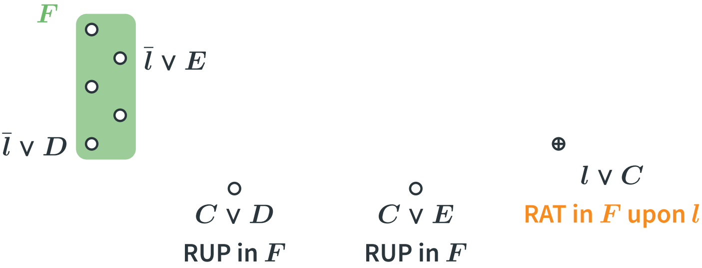
Resolution asymmetric tautologies

A clause C is a resolution asymmetric tautology (RAT) in a CNF formula F upon a literal l if every resolvent $C \otimes D$ upon l, where $D \in F$, is a RUP in F



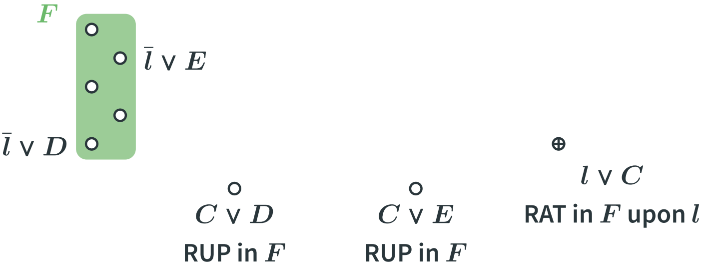
Resolution asymmetric tautologies

A clause C is a resolution asymmetric tautology (RAT) in a CNF formula F upon a literal l if every resolvent $C \otimes D$ upon l, where $D \in F$, is a RUP in F



Resolution asymmetric tautologies

A clause C is a resolution asymmetric tautology (RAT) in a CNF formula F upon a literal l if every resolvent $C \otimes D$ upon l, where $D \in F$, is a RUP in F

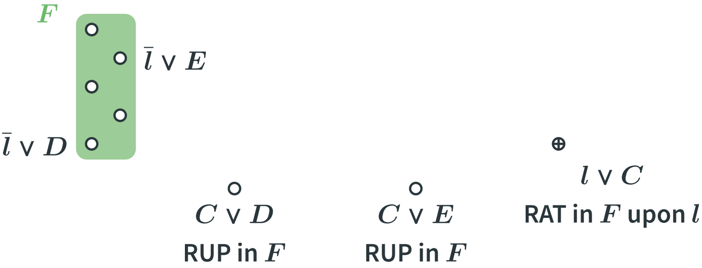


Theorem If C is a RAT in F, then F is satisfiable if and only if $F \cup \{C\}$ is

RAT introduction can be used as an inference rule of a proof system

Resolution asymmetric tautologies

A clause C is a resolution asymmetric tautology (RAT) in a CNF formula F upon a literal l if every resolvent $C \otimes D$ upon l, where $D \in F$, is a RUP in F



Theorem If C is a RAT in F, then F is satisfiable if and only if $F \cup \{C\}$ is

RAT introduction can be used as an inference rule of a proof system

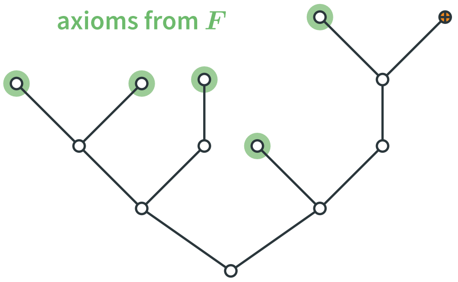
DRAT proof system

RUP introduction + RAT introduction + arbitrary clause deletion

- polynomially simulates extended resolution
  [Heule Kiesl Rebola-Pardo ’18]

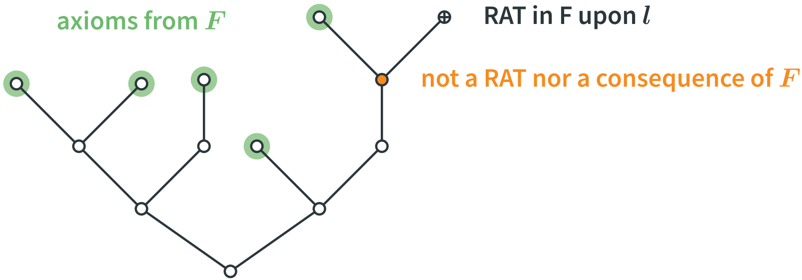
Interpolation from DRAT proofs

RATs, consequences and latency



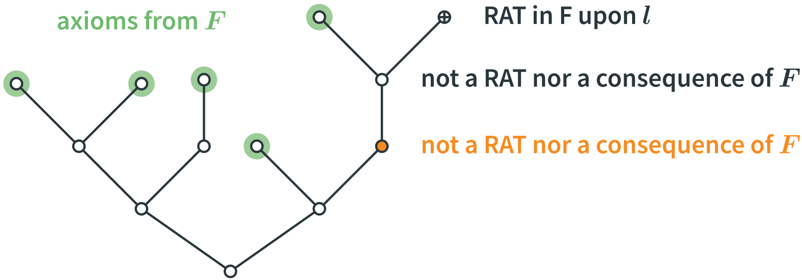
Why does RAT work? Eventually, some successor of every RAT becomes a consequence

RATs, consequences and latency



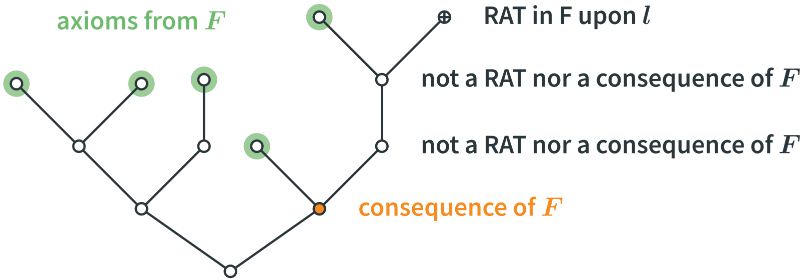
Why does RAT work? Eventually, some successor of every RAT becomes a consequence

RATs, consequences and latency



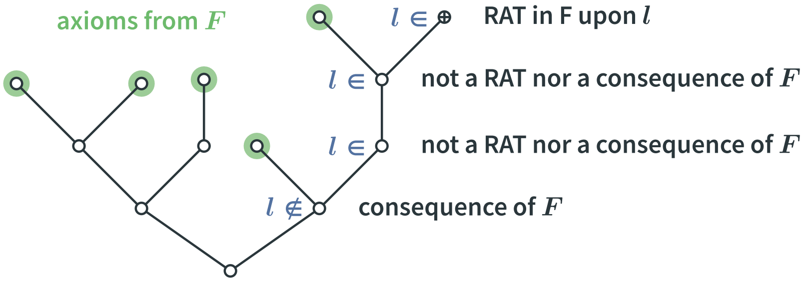
Why does RAT work? Eventually, some successor of every RAT becomes a consequence

RATs, consequences and latency



Why does RAT work? Eventually, some successor of every RAT becomes a consequence

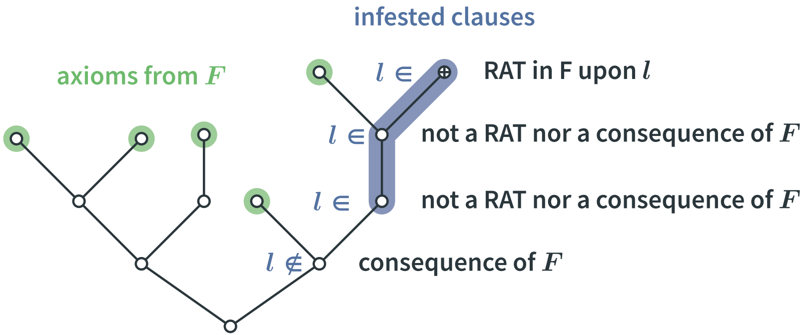
RATs, consequences and latency



Why does RAT work? Eventually, some successor of every RAT becomes a consequence

But *when*? As soon as the pivot literal is eliminated by resolution

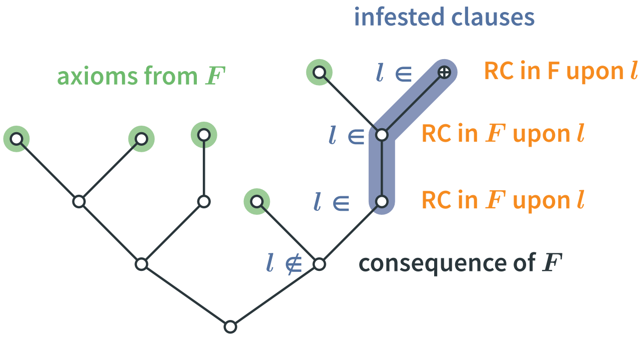
RATs, consequences and latency



Why does RAT work? Eventually, some successor of every RAT becomes a consequence

But *when*? As soon as the pivot literal is eliminated by resolution

RATs, consequences and latency

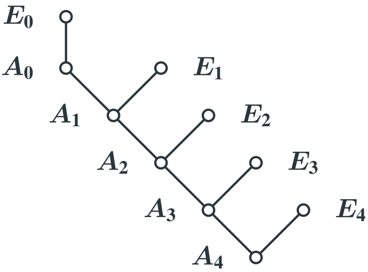


Why does RAT work? Eventually, some successor of every RAT becomes a consequence

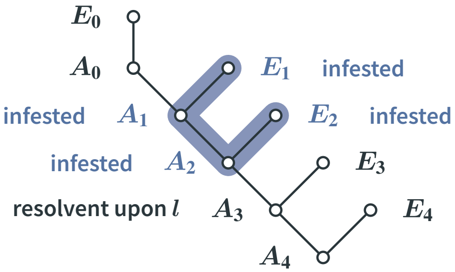
But *when*? As soon as the pivot literal is eliminated by resolution

Question Can we obtain a resolution proof of that consequence clause?

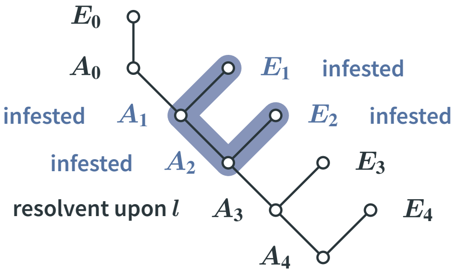
Infested clause elimination



Infested clause elimination



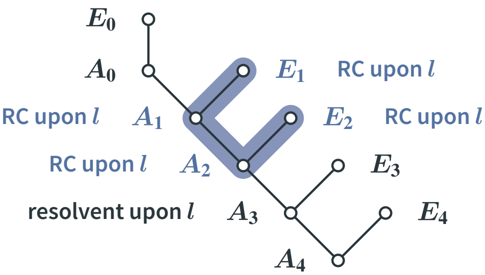
Infested clause elimination



Resolution consequence every resolvent upon l is a consequence of F

all infested clauses are RCs upon l

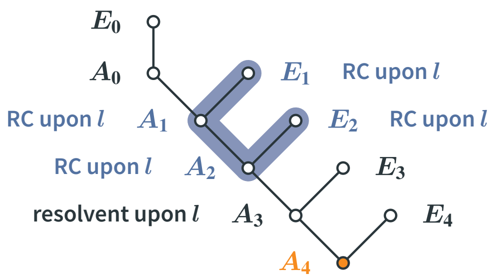
Infested clause elimination



Resolution consequence every resolvent upon l is a consequence of F

all infested clauses are RCs upon l

Infested clause elimination

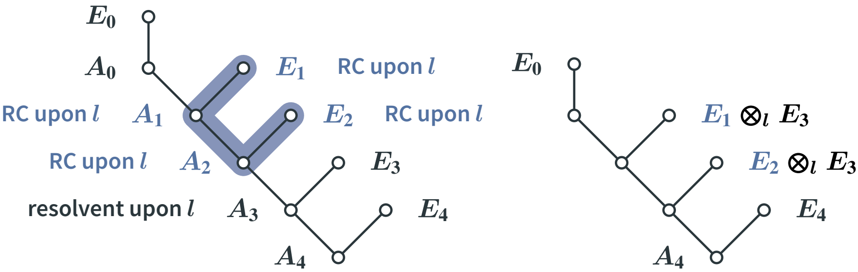


Resolution consequence every resolvent upon l is a consequence of F

all infested clauses are RCs upon l

Goal derive A_4 without using infested clauses

Infested clause elimination

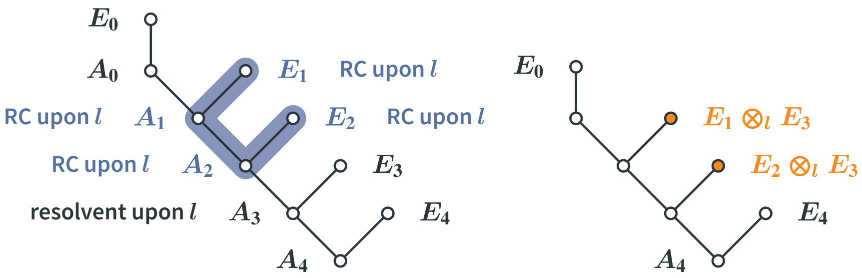


Resolution consequence every resolvent upon l is a consequence of F

all infested clauses are RCs upon l

Goal derive A_4 without using infested clauses

Infested clause elimination



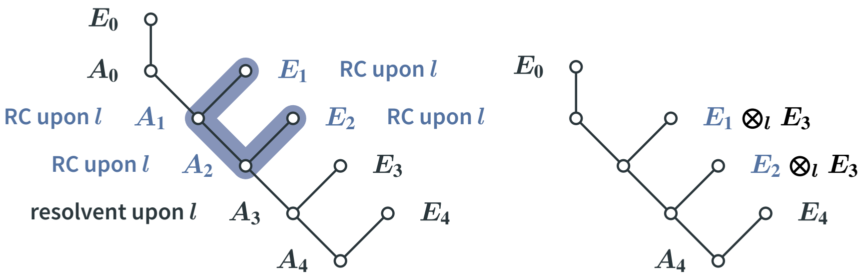
Resolution consequence every resolvent upon l is a consequence of F

all infested clauses are RCs upon l

Goal derive A_4 without using infested clauses

$\Rightarrow$ derive $E_1 \otimes_l E_3$ and $E_2 \otimes_l E_3$ without using infested clauses

Infested clause elimination



Resolution consequence every resolvent upon l is a consequence of F

all infested clauses are RCs upon l

Goal derive A_4 without using infested clauses

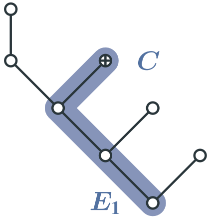
⇒ derive $E_1 \otimes_l E_3$ and $E_2 \otimes_l E_3$ without using infested clauses

Elimination repeat until only clauses $C \otimes_l D$ need to be derived

C is the original RAT upon l in F

D is a clause in F

Infested clause elimination



Resolution consequence every resolvent upon l is a consequence of F

all infested clauses are RCs upon l

Goal derive A_4 without using infested clauses

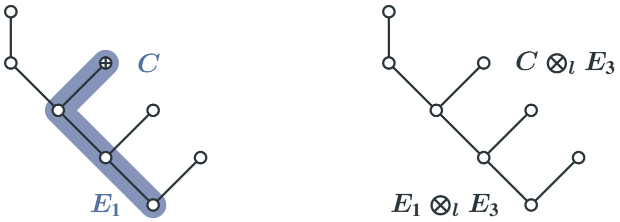
$\Rightarrow$ derive $E_1 \otimes_l E_3$ and $E_2 \otimes_l E_3$ without using infested clauses

Elimination repeat until only clauses $C \otimes_l D$ need to be derived

C is the original RAT upon l in F

D is a clause in F

Infested clause elimination



Resolution consequence every resolvent upon l is a consequence of F

all infested clauses are RCs upon l

Goal derive A_4 without using infested clauses

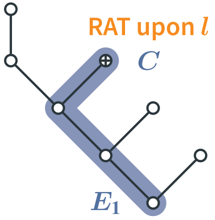
$\Rightarrow$ derive $E_1 \otimes_l E_3$ and $E_2 \otimes_l E_3$ without using infested clauses

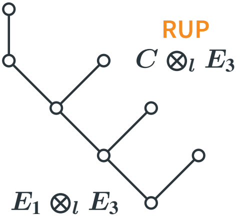
Elimination repeat until only clauses $C \otimes_l D$ need to be derived

C is the original RAT upon l in F

D is a clause in F

Infested clause elimination





Resolution consequence every resolvent upon l is a consequence of F

all infested clauses are RCs upon l

Goal derive A_4 without using infested clauses

$\Rightarrow$ derive $E_1 \otimes_l E_3$ and $E_2 \otimes_l E_3$ without using infested clauses

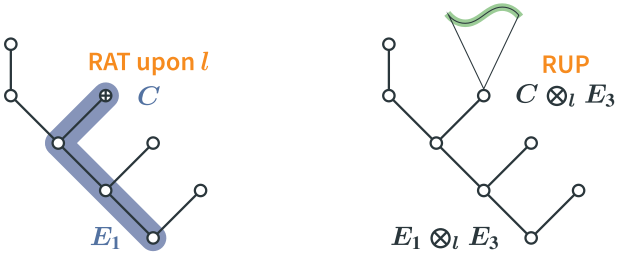
Elimination repeat until only clauses $C \otimes_l D$ need to be derived

C is the original RAT upon l in F

D is a clause in F

$\Rightarrow$ $C \otimes_l D$ is a RUP

Infested clause elimination



Resolution consequence every resolvent upon l is a consequence of F

all infested clauses are RCs upon l

Goal derive A_4 without using infested clauses

$\Rightarrow$ derive $E_1 \otimes_l E_3$ and $E_2 \otimes_l E_3$ without using infested clauses

Elimination repeat until only clauses $C \otimes_l D$ need to be derived

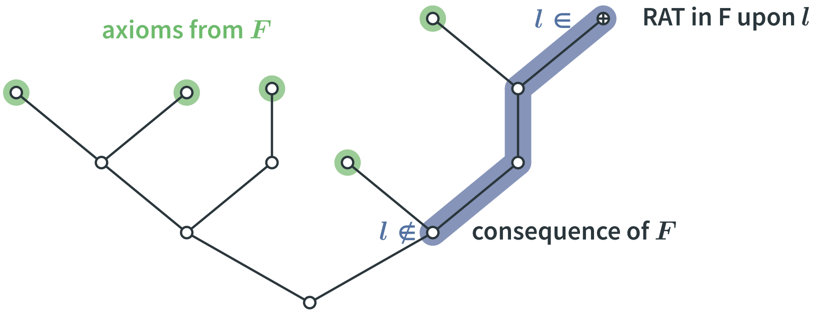
C is the original RAT upon l in F

D is a clause in F

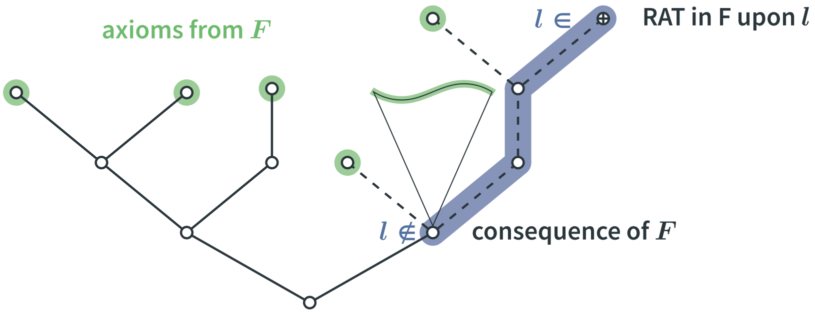
$\Rightarrow$ $C \otimes_l D$ is a RUP

Conclusion

Interpolation through RAT elimination



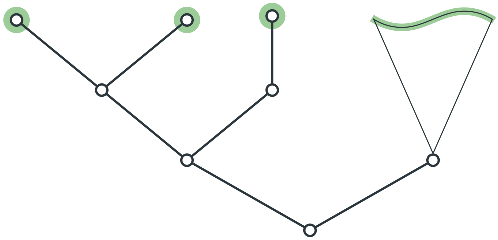
Interpolation through RAT elimination



Interpolant generation from DRAT proofs

Interpolation through RAT elimination

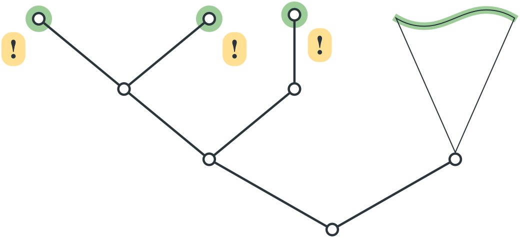
axioms from F



Interpolant generation from DRAT proofs

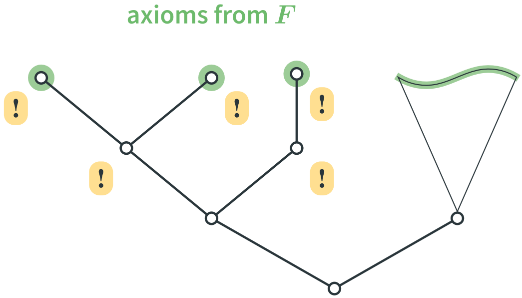
Interpolation through RAT elimination

axioms from F

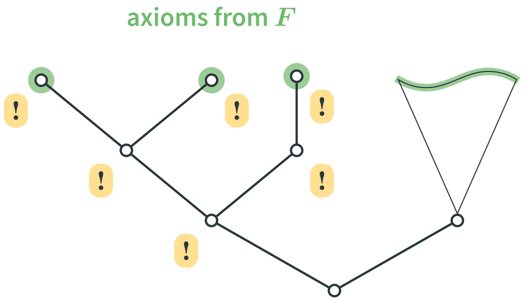


Interpolant generation from DRAT proofs

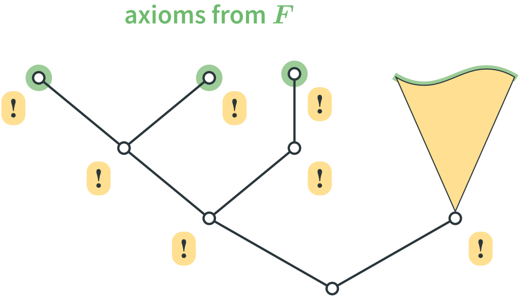
Interpolation through RAT elimination



Interpolation through RAT elimination

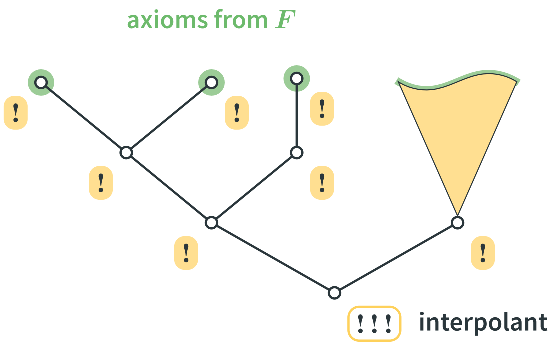


Interpolation through RAT elimination



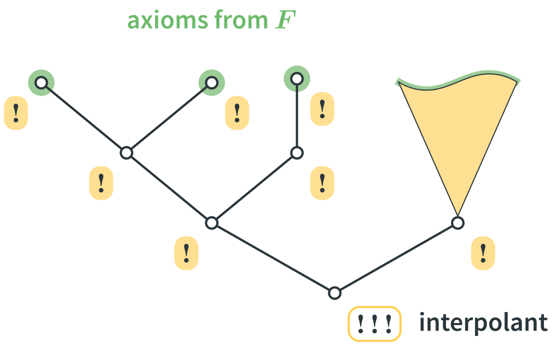
Interpolant generation from DRAT proofs

Interpolation through RAT elimination



Interpolant generation from DRAT proofs

Interpolation through RAT elimination



Issues

- The interpolant may be exponential with respect to the DRAT proof
  ... but DRAT proofs can be exponentially shorter than DRUP proofs
- Currently we only eliminate RATs one by one
  Open question: can PR clauses be exploited to overcome this?
- Prototype by Martin Matak; Implementation by Adrián Rebola-Pardo (evaluation pending)