Informatics

University of Southampton

Constraint Learning for Non-confluent Proof Search

Clemens Eisenhofer (TU Wien)

Joint work with

Michael Rawson (U. Southampton) & *Laura Kovács* (TU Wien)

(First-Order) Connection Calculus (CC)

(First-Order) Connection Calculus (CC)

- "First calculus towards ATP" [Bibel; Loveland – late 70s]

(First-Order) Connection Calculus (CC)

- "First calculus towards ATP" [Bibel; Loveland – late 70s]
- Variant of (**first-order) tableaux** and **binary resolution**

(First-Order) Connection Calculus (CC)

- "First calculus towards ATP" [Bibel; Loveland – late 70s]
- Variant of (**first-order) tableaux** and **binary resolution**
- **Sound** & **complete**

(First-Order) Connection Calculus (CC)

- "First calculus towards ATP" [Bibel; Loveland – late 70s]
- Variant of (**first-order) tableaux** and **binary resolution**
- **Sound** & **complete**
- **Goal-directed**

(First-Order) Connection Calculus (CC)

- "First calculus towards ATP" [Bibel; Loveland – late 70s]
- Variant of (**first-order) tableaux** and **binary resolution**
- **Sound** & **complete**
- **Goal-directed**
- Unlike "ordinary" tableaux or resolution: **Non-confluent**
  - Some proof steps are wrong and result in "dead ends"

(First-Order) Connection Calculus (CC)

- "First calculus towards ATP" [Bibel; Loveland – late 70s]
- Variant of (**first-order) tableaux** and **binary resolution**
- **Sound** & **complete**
- **Goal-directed**
- Unlike "ordinary" tableaux or resolution: **Non-confluent**
  - Some proof steps are wrong and result in "dead ends"
- Iterative deepening & proof enumeration

(First-Order) Connection Tableaux

- **Given** a set of (first-order) input clauses
- Just **3** kinds of **rules (+ first-order unifier)**

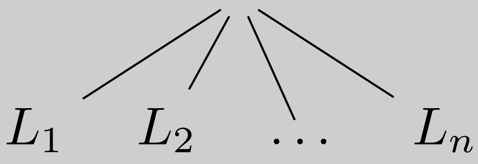
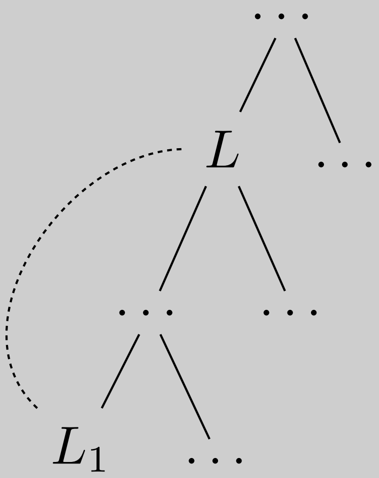
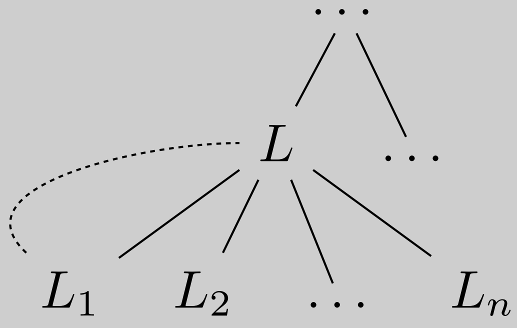
(First-Order) Connection Tableaux

- **Given** a set of (first-order) input clauses
- Just **3** kinds of **rules (+ first-order unifier)**

Start	Reduction	Extension

(First-Order) Connection Tableaux

- **Given** a set of (first-order) input clauses
- Just **3** kinds of **rules (+ first-order unifier)**

<table>
<tr><th>Start</th><th>Reduction</th><th>Extension</th></tr>
<tr>
<td>
</td>
<td>
</td>
<td>
</td>
</tr>
</table>

An Example

$C_1 : \forall z \left(P(z) \lor P\left(f(z)\right) \right)$, $C_2 : \forall x \forall y \left(\neg P(x) \lor \neg P\left(f(y)\right) \right)$

An Example

$C_1\!: \forall z \left(P(z) \vee P\big(f(z)\big)\right)$, $C_2\!: \forall x \forall y \left(\neg P(x) \vee \neg P\big(f(y)\big)\right)$

Start

$$P(z_1) \quad P(f(z_1))$$

$\sigma(z_1) \mapsto z_1$ $\quad\sigma(x_1) \mapsto x_1$ $\quad\sigma(x_2) \mapsto x_2$ $\quad\sigma(y_2) \mapsto y_2$

An Example

$C_1 : \forall z \left(P(z) \vee P\left(f(z)\right)\right)$, $C_2 : \forall x \forall y \left(\neg P(x) \vee \neg P\left(f(y)\right)\right)$

$P(z_1) \quad P(f(z_1))$

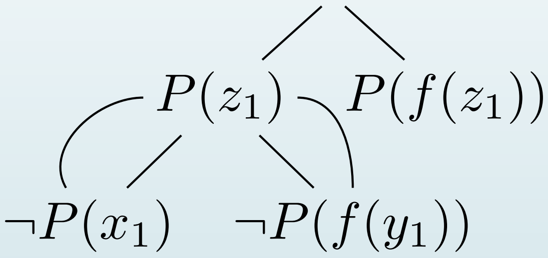
Extension $\quad \neg P(x_1) \quad \neg P(f(y_1))$

$\sigma(z_1) \mapsto z_1 \qquad \sigma(x_1) \mapsto z_1 \qquad \sigma(x_2) \mapsto x_2 \qquad \sigma(y_2) \mapsto y_2$

An Example

$C_1 : \forall z \left(P(z) \vee P\left(f(z)\right) \right)$, $C_2 : \forall x \forall y \left(\neg P(x) \vee \neg P\left(f(y)\right) \right)$

Reduction



$\sigma(z_1) \mapsto f(y_1)$ $\sigma(x_1) \mapsto f(y_1)$ $\sigma(x_2) \mapsto x_2$ $\sigma(y_2) \mapsto y_2$

An Example

$C_1 : \forall z \big(P(z) \vee P\big(f(z)\big)\big)$, $C_2 : \forall x \forall y \big(\neg P(x) \vee \neg P\big(f(y)\big)\big)$

$$P(z_1) \qquad P(f(z_1))$$

Extension

$$\neg P(x_1) \quad \neg P(f(y_1)) \quad \neg P(x_2) \quad \neg P(f(y_2))$$

$\sigma(z_1) \mapsto f(y_1)$ $\sigma(x_1) \mapsto f(y_1)$ $\sigma(x_2) \mapsto f\big(f(y_1)\big)$ $\sigma(y_2) \mapsto y_2$

An Example

$C_1\!: \forall z \left(P(z) \vee P\left(f(z)\right)\right)$, $C_2\!: \forall x \forall y \left(\neg P(x) \vee \neg P\left(f(y)\right)\right)$

Reduction

$$P(z_1) \qquad\qquad P(f(z_1))$$

$$\neg P(x_1) \quad \neg P(f(y_1)) \quad \neg P(x_2) \quad \neg P(f(y_2))$$

$\sigma(z_1) \mapsto f(y_1)$ $\quad$ $\sigma(x_1) \mapsto f(y_1)$ $\quad$ $\sigma(x_2) \mapsto f\left(f(y_1)\right)$ $\quad$ $\sigma(y_2) \mapsto f(y_1)$

An Example

$C_1: \forall z \left(P(z) \vee P(f(z))\right)$, $C_2: \forall x \forall y \left(\neg P(x) \vee \neg P(f(y))\right)$

Reduction

$$P(z_1) \qquad P(f(z_1))$$

$$\neg P(x_1) \quad \neg P(f(y_1)) \quad \neg P(x_2) \quad \neg P(f(y_2))$$

$\sigma(z_1) \mapsto f(y_1)$ $\sigma(x_1) \mapsto f(y_1)$ $\sigma(x_2) \mapsto f(f(y_1))$ $\sigma(y_2) \mapsto f(y_1)$

Conflict learning – A Brief Introduction

Conflict learning – A Brief Introduction

$$(\neg A \lor \neg B \lor C) \land (\neg C \lor D) \land (\neg A \lor \neg B \lor \neg D) \land (A \lor B)$$

Conflict learning – A Brief Introduction

$$(\neg A \lor \neg B \lor C) \land (\neg C \lor D) \land (\neg A \lor \neg B \lor \neg D) \land \ (\mathrm{A} \ \lor \mathrm{B})$$

- 1st „Guess“ $A \ \mapsto \ \top$

$$(\neg A \lor \neg B \lor C) \land (\neg C \lor D) \land (\neg A \lor \neg B \lor \neg D) \land \ (\mathrm{A} \ \lor \mathrm{B})$$

Conflict learning – A Brief Introduction

$$(\neg A \lor \neg B \lor C) \land (\neg C \lor D) \land (\neg A \lor \neg B \lor \neg D) \land \ (\mathrm{A} \ \lor \mathrm{B})$$

- 1st „Guess“ $A \ \mapsto \ \mathrm{T}$

$$(\neg A \lor \neg B \lor C) \land (\neg C \lor D) \land (\neg A \lor \neg B \lor \neg D) \land \ (\mathrm{A} \ \lor \mathrm{B})$$

- 2nd „Guess“ $\mathrm{B} \ \mapsto \ \mathrm{T}$

$$(\neg A \lor \neg B \lor C) \land (\neg C \lor D) \land (\neg A \lor \neg B \lor \neg D) \land \ (\mathrm{A} \ \lor \mathrm{B})$$

Conflict learning – A Brief Introduction

$$(\neg A \lor \neg B \lor C) \land (\neg C \lor D) \land (\neg A \lor \neg B \lor \neg D) \land \ (\mathrm{A} \ \lor \mathrm{B})$$

- 1st „Guess“ $A \ \mapsto \ \mathrm{T}$

$$(\neg A \lor \neg B \lor C) \land (\neg C \lor D) \land (\neg A \lor \neg B \lor \neg D) \land \ (\mathrm{A} \ \lor \mathrm{B})$$

- 2nd „Guess“ $\mathrm{B} \ \mapsto \ \mathrm{T}$

$$(\neg A \lor \neg B \lor C) \land (\neg C \lor D) \land (\neg A \lor \neg B \lor \neg D) \land \ (\mathrm{A} \ \lor \mathrm{B})$$

- Propagate $\mathrm{C} \ \mapsto \ \mathrm{T}$

$$(\neg A \lor \neg B \lor C) \land (\neg C \lor D) \land (\neg A \lor \neg B \lor \neg D) \land \ (\mathrm{A} \ \lor \mathrm{B})$$

Conflict learning – A Brief Introduction

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

- 1^{st} „Guess“ $A \mapsto$ T

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

- 2^{nd} „Guess“ $\mathrm{B} \mapsto$ T

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

- Propagate $\mathrm{C} \mapsto$ T

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

- Propagate $\mathrm{D} \mapsto$ T

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

Conflict learning – A Brief Introduction

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

- 1st „Guess“ $A \mapsto \top$

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

- 2nd „Guess“ $\mathrm{B} \mapsto \top$

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

- Propagate $\mathrm{C} \mapsto \top$

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

- Propagate $\mathrm{D} \mapsto \top$

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

- We backjump and **learn** the clause: $\neg A \vee \neg B$

Conflict learning – A Brief Introduction

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

- 1st „Guess" $A \mapsto \top$

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

- 2nd „Guess" $\mathrm{B} \mapsto \top$

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

- Propagate $\mathrm{C} \mapsto \top$

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

- Propagate $\mathrm{D} \mapsto \top$

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B})$$

- We backjump and **learn** the clause: $\neg A \vee \neg B$
- We resume with

$$(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D) \wedge (\neg A \vee \neg B \vee \neg D) \wedge (\mathrm{A} \vee \mathrm{B}) \wedge (\neg A \vee \neg B)$$

Summary So Far

Summary So Far

- Theoretical argument for using conflict learning in CC search

Summary So Far

- Theoretical argument for using conflict learning in CC search
- Overhead of using SAT core

Summary So Far

- Theoretical argument for using conflict learning in CC search
- Overhead of using SAT core
- SAT/SMT solvers are overwhelmed by massive constraints

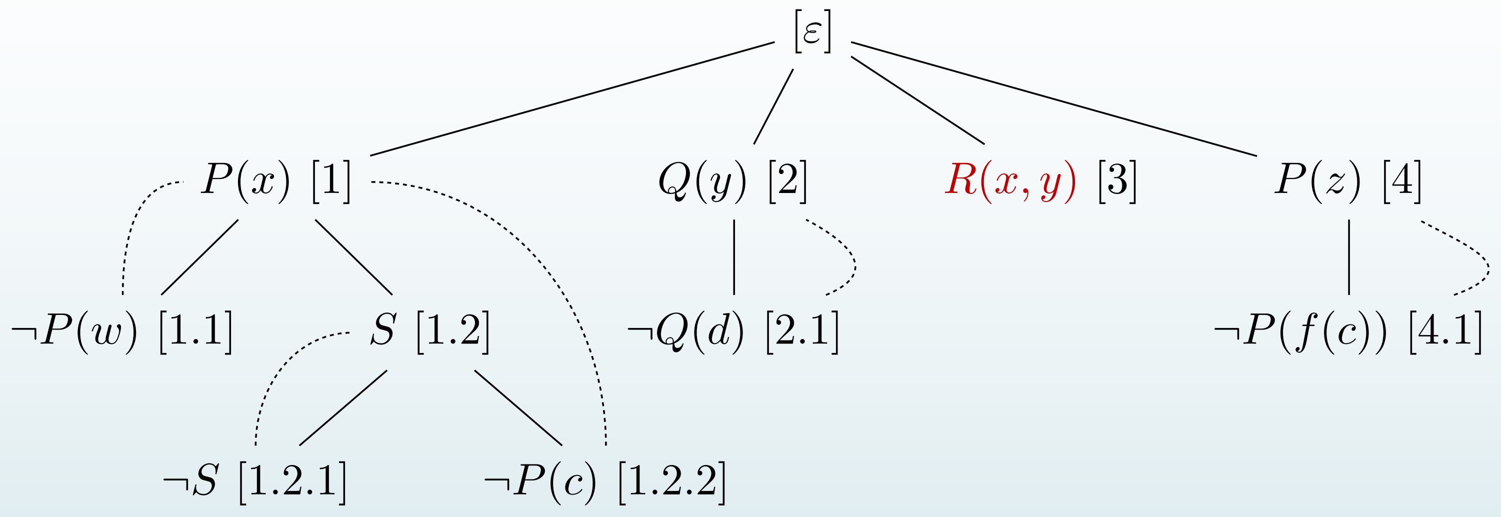
Summary So Far

- Theoretical argument for using conflict learning in CC search
- Overhead of using SAT core
- SAT/SMT solvers are overwhelmed by massive constraints

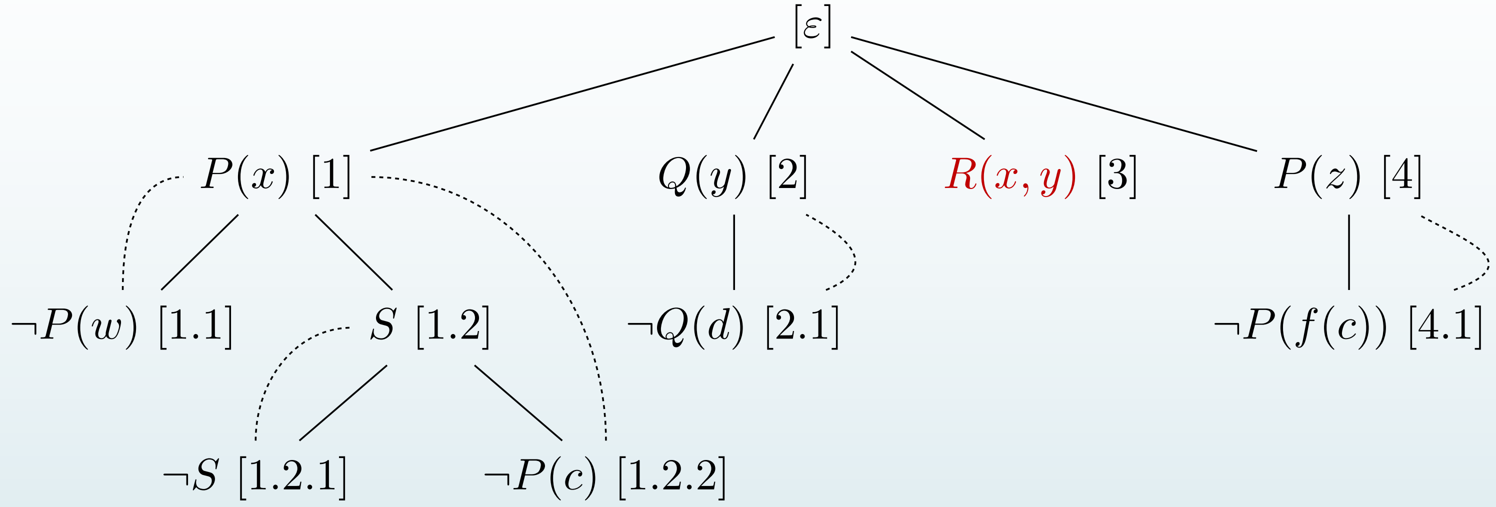
➔ **Specialized Learning Engine**

Custom Learning Engine for CC

Encoding Connection Tableaux

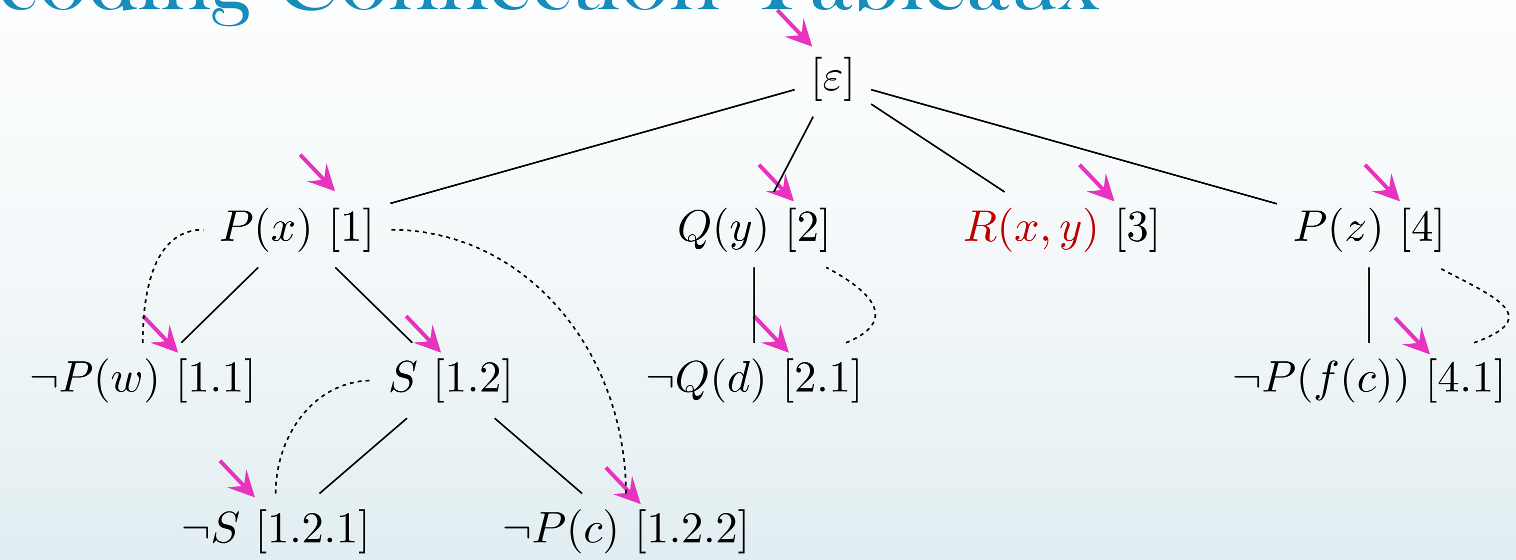


Encoding Connection Tableaux



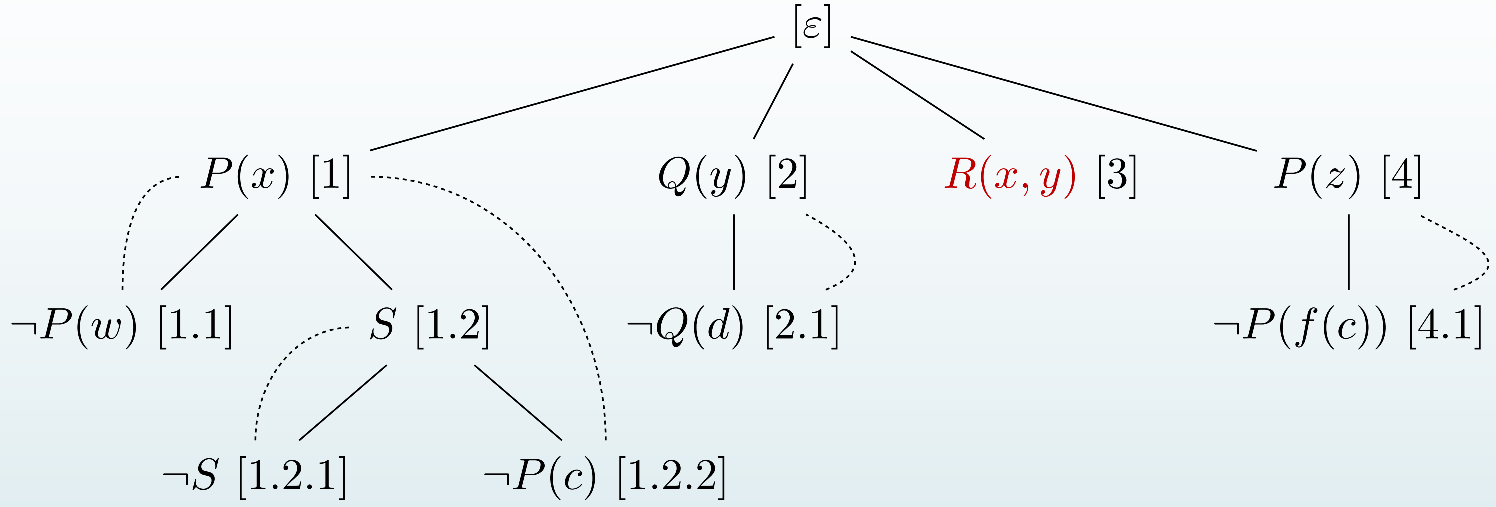
- We maintain explicit positions

Encoding Connection Tableaux



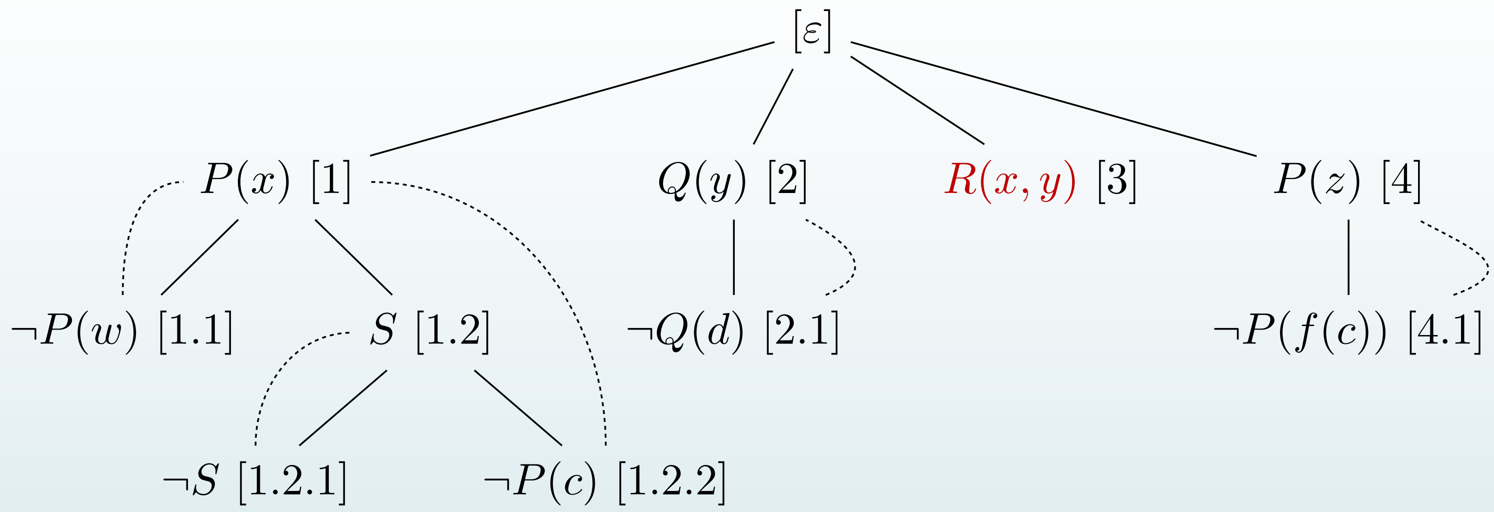
- We maintain explicit positions

Encoding Connection Tableaux



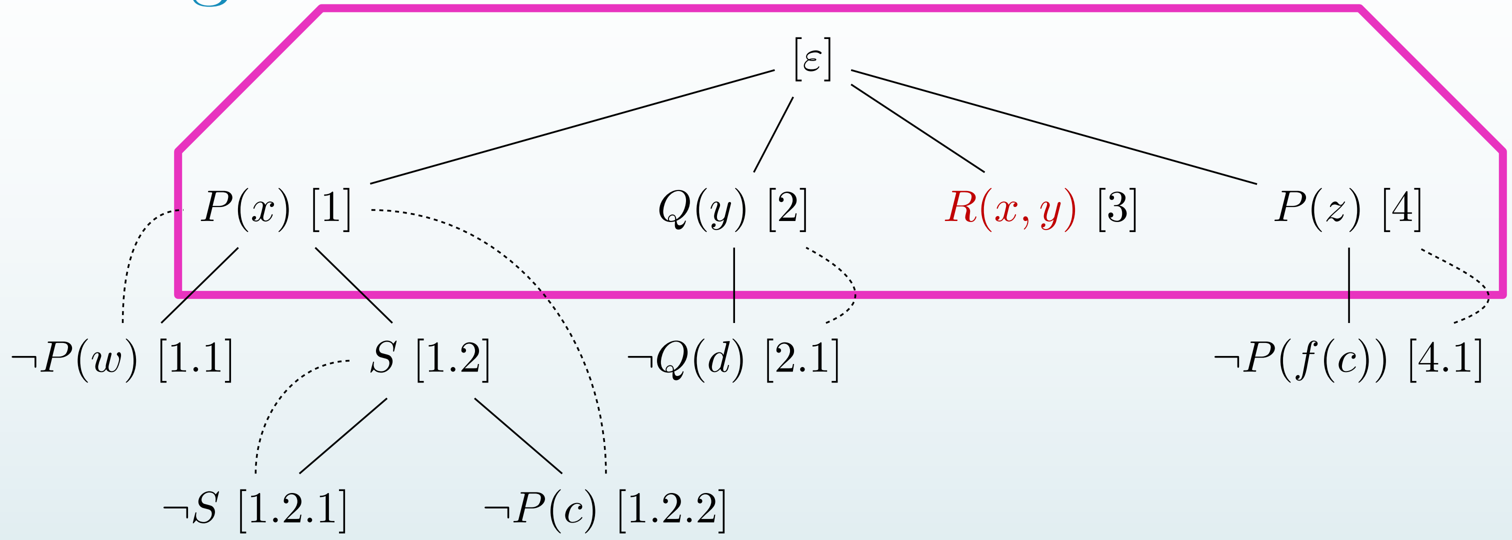
- We maintain explicit positions

Encoding Connection Tableaux



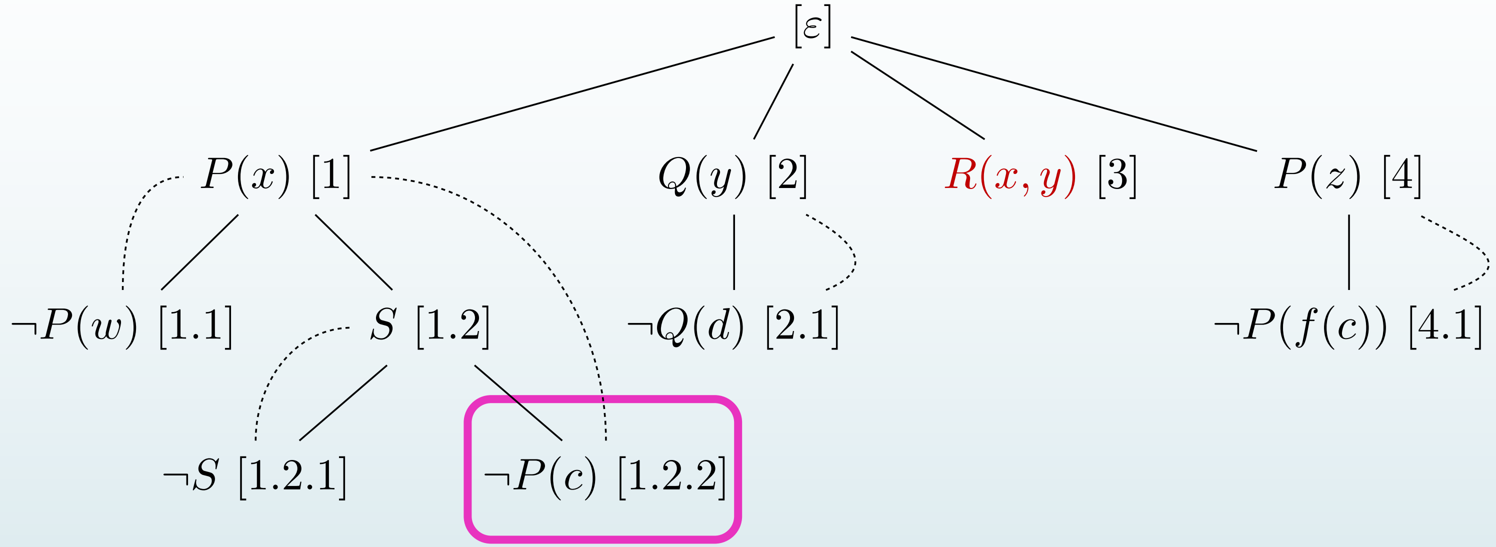
- We maintain explicit positions
- We have **three kinds** of literals

Encoding Connection Tableaux



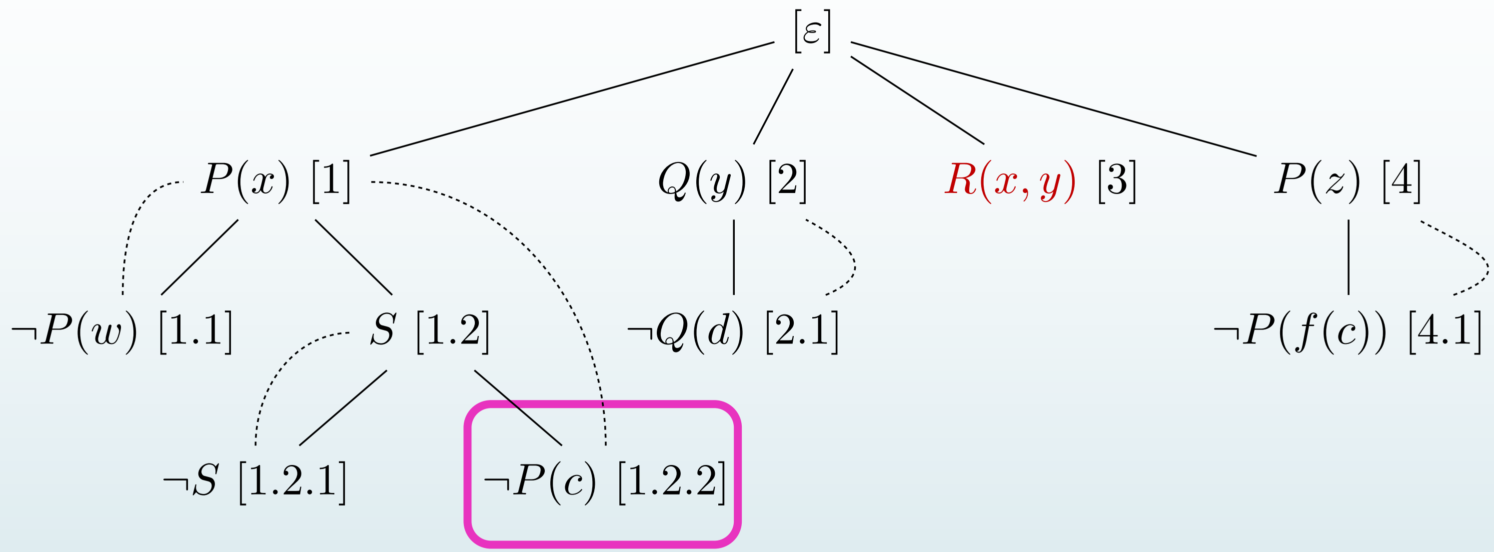
- We maintain explicit positions
- We have **three kinds** of literals
  1. Choose **Start** Clause – e.g., $S_{P(x) \vee Q(y) \vee R(x,y) \vee P(z)}$

Encoding Connection Tableaux



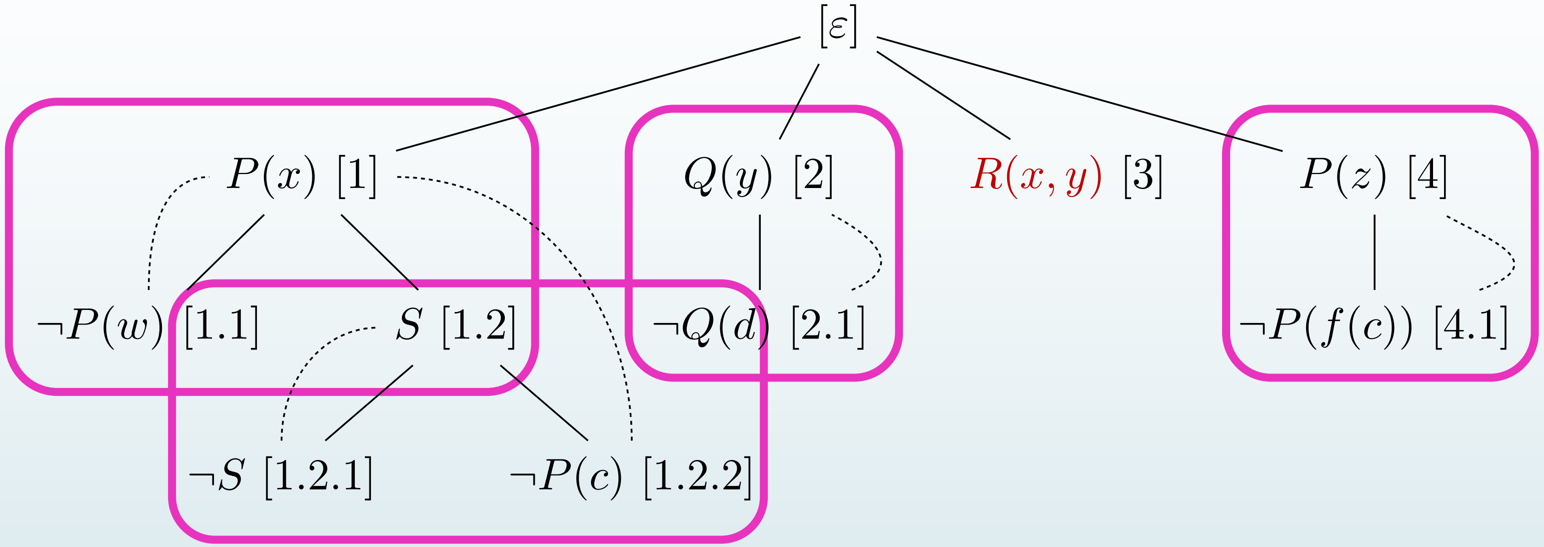
- We maintain explicit positions
- We have **three kinds** of literals
  1. Choose **Start** Clause – e.g., $S_{P(x)\vee Q(y)\vee R(x,y)\vee P(z)}$
  2. Apply **Reduction** – e.g., $R^1_{1.2.2}$

Encoding Connection Tableaux



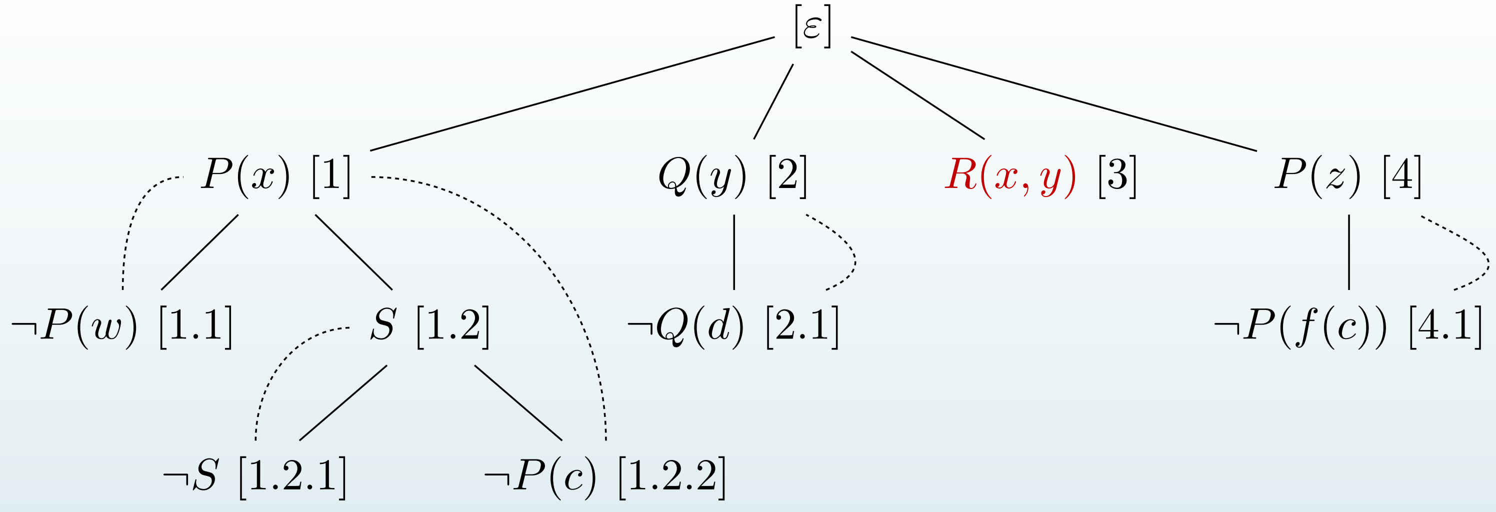
- We maintain explicit positions
- We have **three kinds** of literals
  1. Choose **Start** Clause – e.g., $S_{P(x) \vee Q(y) \vee R(x,y) \vee P(z)}$
  2. Apply **Reduction** – e.g., $R^1_{1.2.2}$
  3. Apply **Extension** – e.g., $E^1_{\neg P(w) \vee S/1}$

Encoding Connection Tableaux



- We maintain explicit positions
- We have **three kinds** of literals
  1. Choose **Start** Clause – e.g., $S_{\mathrm{P(x)\vee Q(y)\vee R(x,y)\vee P(z)}}$
  2. Apply **Reduction** – e.g., $R^1_{1.2.2}$
  3. Apply **Extension** – e.g., $E^1_{\neg P(w)\vee S/1}$
- Position i is **open** iff we have neither R^j_i nor $E^i_{C/k}$

Encoding Connection Tableaux



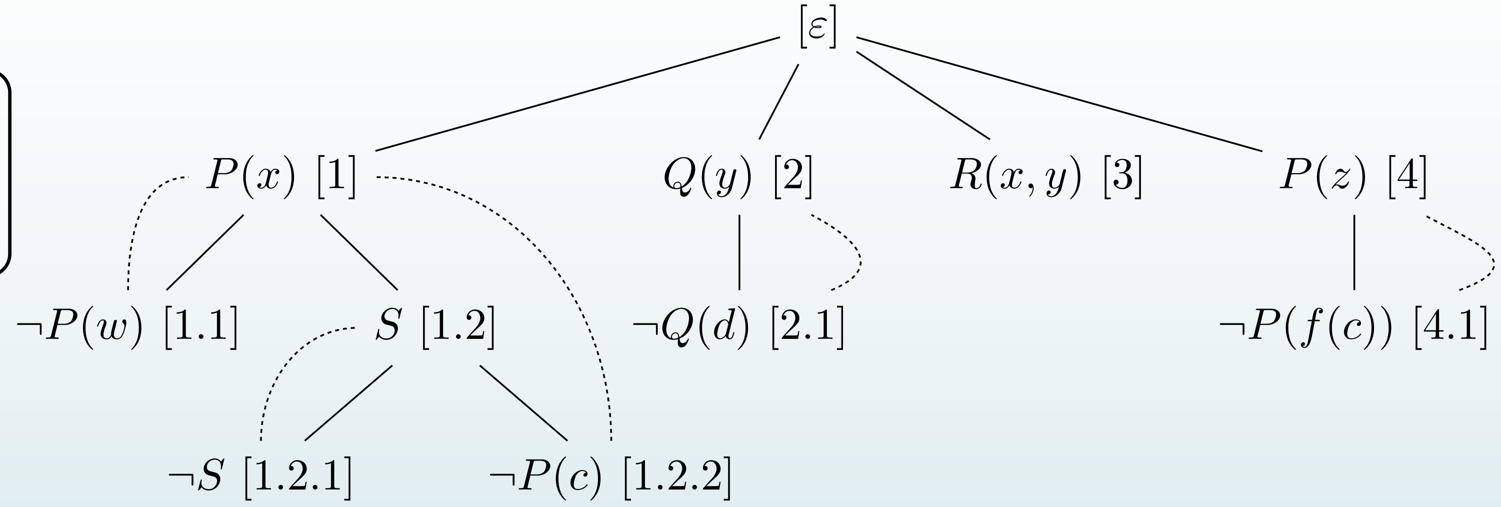
- We maintain explicit positions
- We have **three kinds** of literals
  1. Choose **Start** Clause – e.g., $S_{P(x) \vee Q(y) \vee R(x,y) \vee P(z)}$
  2. Apply **Reduction** – e.g., $R^1_{1.2.2}$
  3. Apply **Extension** – e.g., $E^1_{\neg P(w) \vee S/1}$
- Position i is **open** iff we have neither R^j_i nor $E^i_{C/k}$

Conflict Learning

Additional clauses:

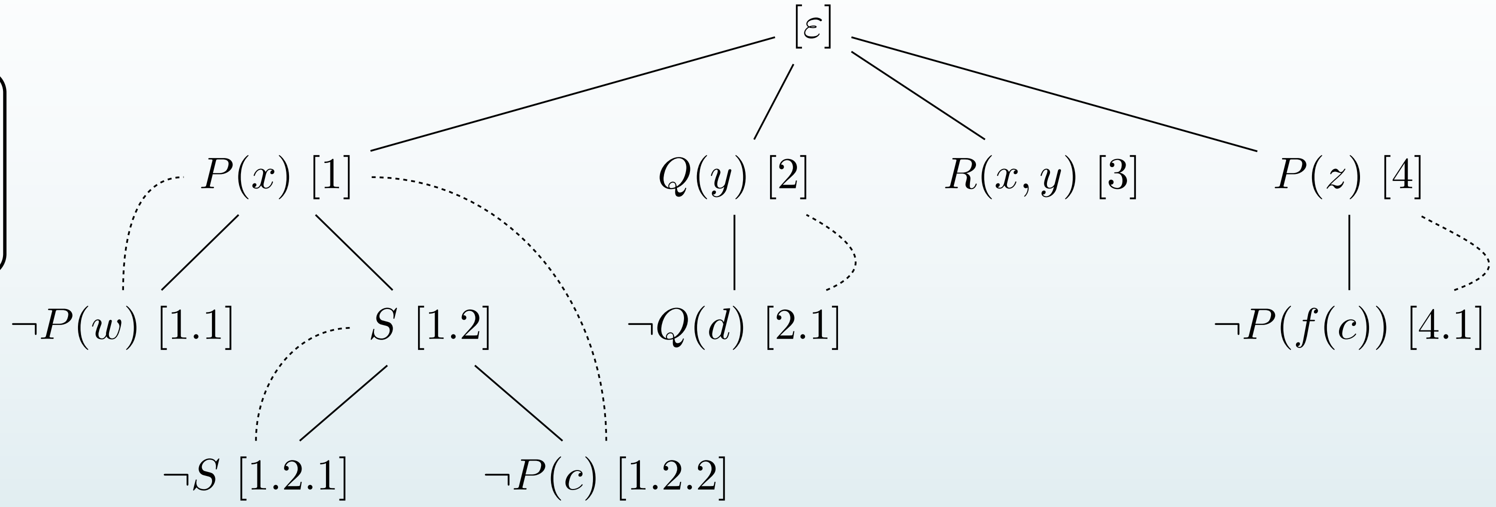
$\forall x. \neg R(d, x)$

$\forall x. \neg R(x, c)$



Conflict Learning

Additional clauses:
$\forall x. \neg R(d, x)$
$\forall x. \neg R(x, c)$



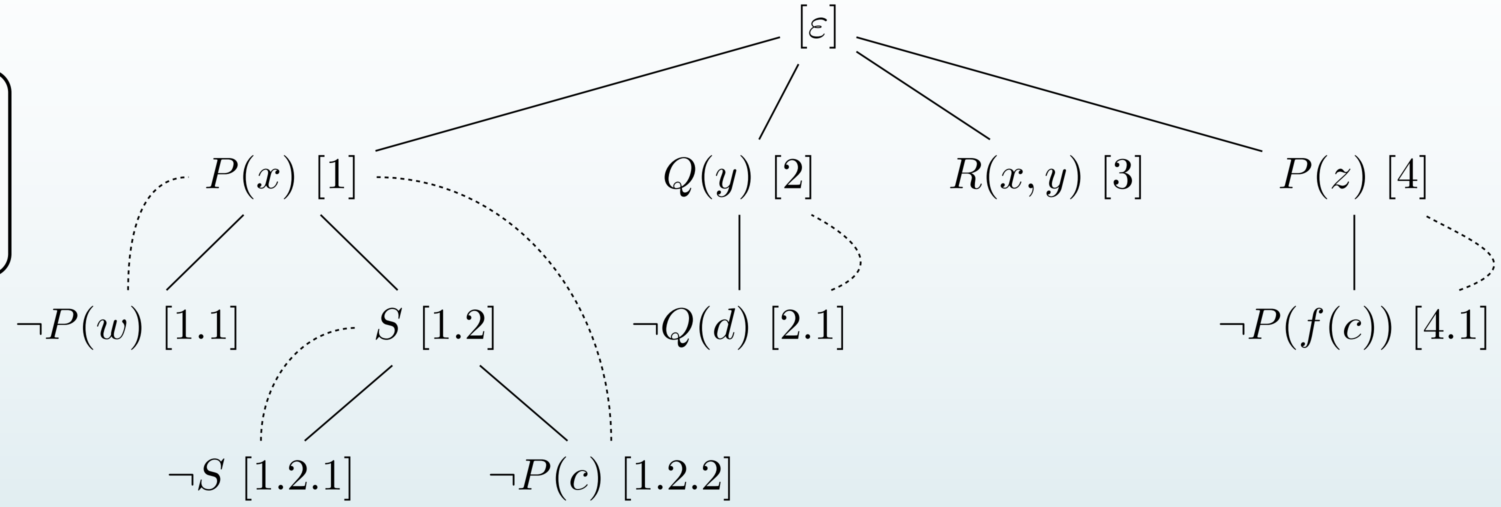
- As $x \mapsto c$ and $y \mapsto d$ we need to connect to $R(c, d)$ – but we cannot

Conflict Learning

Additional clauses:

$\forall x. \neg R(d, x)$

$\forall x. \neg R(x, c)$



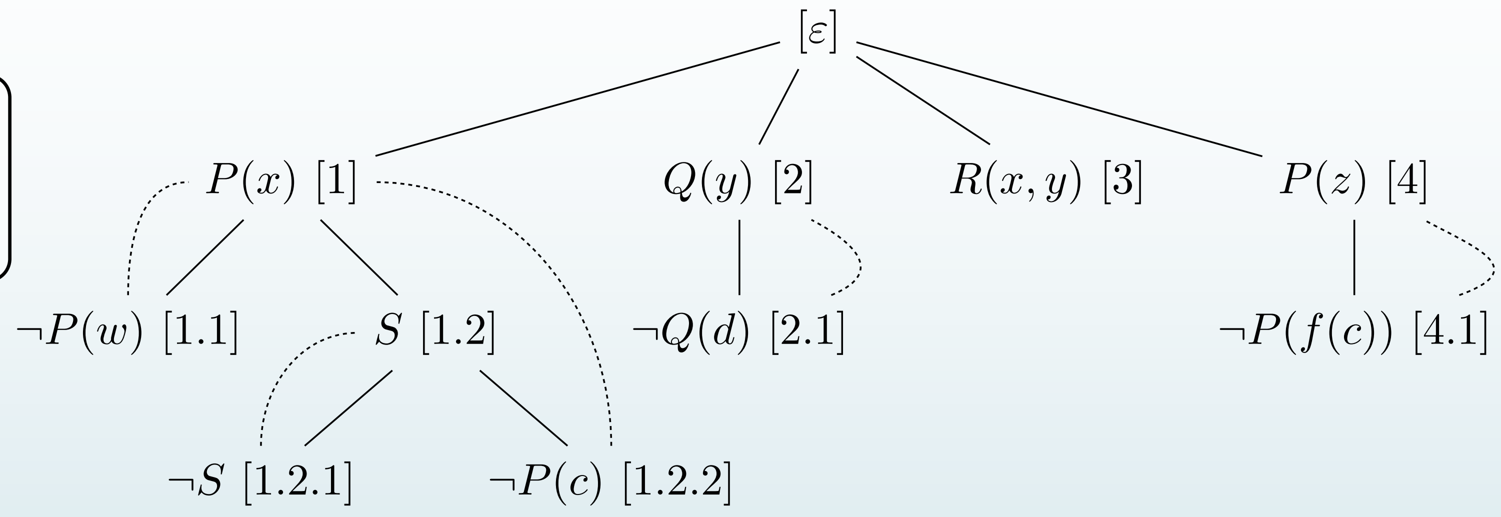
- As $x \mapsto c$ and $y \mapsto d$ we need to connect to $R(c, d)$ – but we cannot
- Stuck by union of justification

$$\{ E^{1}_{\neg P(x) \vee S/1},\ E^{1.2}_{\neg S \vee \neg P(c)/1},\ R^{1}_{1.2.2} \} \cup$$

$$\{ E^{2}_{\neg \mathrm{Q(y)}/1} \}$$

Conflict Learning

Additional clauses:
$\forall x. \neg R(d, x)$
$\forall x. \neg R(x, c)$



- As $x \mapsto c$ and $y \mapsto d$ we need to connect to $R(c, d)$ – but we cannot
- Stuck by union of justification

$$\{\, E^{1}_{\neg P(x) \vee S/1},\ E^{1.2}_{\neg S \vee \neg P(c)/1},\ R^{1}_{1.2.2} \,\} \cup$$

$$\{\, E^{2}_{\neg \mathrm{Q(y)}/1} \,\}$$

We learn the clause:

$$\neg S_{P(\mathrm{x}) \vee Q(\mathrm{y}) \vee R(x,y) \vee P(z)} \vee \neg E^{1}_{\neg P(x) \vee S/1} \vee \neg E^{1.2}_{\neg S \vee \neg P(c)/1} \vee \neg R^{1}_{1.2.2} \vee \neg E^{2}_{\neg \mathrm{Q(y)}/1}$$

Observations

Observations

- $E^{4}_{P(f(c))}$ is not part of the conflict
  - We indeed learn non-trivial stuff

Observations

- $E^{4}_{P(f(c))}$ is not part of the conflict
  - We indeed learn non-trivial stuff
- Learned clauses are purely negative

Observations

- $E^4_{P(f(c))}$ is not part of the conflict
  - We indeed learn non-trivial stuff
- Learned clauses are purely negative
- Decisions are purely positive

Observations

- $E^4_{P(f(c))}$ is not part of the conflict
  - We indeed learn non-trivial stuff
- Learned clauses are purely negative
- Decisions are purely positive
- No unit propagation (only detect violations)
  - 1-watched literal schema

Observations

- $E^{4}_{P(f(c))}$ is not part of the conflict
  - We indeed learn non-trivial stuff
- Learned clauses are purely negative
- Decisions are purely positive
- No unit propagation (only detect violations)
  - 1-watched literal schema
- **Sound**

Observations

- $E^4_{P(f(c))}$ is not part of the conflict
  - We indeed learn non-trivial stuff
- Learned clauses are purely negative
- Decisions are purely positive
- No unit propagation (only detect violations)
  - 1-watched literal schema
- **Sound**
- **Complete**

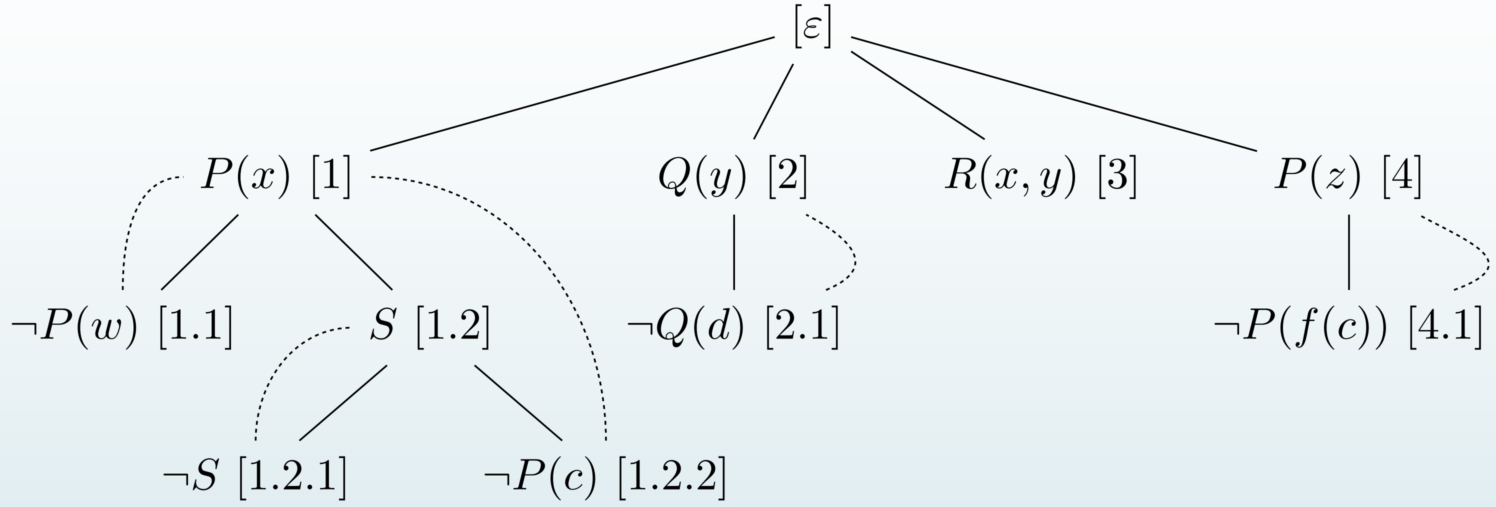
Observations

- $E^4_{P(f(c))}$ is not part of the conflict
  - We indeed learn non-trivial stuff
- Learned clauses are purely negative
- Decisions are purely positive
- No unit propagation (only detect violations)
  - 1-watched literal schema
- **Sound**
- **Complete**
- Terminating (for a fixed depth limit)

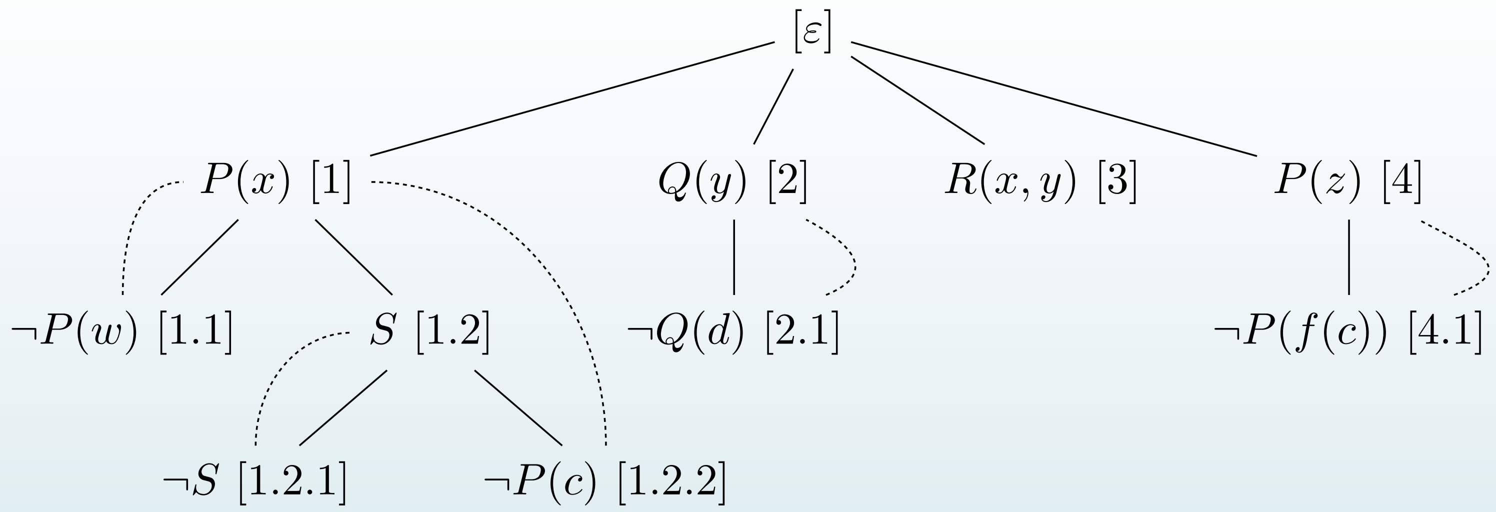
Observations

- $E^4_{P(f(c))}$ is not part of the conflict
  - We indeed learn non-trivial stuff
- Learned clauses are purely negative
- Decisions are purely positive
- No unit propagation (only detect violations)
  - 1-watched literal schema
- **Sound**
- **Complete**
- Terminating (for a fixed depth limit)
- Conflicts inherently **depending** on **precise paths** ☹

More Refined Encoding

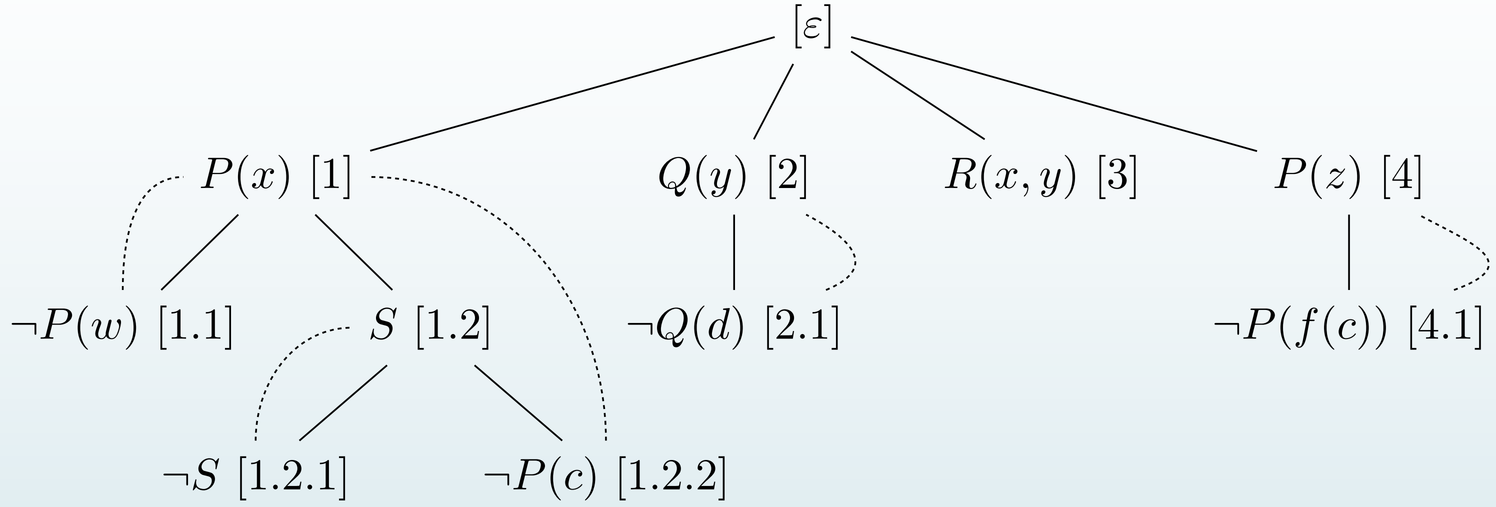


More Refined Encoding



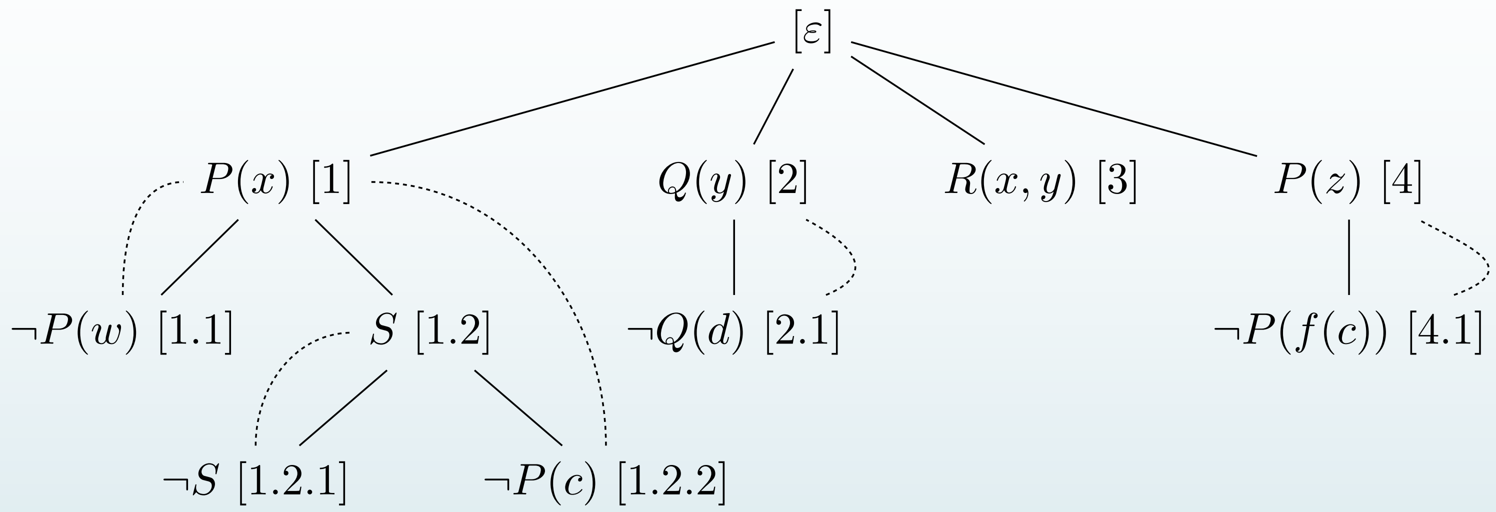
- Still maintain positions

More Refined Encoding



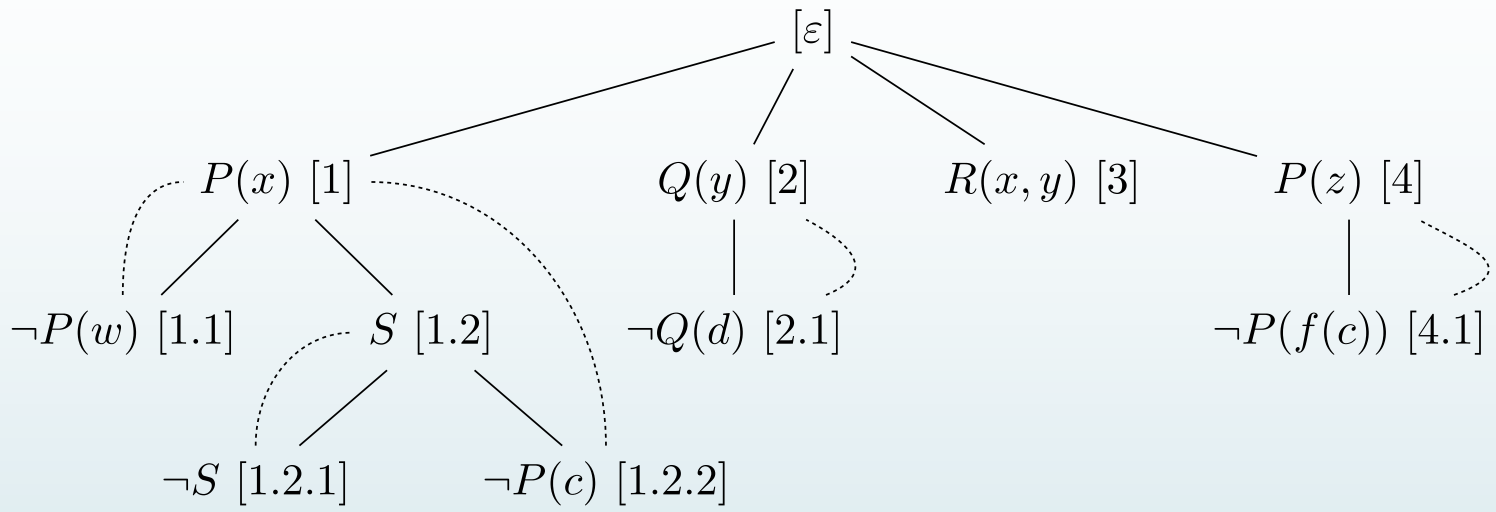
- Still maintain positions
- We have **two kinds** of literals

More Refined Encoding



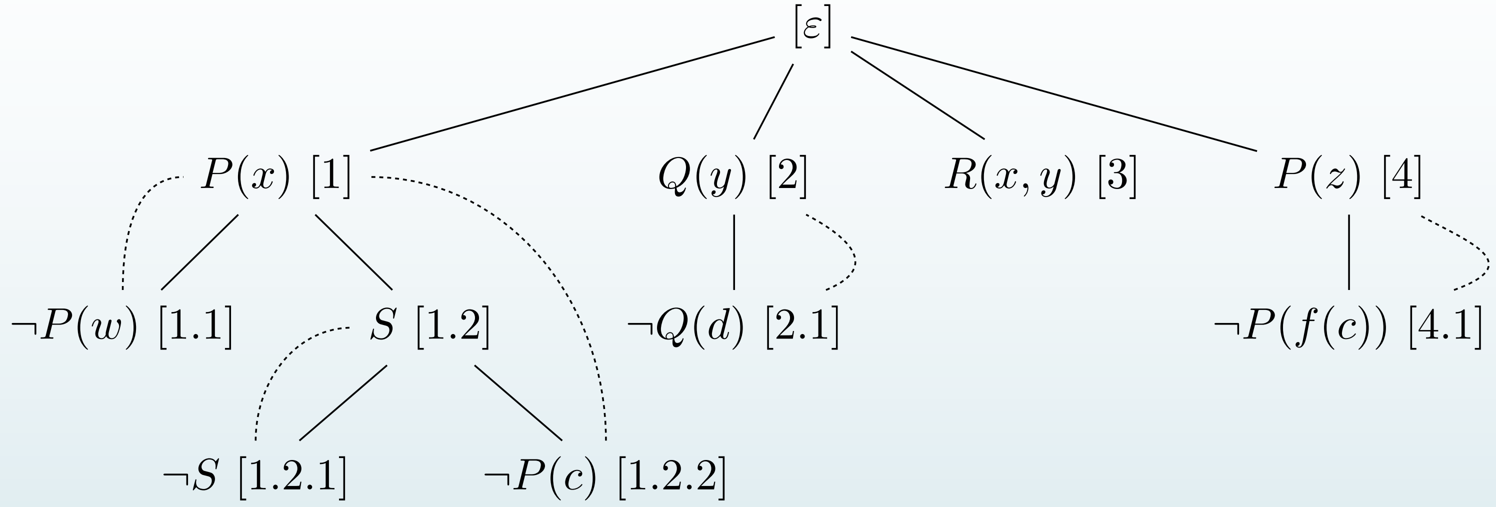
- Still maintain positions
- We have **two kinds** of literals
  1. **Literal** at **position** – e.g., $\langle P(x)@1\rangle$ or $\langle \neg P(c)@1.2.2\rangle$

More Refined Encoding



- Still maintain positions
- We have **two kinds** of literals
  1. **Literal** at **position** – e.g., $\langle P(x)@1\rangle$ or $\langle \neg P(c)@1.2.2\rangle$
  2. Variable **bindings** – e.g., $x \mapsto w$ or $x \mapsto c$

More Refined Encoding



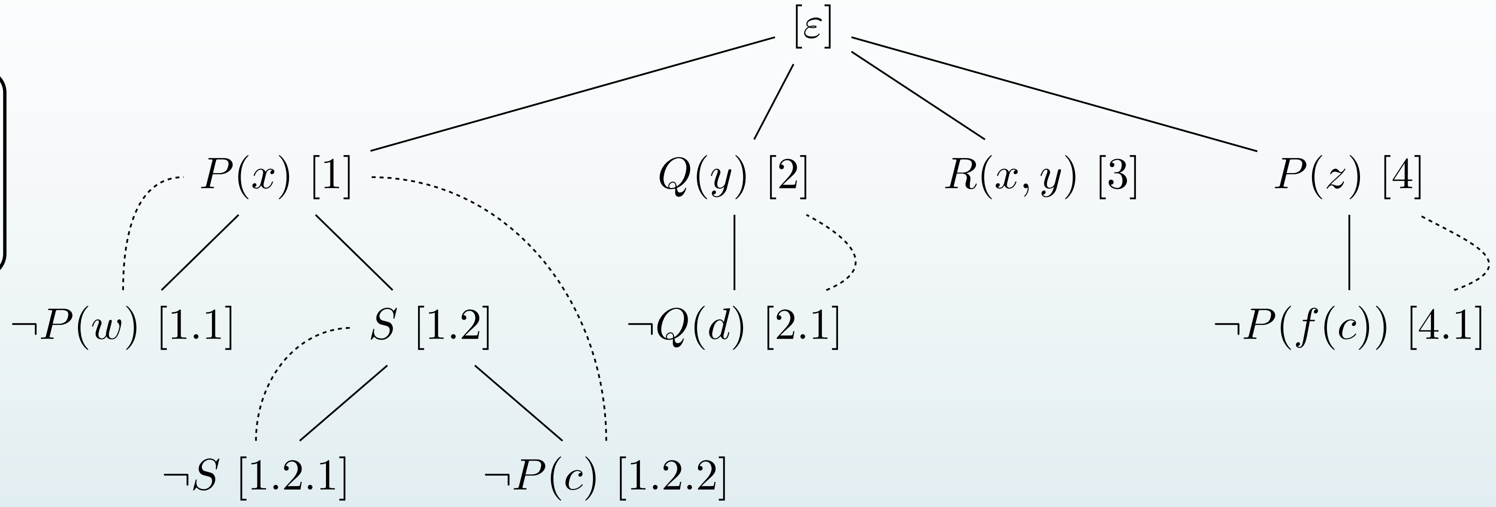
- Still maintain positions
- We have **two kinds** of literals
  1. **Literal** at **position** – e.g., ⟨P(x)@1⟩ or ⟨¬P(c)@1.2.2⟩
  2. Variable **bindings** – e.g., $x \mapsto w$ or $x \mapsto c$
- Major difference: The **origin of bindings** [extension/reduction] is **not tracked**

Refined Conflict Learning

Additional clauses:

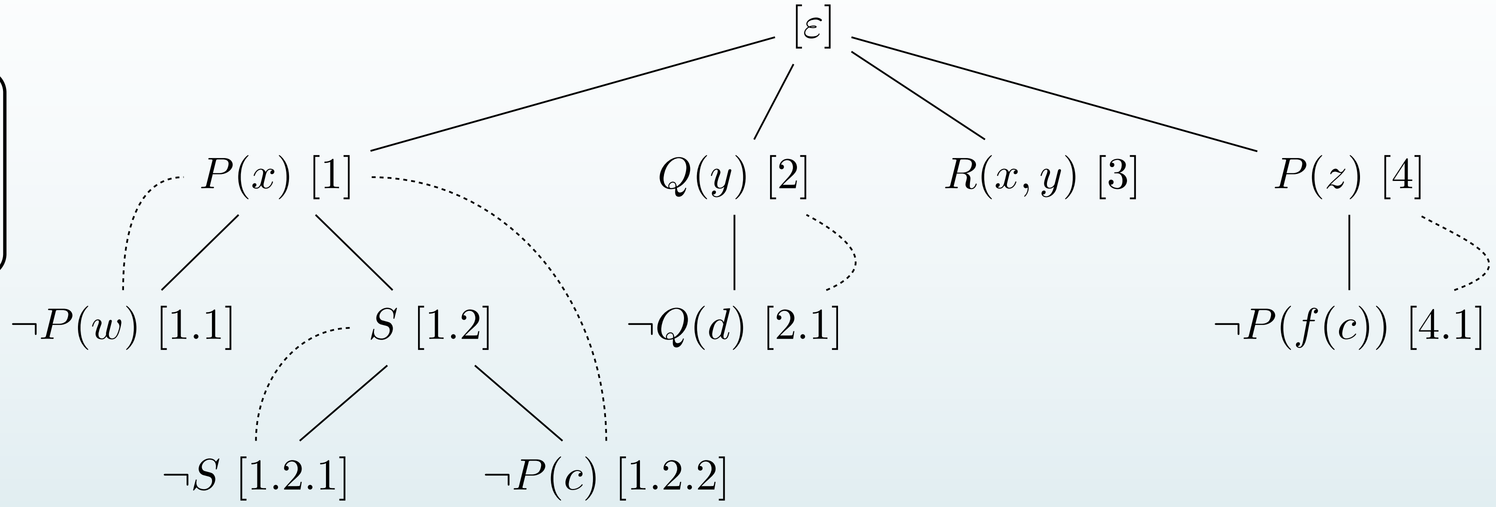
$\forall x. \neg R(d, x)$

$\forall x. \neg R(x, c)$



Refined Conflict Learning

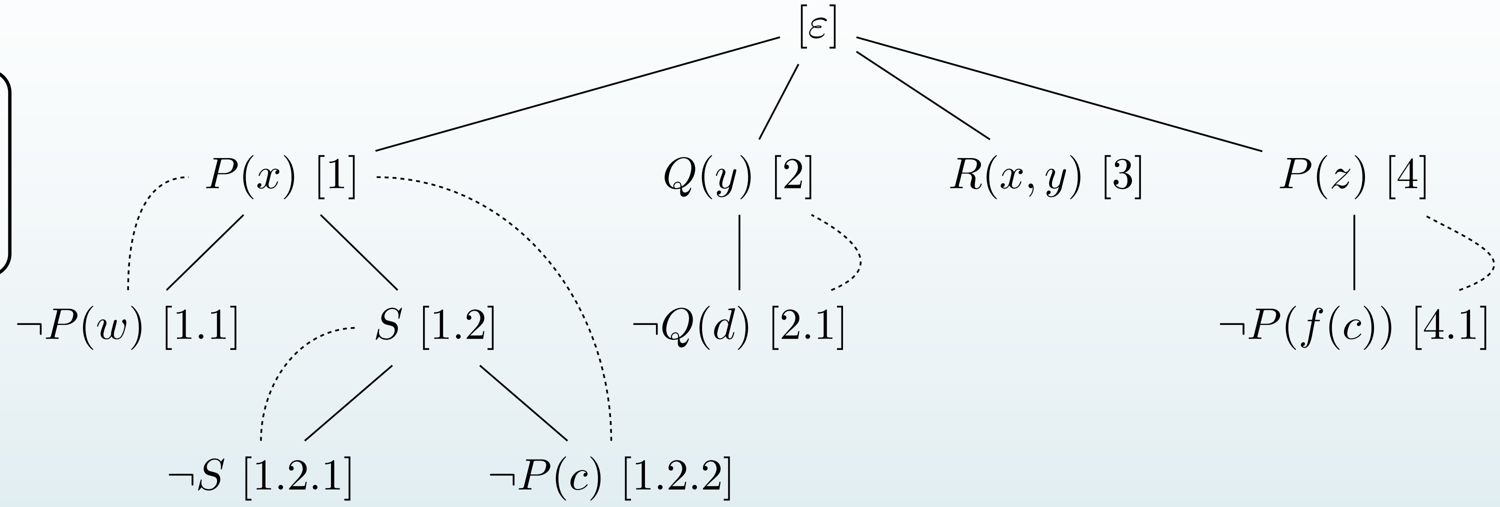
Additional clauses:
$\forall x. \neg R(d,x)$
$\forall x. \neg R(x,c)$



- As $x \mapsto c$ and $y \mapsto d$ we need to connect to $R(c,d)$ – but we cannot

Refined Conflict Learning

Additional clauses:
$\forall x. \neg R(d, x)$
$\forall x. \neg R(x, c)$



- As $x \mapsto c$ and $y \mapsto d$ we need to connect to $R(c, d)$ – but we cannot
- Stuck by union of justification

  $\{x \mapsto c\} \cup$

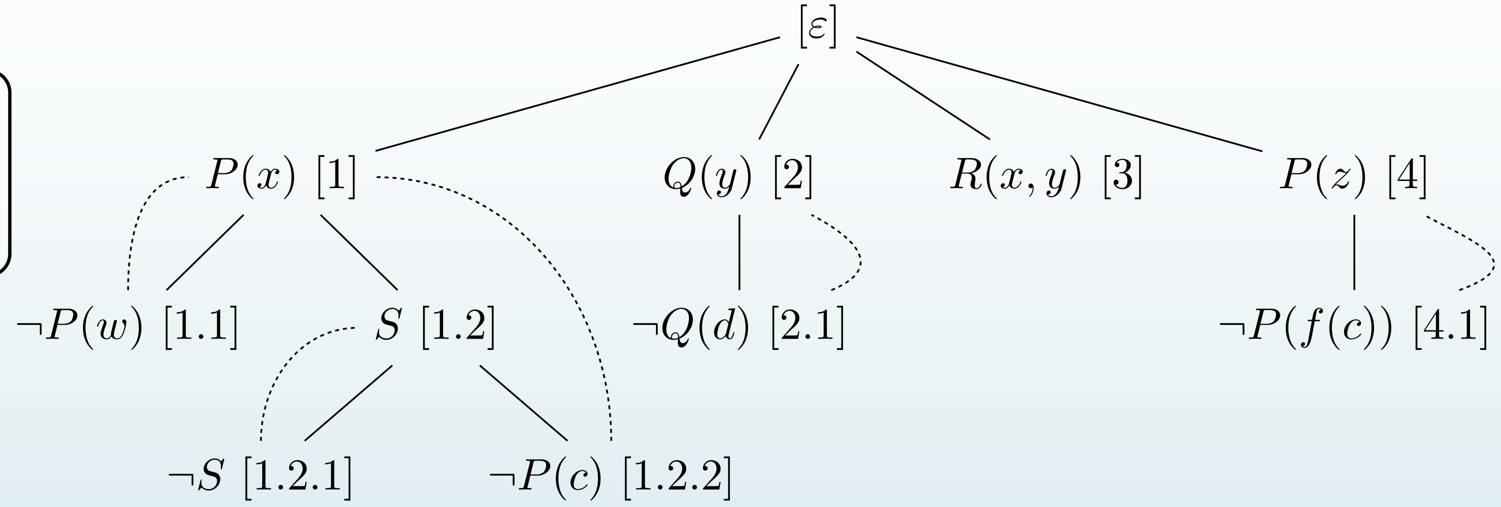
  $\{y \mapsto d\}$

Refined Conflict Learning

Additional clauses:

$\forall x. \neg R(d, x)$

$\forall x. \neg R(x, c)$



- As $x \mapsto c$ and $y \mapsto d$ we need to connect to $R(c, d)$ – but we cannot
- Stuck by union of justification

$\{ x \mapsto c \} \cup$

$\{ y \mapsto d \}$

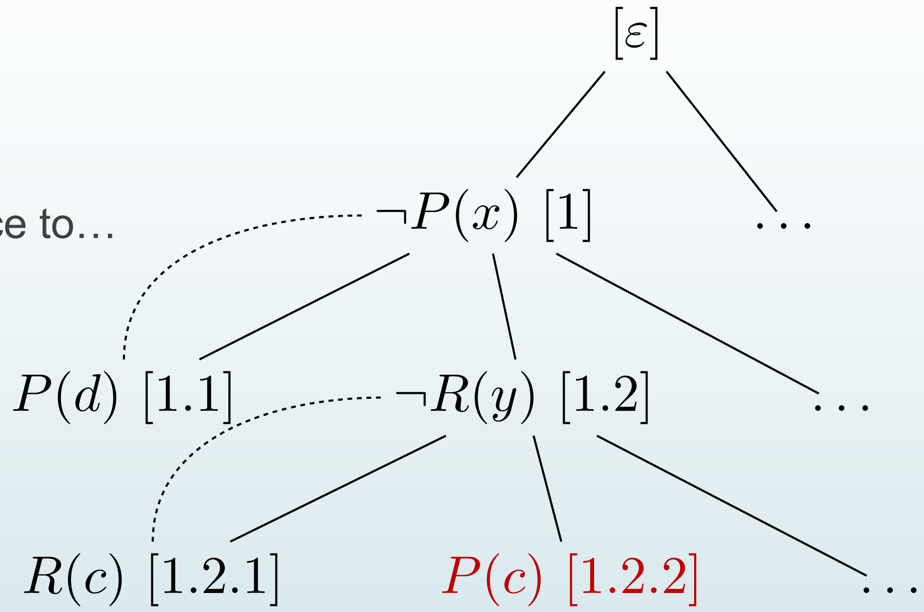
We learn the clause:

$$\neg\langle R(x, y)@3\rangle \lor \neg(x \mapsto c) \lor \neg(y \mapsto d)$$

Wait a Minute! What about Reduction?

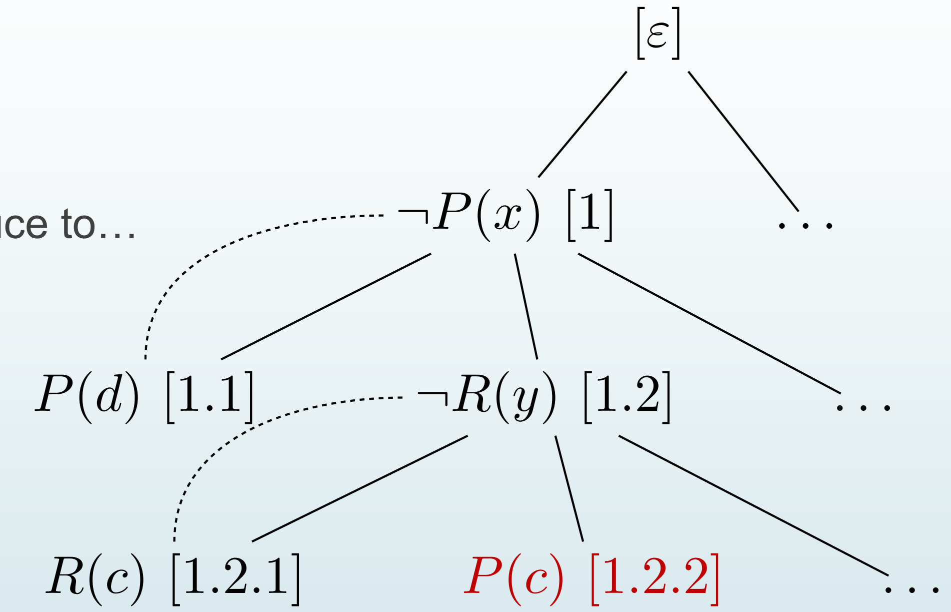
Wait a Minute! What about Reduction?

- More complicated
  - Express: there is nothing we can reduce to…



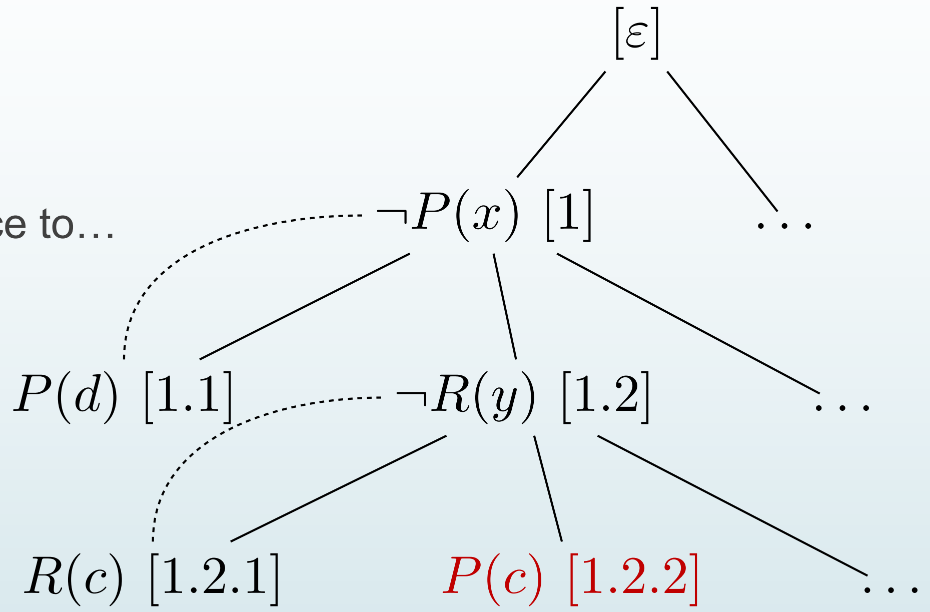
Wait a Minute! What about Reduction?

- More complicated
  - Express: there is nothing we can reduce to…
  - Given $\mu(x) \mapsto d,\ \mu(y) \mapsto c$



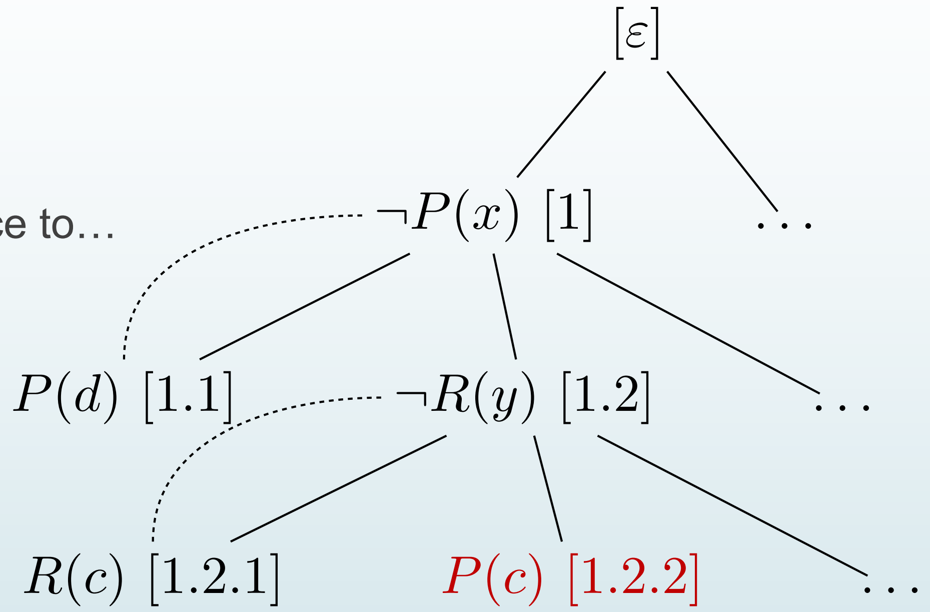
Wait a Minute! What about Reduction?

- More complicated
  - Express: there is nothing we can reduce to…
  - Given $\mu(x) \mapsto d,\ \mu(y) \mapsto c$



  - Either $P(c)$ can be extended **or** it reduces with anything above
    - $\neg\langle P(c)@1.2.2\rangle \lor \neg Ext_1 \lor \dots \lor \neg Ext_n \lor \neg\langle \neg P(x)@1\rangle \lor \neg(x \mapsto \mathrm{c}) \lor \neg\langle \neg \boldsymbol{R}(\boldsymbol{y})@\mathbf{1.2}\rangle$
    - $\neg\langle \neg \boldsymbol{R}(\boldsymbol{y})@\mathbf{1.2}\rangle$ is unnecessarily specific

Wait a Minute! What about Reduction?



- More complicated
  - Express: there is nothing we can reduce to…
  - Given $\mu(x) \mapsto d,\ \mu(y) \mapsto c$
  - Either $P(c)$ can be extended **or** it reduces with anything above
    - $\neg\langle P(c)@1.2.2\rangle \lor \neg Ext_1 \lor \dots \lor \neg Ext_n \lor \neg\langle\neg P(x)@1\rangle \lor \neg(x \mapsto \mathrm{c}) \lor \neg\langle\neg \boldsymbol{R(y)@1.2}\rangle$
    - $\neg\langle\neg \boldsymbol{R(y)@1.2}\rangle$ is unnecessarily specific
  - ➔ We use auxiliary **"could connect" literals** $p_i \sim p_j$
    - $\neg\langle P(c)@1.2.2\rangle \lor \neg Ext_1 \lor \dots \lor \neg Ext_n \lor \neg\langle\neg P(x)@1\rangle \lor \neg(x \mapsto \mathrm{c}) \lor \mathbf{1.2.2} \sim \mathbf{1.2}$

Results - I

- Prototype *hopCoP*
- Compared against *meanCoP*

Results - I

- Prototype *hopCoP*
- Compared against *meanCoP*

- *Solved instances*

	M2k	Miz40	MPTP - bushy	MPTP - chainy	TPTP
hopCoP	**1 050**	**13 040**	**589**	203	**4 026**
meanCoP	795	7 592	480	157	3 578
meanCoP ✄	878	9 748	562	**337**	3 283

Results - I

- Prototype *hopCoP*
- Compared against *meanCoP*

Sounds way better than before!

- *Solved instances*

	M2k	Miz40	MPTP - bushy	MPTP - chainy	TPTP
hopCoP	**1 050**	**13 040**	**589**	203	**4 026**
meanCoP	795	7 592	480	157	3 578
meanCoP ✂	878	9 748	562	**337**	3 283

Results - II

Results - II

- Extension steps for *PUZ005-1.p* (lower = better)

	Lvl. 1	Lvl. 2	Lvl. 3	Lvl. 4	Lvl. 5	Lvl. 6	Lvl. 7
hopCoP	1	4	89	495	2 309	10 066	**48 517**
meanCoP	**1**	**4**	**24**	**108**	**535**	**9 963**	6 445 008

CASC Participation

CASC Participation

- *hopCoP* participated in **CASC30** [2025]

CASC Participation

- *hopCoP* participated in **CASC30** [2025]
  - Random restart + random literal selection
  - Solved 88 out of 500 inputs

First-order Theorems	*Vampire* *4.9*	Vampire 5.0	CSI_Enig 1.0.6	iProver 3.9.3	E 3.3.0	Drodi 4.1.0	CSE_E 1.7	cvc5 1.3.0	Zipperpin 2.1.9999	Prover9 1109a	ConnectP 0.6.1	hopCoP 0.1	LisaTT 0.9.1	SPASS-SC 0.1	LastButN 0
Solved/500	466/500	455/500	402/500	367/500	364/500	325/500	295/500	290/500	267/500	119/500	102/500	88/500	3/500	11/500	0/500
Solutions/500	466/500	455/500	402/500	367/500	364/500	325/500	293/500	290/500	267/500	119/500	102/500	88/500	3/500	0/500	0/500

- **Not bad** for a newcomer based on CC!

Further Work

Further Work

- We block certain assignments
  - e.g., $\neg(x \mapsto d)$

Further Work

- We block certain assignments
  - e.g., $\neg(x \mapsto d)$
  - Enforce unification instead
    - $x \sim c$

  1. Harder to track violation
  2. Logical additional step: Propagate consequences

Further Work

- We block certain assignments
  - e.g., $\neg(\mathrm{x} \mapsto \mathrm{d})$
  - Enforce unification instead
    - $\mathrm{x} \sim \mathrm{c}$

    1. Harder to track violation
    2. Logical additional step: Propagate consequences

- Getting rid of position:
  - e.g., $\neg\langle P(\mathrm{x})@1.1.1\rangle \vee \neg Ext_1 \vee \ldots \vee \neg Ext_n \vee \neg\langle\neg P(\mathrm{c})@1\rangle \vee \neg(x \mapsto c) \vee 1.1.1 \sim 1.1$

Further Work

- We block certain assignments
  - e.g., $\neg(\mathrm{x} \mapsto \mathrm{d})$
  - Enforce unification instead
    - $\mathrm{x} \sim \mathrm{c}$
    1. Harder to track violation
    2. Logical additional step: Propagate consequences

- Getting rid of position:
  - e.g., $\neg\langle P(\mathrm{x})@1.1.1\rangle \vee \neg Ext_1 \vee \ldots \vee \neg Ext_n \vee \neg\langle\neg P(\mathrm{c})@1\rangle \vee \neg(x \mapsto c) \vee 1.1.1 \sim 1.1$
  - Encode there is no parent to reduce
    - $\forall p \forall p' \prec p{:}\ \neg P(x^p)@p \vee \neg Ext_1 \vee \ldots \vee \neg Ext_n \vee p \sim p'$
    1. Even harder to track violation

Summary

Summary

- Introduced two "languages"
  - **Learning from conflicts** during CC tableaux search
  - Strongly influenced by CDCL (SAT)
  - Justifications by "**position**" in the tableau

Summary

- Introduced two "languages"
  - **Learning from conflicts** during CC tableaux search
  - Strongly influenced by CDCL (SAT)
  - Justifications by "**position**" in the tableau
- **Prototype** *hopCoP*

Summary

- Introduced two "languages"
  - **Learning from conflicts** during CC tableaux search
  - Strongly influenced by CDCL (SAT)
  - Justifications by "**position**" in the tableau
- **Prototype** *hopCoP*
- Initial empirical results: **Promising** ☺

Summary

Finally done!

- Introduced two "languages"
  - **Learning from conflicts** during CC tableaux search
  - Strongly influenced by CDCL (SAT)
  - Justifications by "**position**" in the tableau
- **Prototype** *hopCoP*
- Initial empirical results: **Promising** ☺

Summary

Finally done!

- Introduced two "languages"
  - **Learning from conflicts** during CC tableaux search
  - Strongly influenced by CDCL (SAT)
  - Justifications by "**position**" in the tableau
- **Prototype** *hopCoP*
- Initial empirical results: **Promising** ☺

Questions?