DE LA RECHERCHE À L'INDUSTRIE

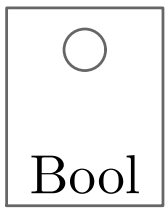
www.cea.fr

SOPRANO : New Solver for Program Analysis

Kickoff | F.Bobot

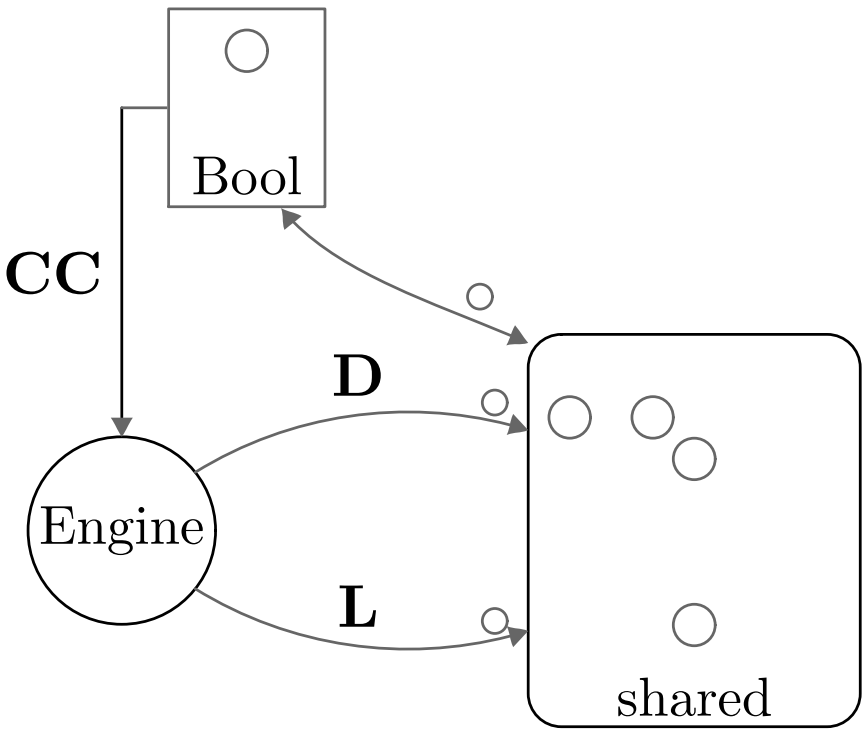
13 Janvier

SMT-solver: Big picture and CDCL



$$(A \vee B \implies C)$$
$$\wedge(C \implies D)$$
$$\wedge(D \wedge E \implies \neg A)$$

SMT-solver: Big picture and CDCL

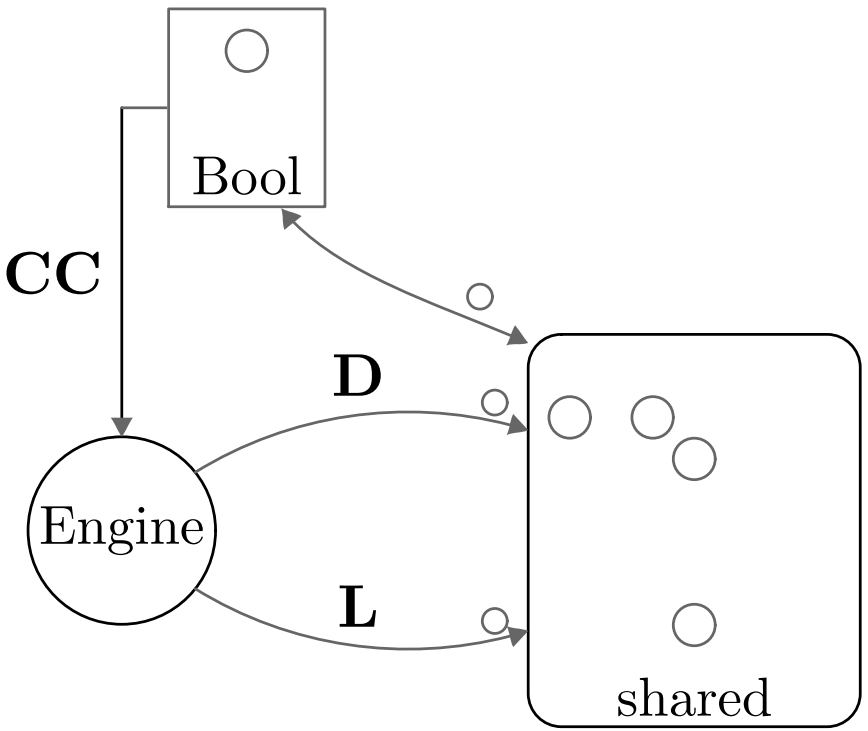


$$(A \vee B \implies C)$$
$$\wedge (C \implies D)$$
$$\wedge (D \wedge E \implies \neg A)$$

$$\frac{}{}$$

$$\frac{E}{}$$

SMT-solver: Big picture and CDCL

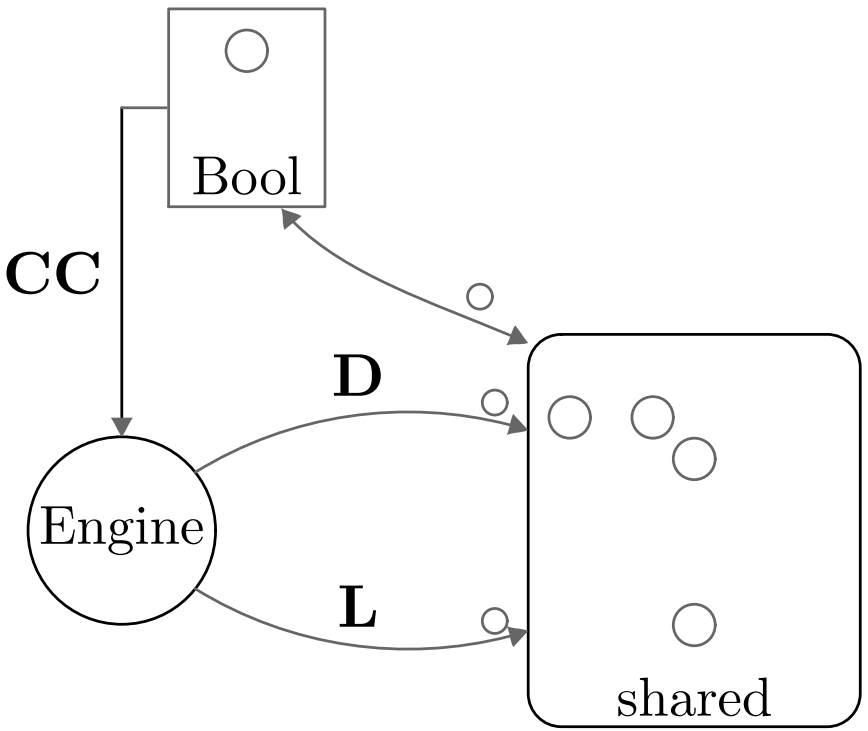


$$(A \vee B \implies C)$$
$$\wedge (C \implies D)$$
$$\wedge (D \wedge E \implies \neg A)$$

$$\frac{}{}$$

$$\frac{E}{}$$

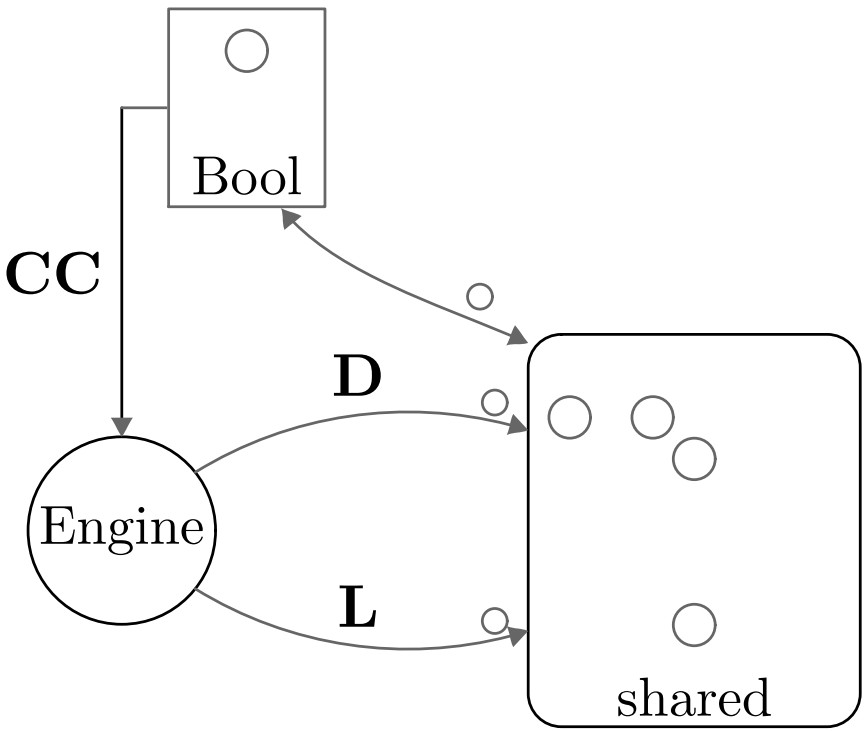
SMT-solver: Big picture and CDCL



$$(A \vee B \implies C)$$
$$\wedge (C \implies D)$$
$$\wedge (D \wedge E \implies \neg A)$$

$$\frac{\vdots}{E}$$

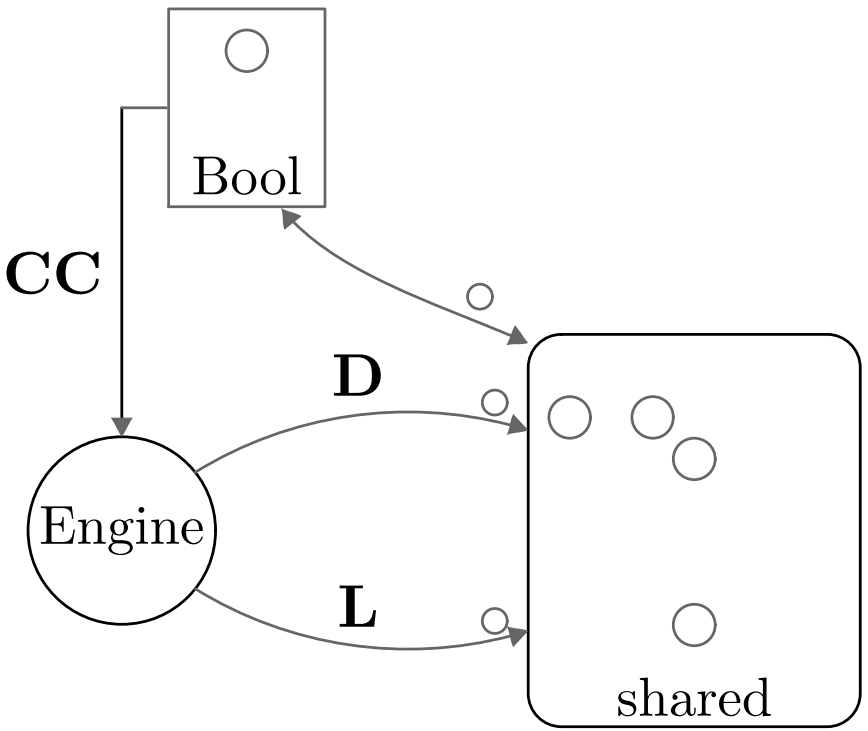
SMT-solver: Big picture and CDCL



$$(A \vee B \implies C)$$
$$\wedge (C \implies D)$$
$$\wedge (D \wedge E \implies \neg A)$$

$$\frac{\begin{array}{c} A_{\mathcal{D}} \\ \vdots \\ E \end{array}}{}$$

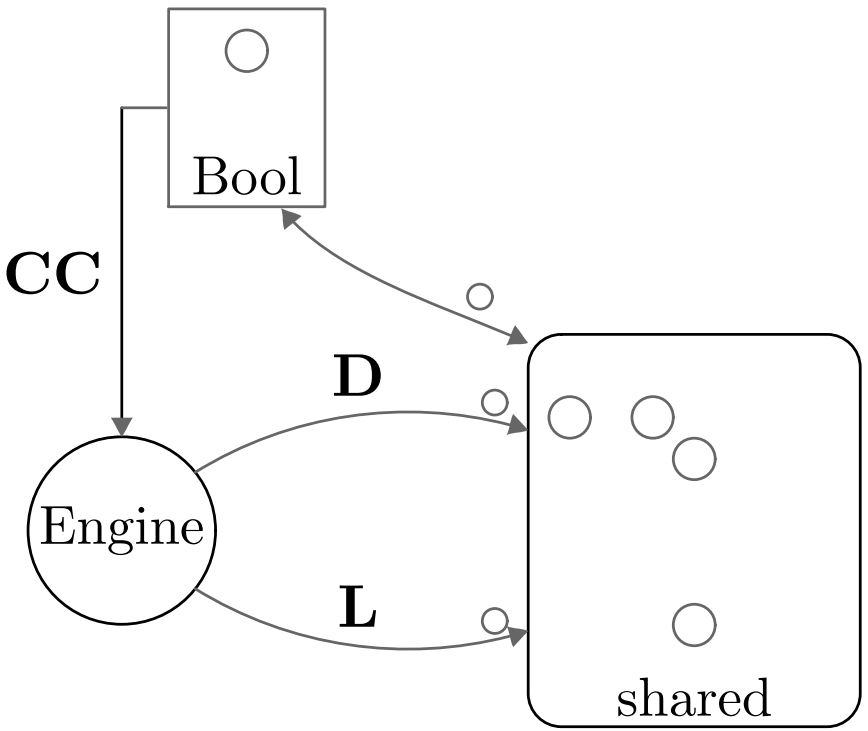
SMT-solver: Big picture and CDCL



$$(A \vee B \implies C)$$
$$\wedge (C \implies D)$$
$$\wedge (D \wedge E \implies \neg A)$$

$$\frac{\begin{array}{c} A_{\mathcal{D}} \\ \vdots \\ E \end{array}}{}$$

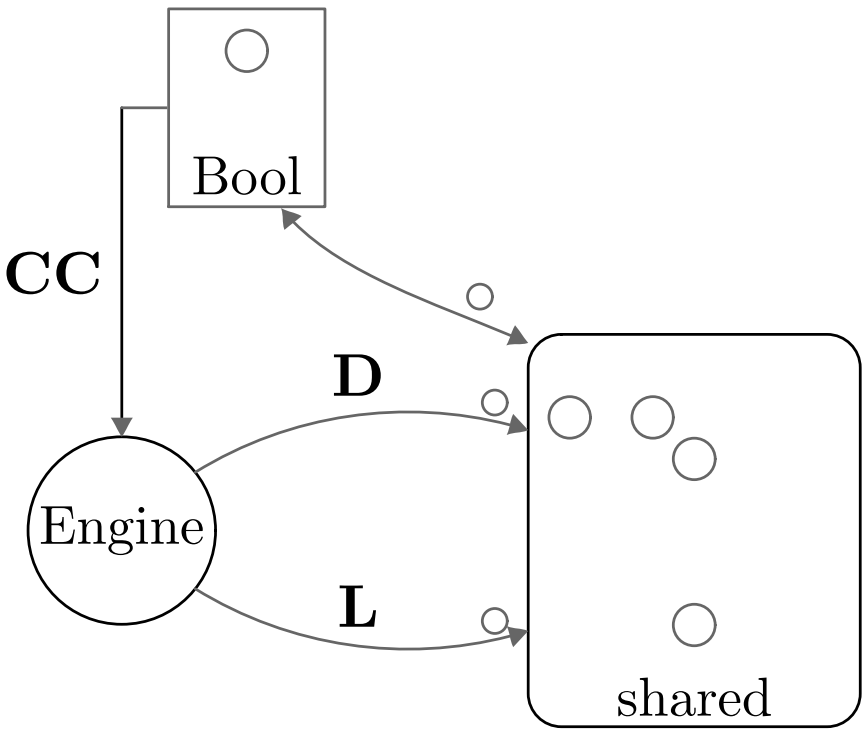
SMT-solver: Big picture and CDCL



$$(A \vee B \implies C)$$
$$\wedge (C \implies D)$$
$$\wedge (D \wedge E \implies \neg A)$$

$$\frac{}{\begin{array}{c} C \\ A_{\mathcal{D}} \\ \vdots \\ E \end{array}}$$

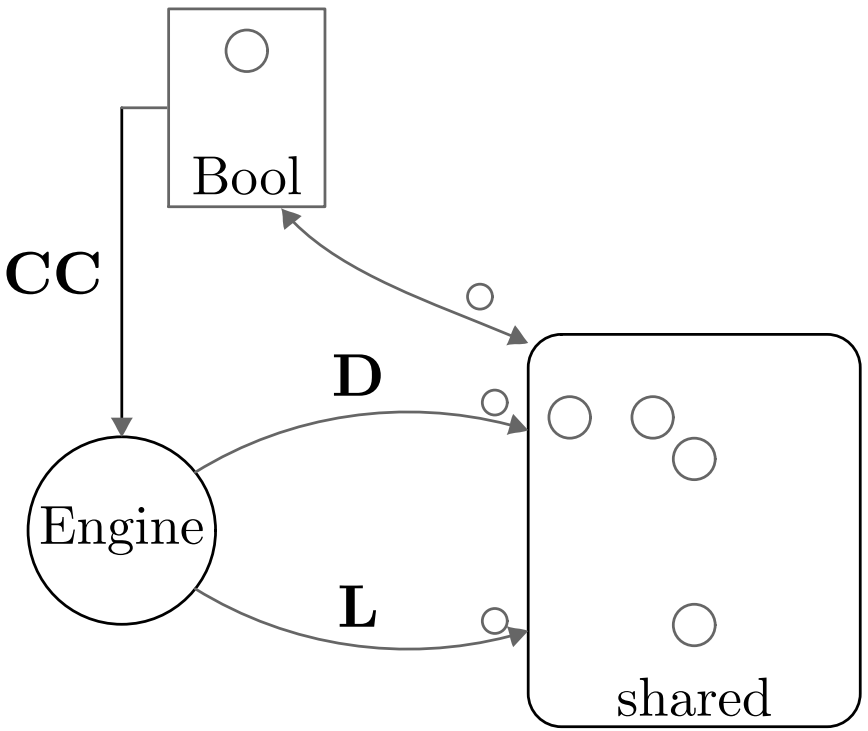
SMT-solver: Big picture and CDCL



$$(A \vee B \implies C)$$
$$\wedge (C \implies D)$$
$$\wedge (D \wedge E \implies \neg A)$$

$$\frac{}{\begin{array}{c} C \\ A_{\mathcal{D}} \\ \vdots \\ E \end{array}}$$

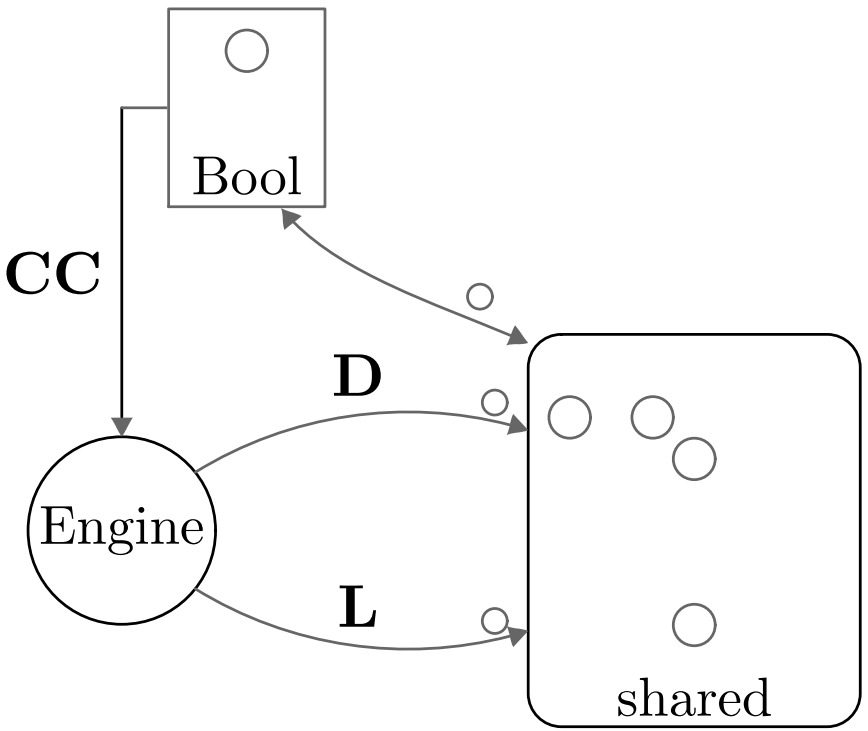
SMT-solver: Big picture and CDCL



$$(A \vee B \implies C)$$
$$\wedge (C \implies D)$$
$$\wedge (D \wedge E \implies \neg A)$$

$$\frac{}{\begin{array}{c} \neg D \\ C \\ A_{\mathcal{D}} \\ \vdots \\ E \end{array}}$$

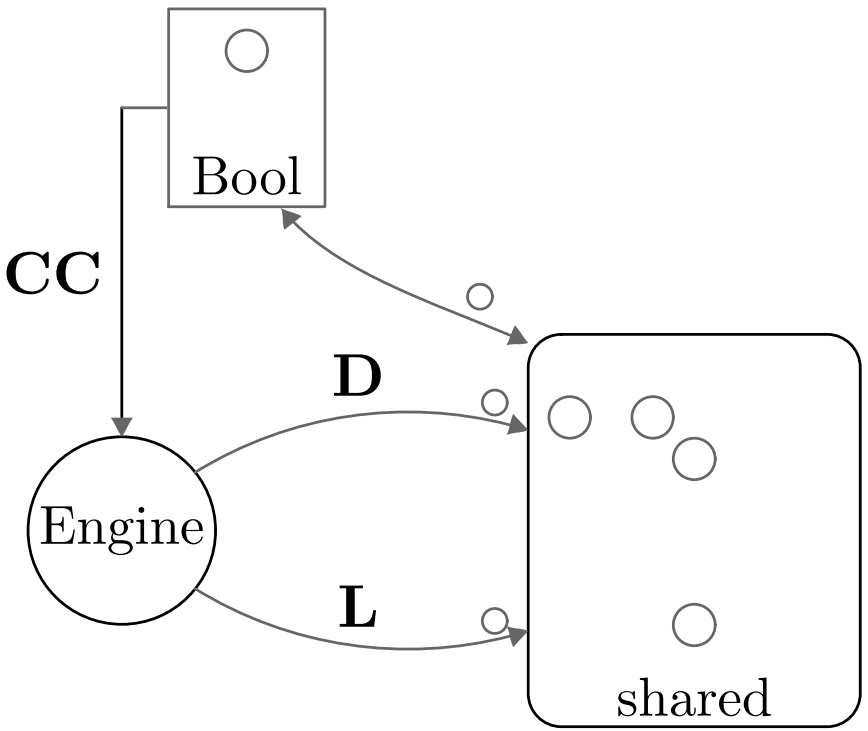
SMT-solver: Big picture and CDCL



$$(A \vee B \implies C)$$
$$\wedge(C \implies D)$$
$$\wedge(D \wedge E \implies \neg A)$$

$$\frac{}{\begin{array}{c}\neg D \\ C \\ A_{\mathcal{D}} \\ \vdots \\ E\end{array}}$$

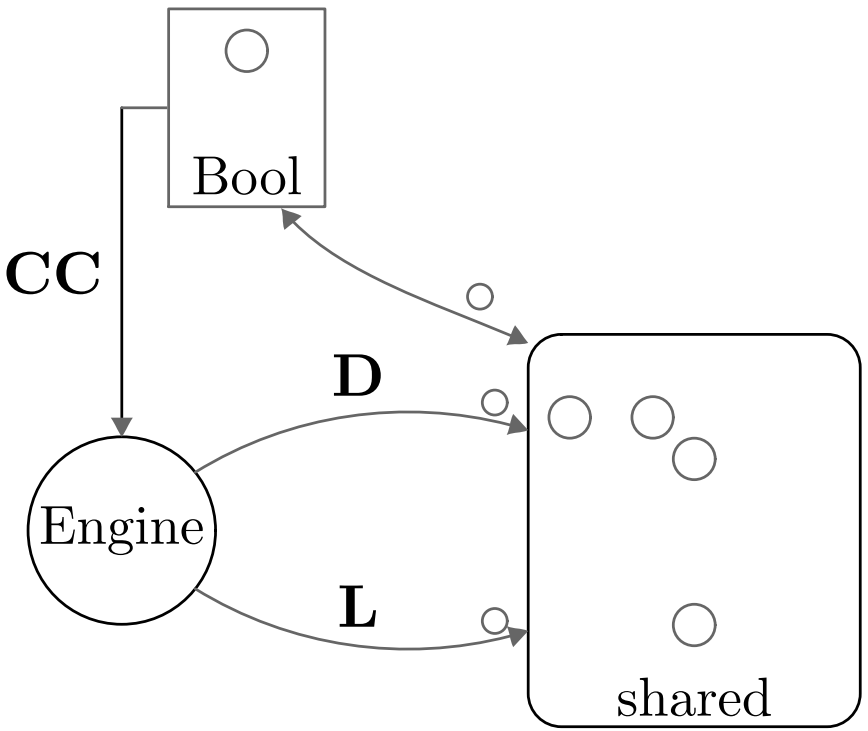
SMT-solver: Big picture and CDCL



$$(A \vee B \implies C)$$
$$\wedge(C \implies D)$$
$$\wedge(D \wedge E \implies \neg A)$$

$$\frac{}{\begin{array}{c}\neg D \\ C \\ A_{\mathcal{D}} \\ \vdots \\ E\end{array}}$$

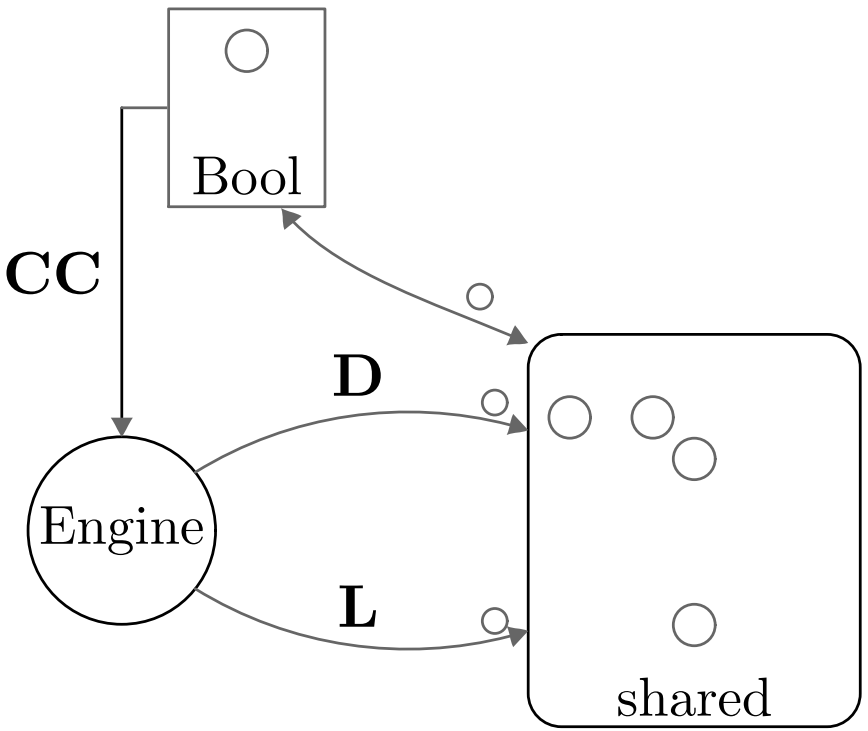
SMT-solver: Big picture and CDCL



$$(A \vee B \implies C)$$
$$\wedge(C \implies D)$$
$$\wedge(D \wedge E \implies \neg A)$$

$$\frac{\begin{array}{c}\bot \\ \neg D \\ C \\ A_{\mathcal{D}} \\ \vdots \\ E\end{array}}{}$$

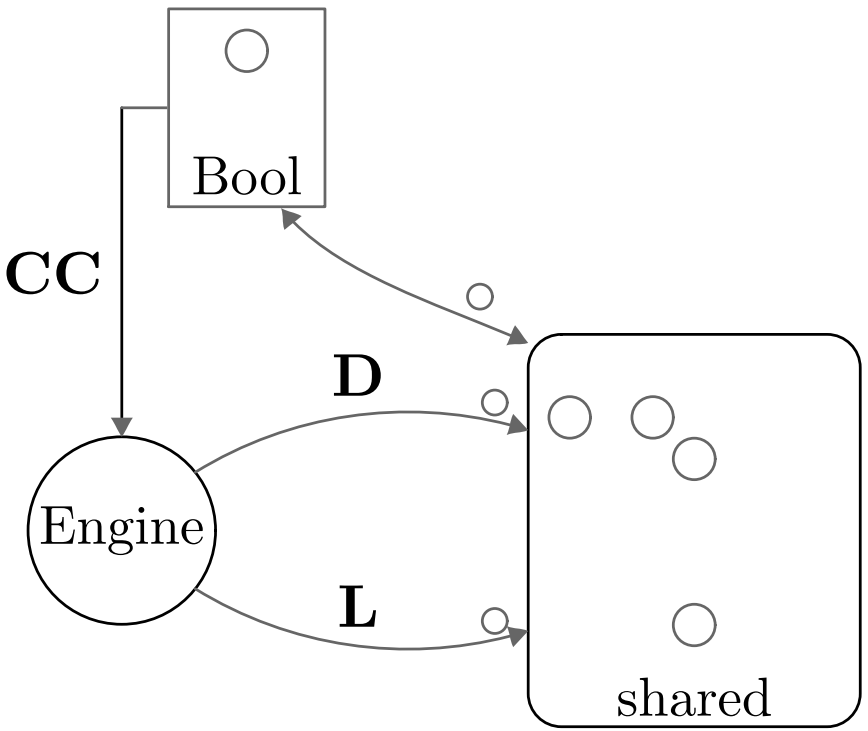
SMT-solver: Big picture and CDCL



$$(A \vee B \implies C)$$
$$\wedge (C \implies D)$$
$$\wedge (D \wedge E \implies \neg A)$$
$$\wedge (\neg E \vee \neg A)$$

$$E$$

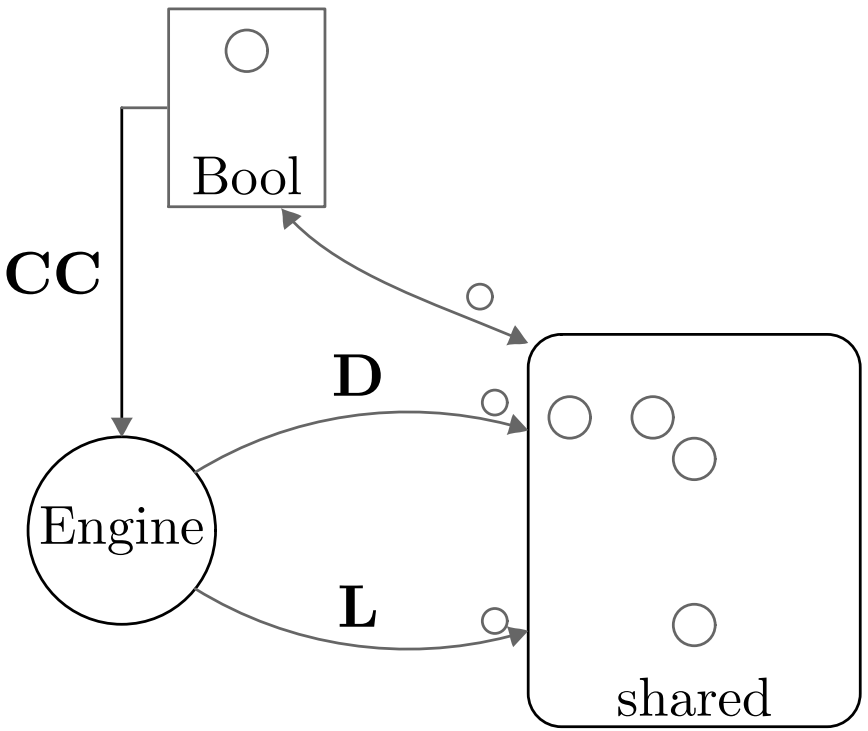
SMT-solver: Big picture and CDCL



$$(A \vee B \implies C)$$
$$\wedge (C \implies D)$$
$$\wedge (D \wedge E \implies \neg A)$$
$$\wedge (\neg E \vee \neg A)$$

$$E$$

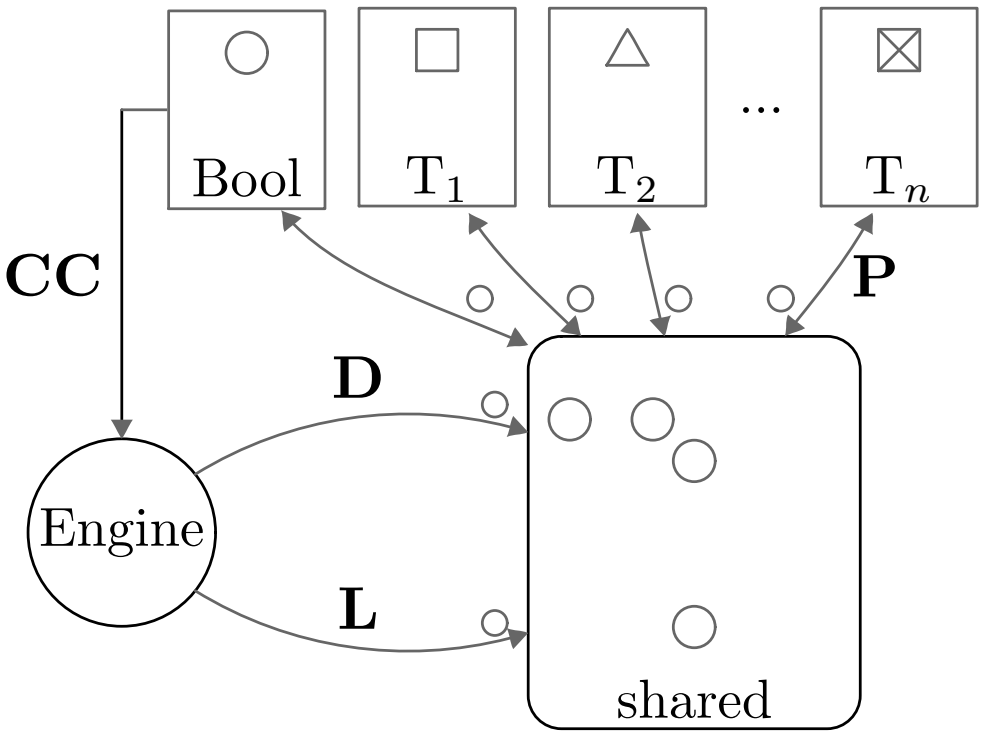
SMT-solver: Big picture and CDCL



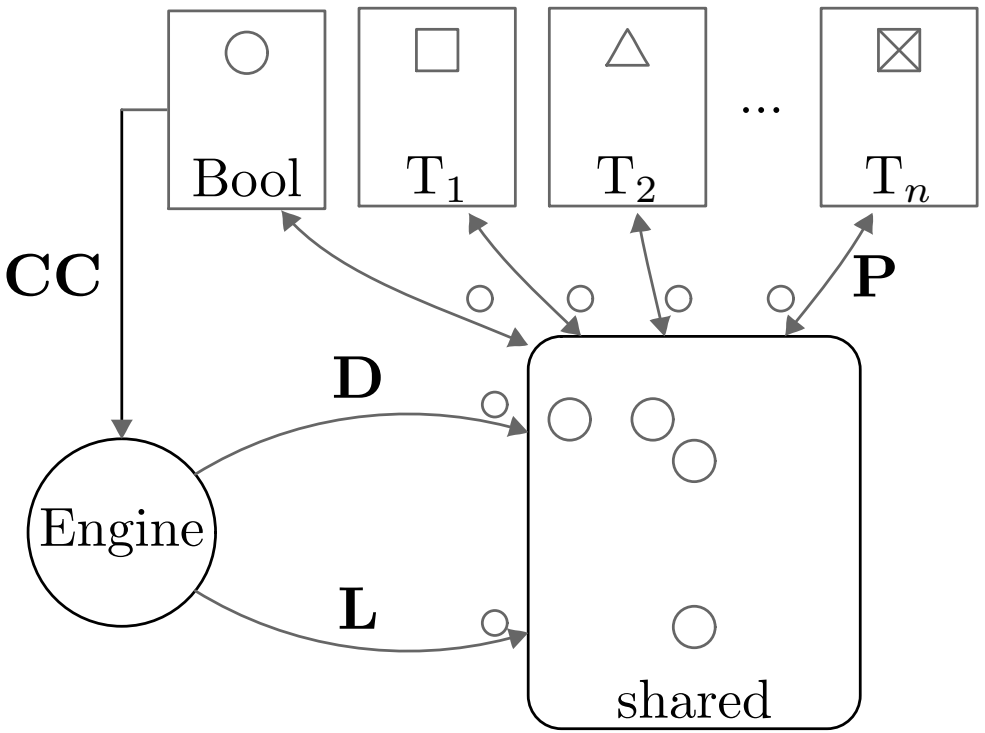
$$(A \vee B \implies C)$$
$$\wedge (C \implies D)$$
$$\wedge (D \wedge E \implies \neg A)$$
$$\wedge (\neg E \vee \neg A)$$

$$\neg A$$
$$E$$

SMT-solver: Big picture and CDCL

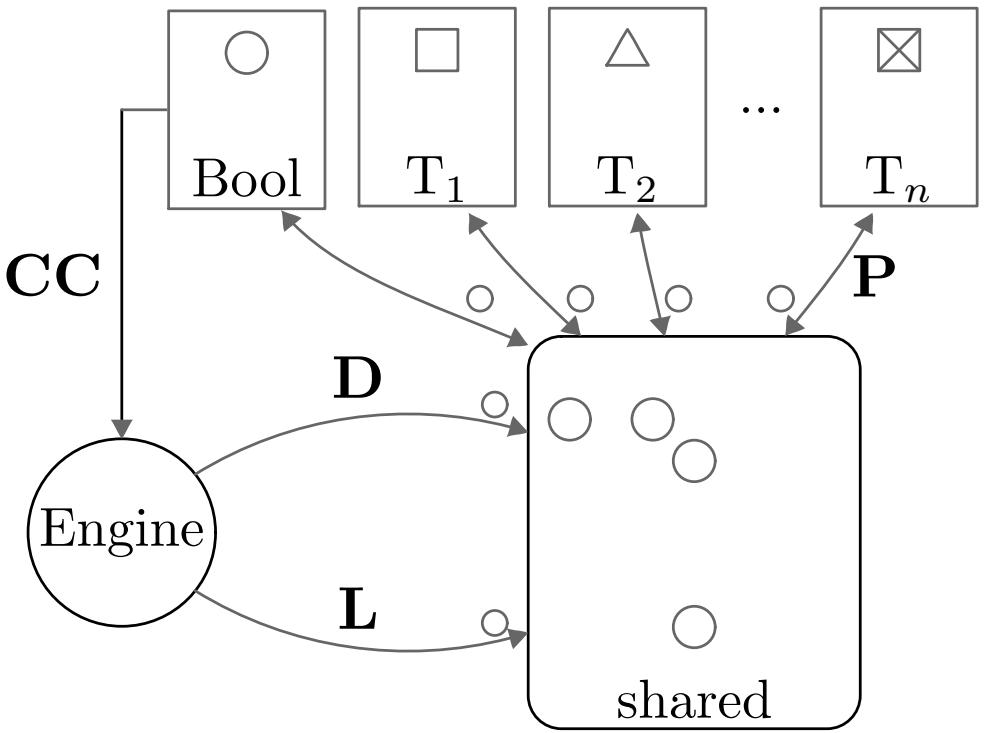


SMT-solver: Big picture and CDCL



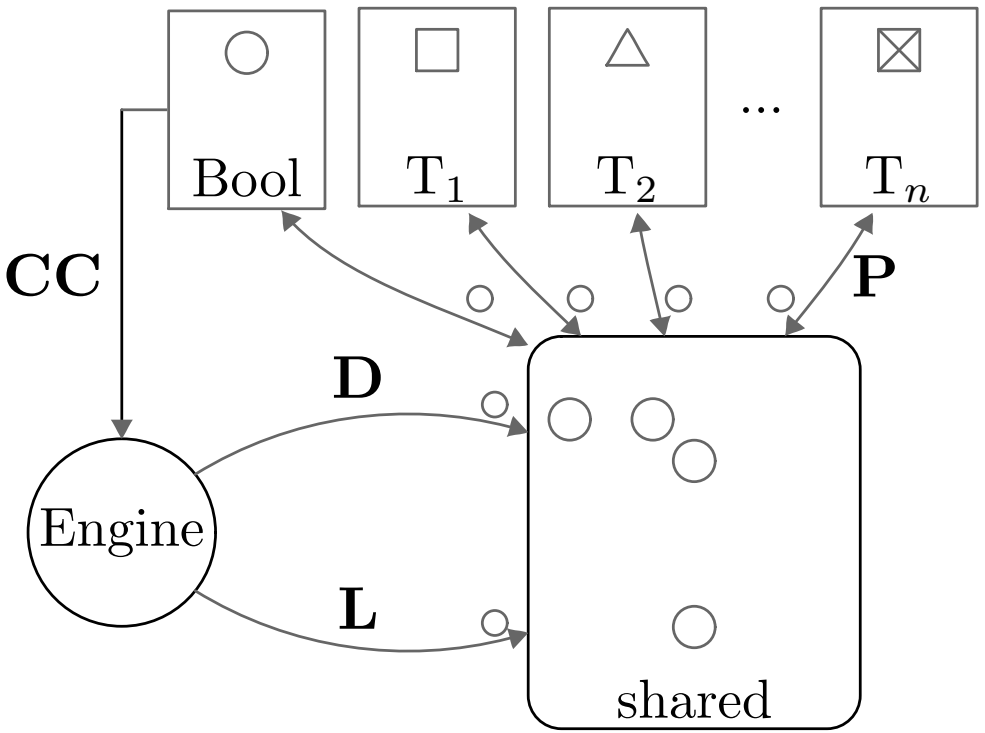
$$(A \vee B \implies C)$$
$$\wedge(C \implies D)$$
$$\wedge(D \wedge E \implies \neg A)$$

SMT-solver: Big picture and CDCL



$$\left(\begin{array}{c} 10 \leq x + t \vee 10 \leq y \\ \Longrightarrow\ 10 \leq z \end{array}\right)$$
$$\wedge\,(10 \leq z \implies t < 0)$$

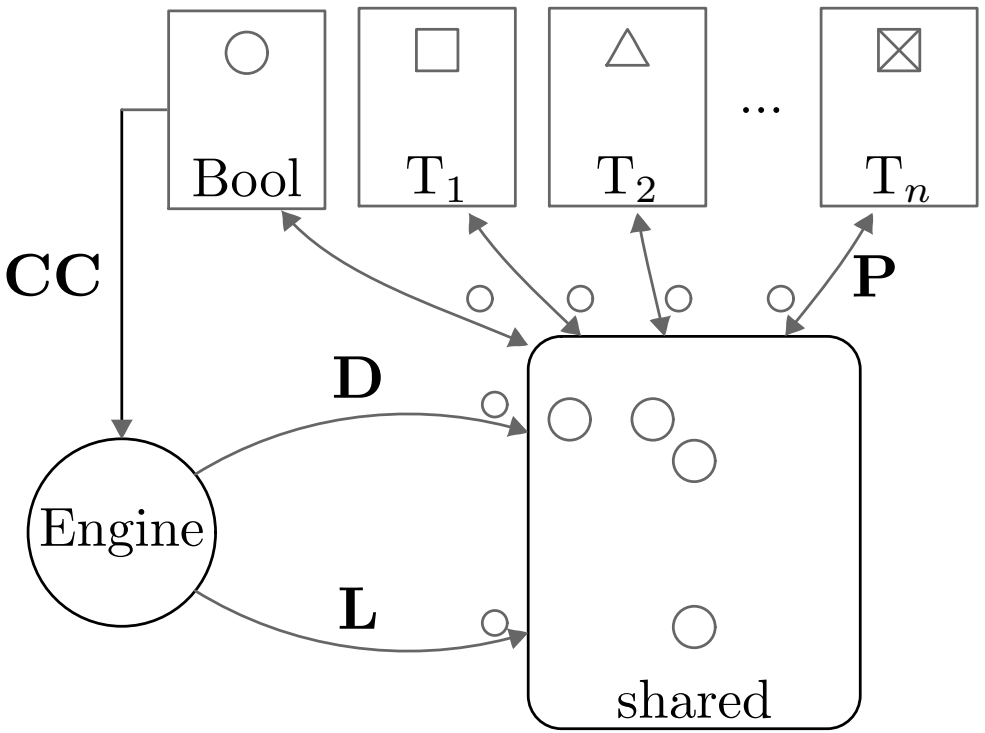
SMT-solver: Big picture and CDCL



$$\left(\begin{array}{c} 10 \leq x + t \vee 10 \leq y \\ \Longrightarrow\ 10 \leq z \end{array}\right)$$
$$\wedge\,(10 \leq z \implies t < 0)$$

$$(x \leq 0)$$

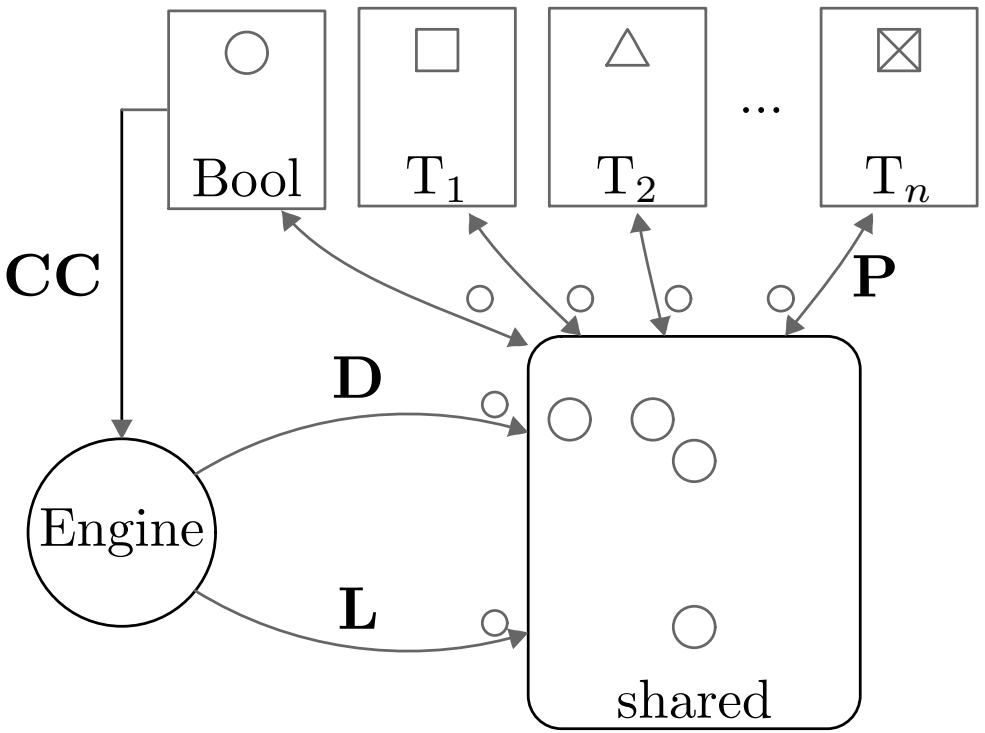
SMT-solver: Big picture and CDCL



$$\left(\begin{array}{c} 10 \leq x + t \vee 10 \leq y \\ \Longrightarrow\ 10 \leq z \end{array}\right)$$
$$\wedge\,(10 \leq z \implies t < 0)$$

$$\frac{\vdots}{(x \leq 0)}$$

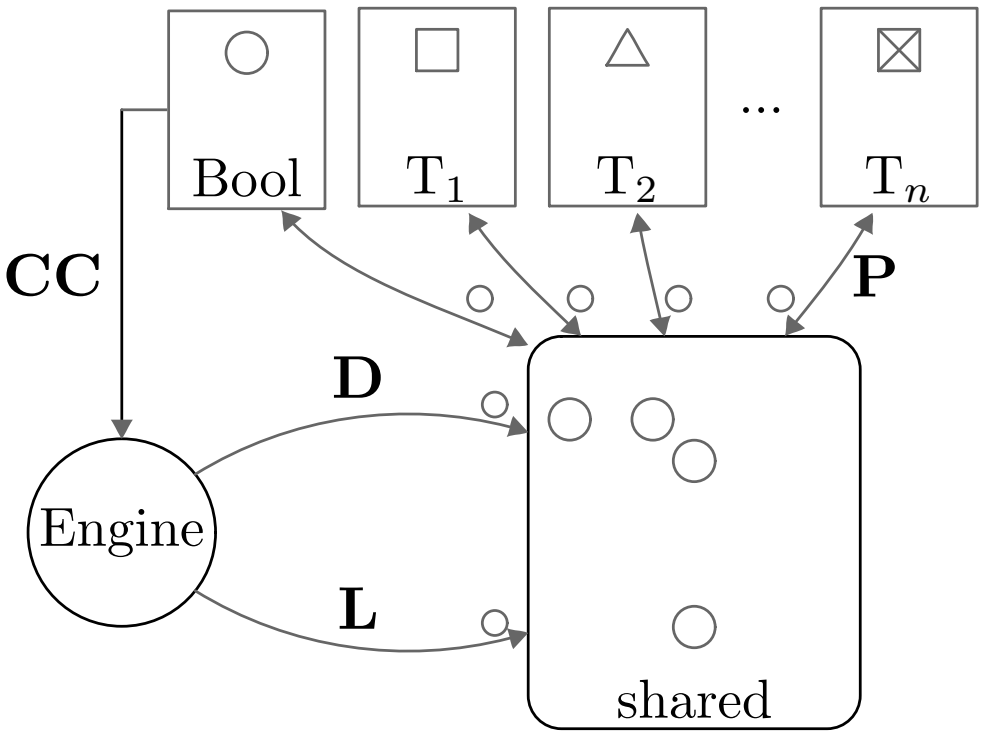
SMT-solver: Big picture and CDCL



$$\left(\begin{array}{c} 10 \leq x + t \vee 10 \leq y \\ \Longrightarrow\ 10 \leq z \end{array}\right)$$
$$\wedge\,(10 \leq z \implies t < 0)$$

$$\frac{\begin{array}{c}(10 \leq x + t)_{\mathcal{D}} \\ \vdots \\ (x \leq 0)\end{array}}{}$$

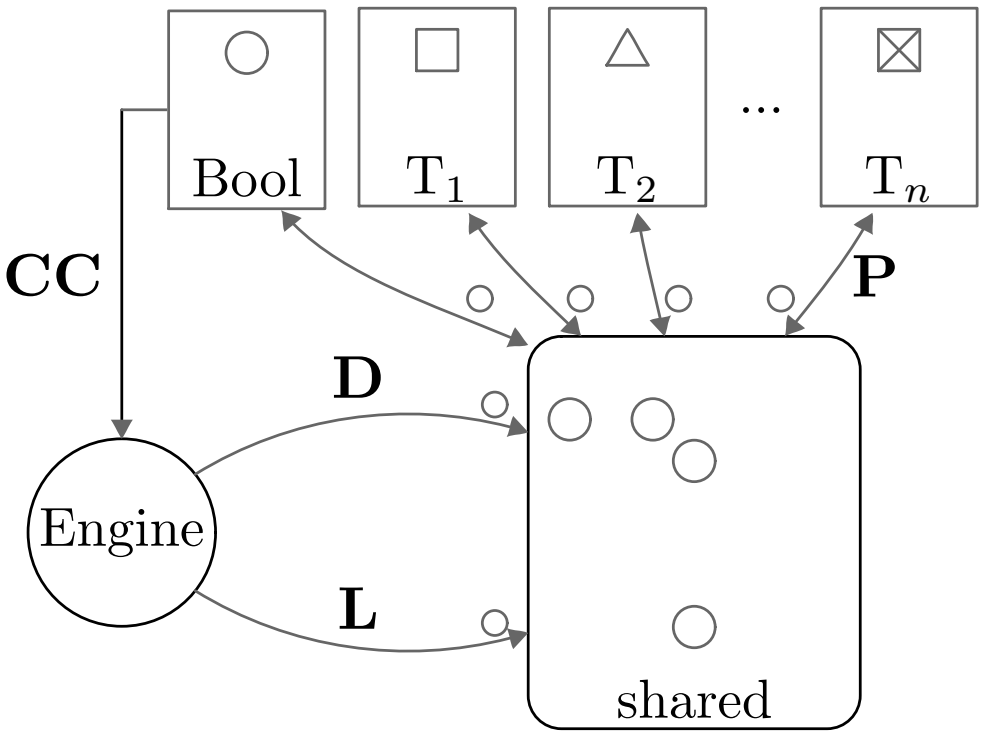
SMT-solver: Big picture and CDCL



$$\left(\begin{array}{c} 10 \leq x + t \vee 10 \leq y \\ \Longrightarrow\ 10 \leq z \end{array}\right) \wedge (10 \leq z \implies t < 0)$$

$$\frac{\begin{array}{c} 10 \leq z \\ (10 \leq x + t)_{\mathcal{D}} \\ \vdots \\ (x \leq 0) \end{array}}{}$$

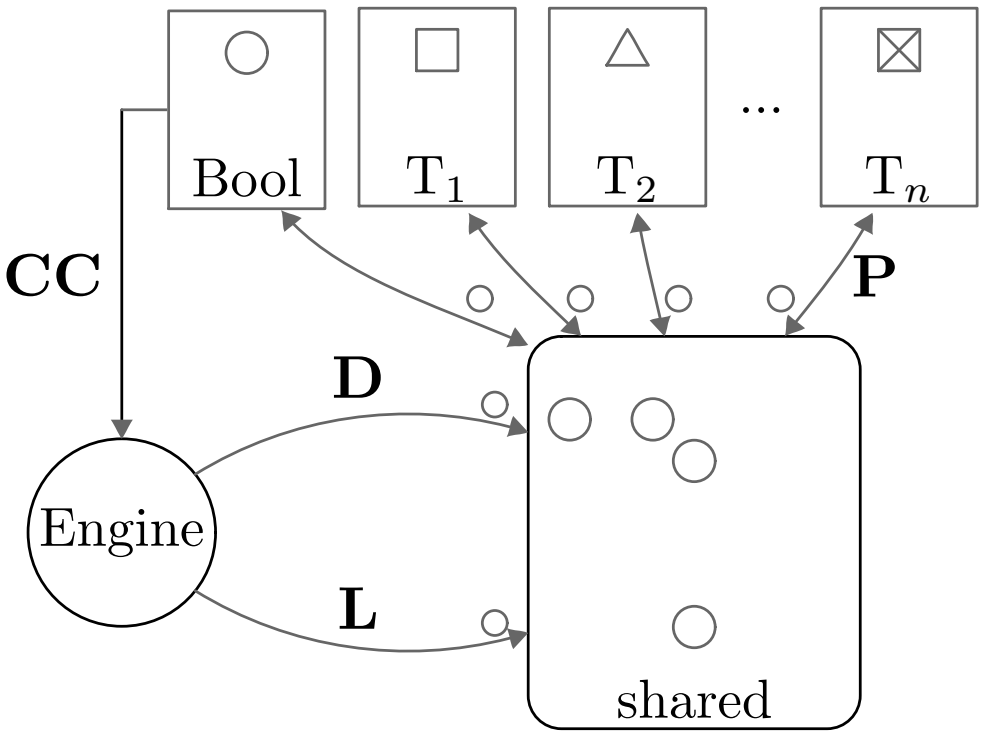
SMT-solver: Big picture and CDCL



$$\left(\begin{array}{c} 10 \leq x + t \vee 10 \leq y \\ \Longrightarrow\ 10 \leq z \end{array}\right)$$
$$\wedge\,(10 \leq z \implies t < 0)$$
$$\wedge\left(\begin{array}{c} t < 0 \wedge x \leq 0 \\ \Longrightarrow\ \neg 10 \leq x + t \end{array}\right)$$

$$\frac{\begin{array}{c} \neg(t < 0) \\ 10 \leq z \\ (10 \leq x + t)_{\mathcal{D}} \\ \vdots \\ (x \leq 0) \end{array}}{}$$

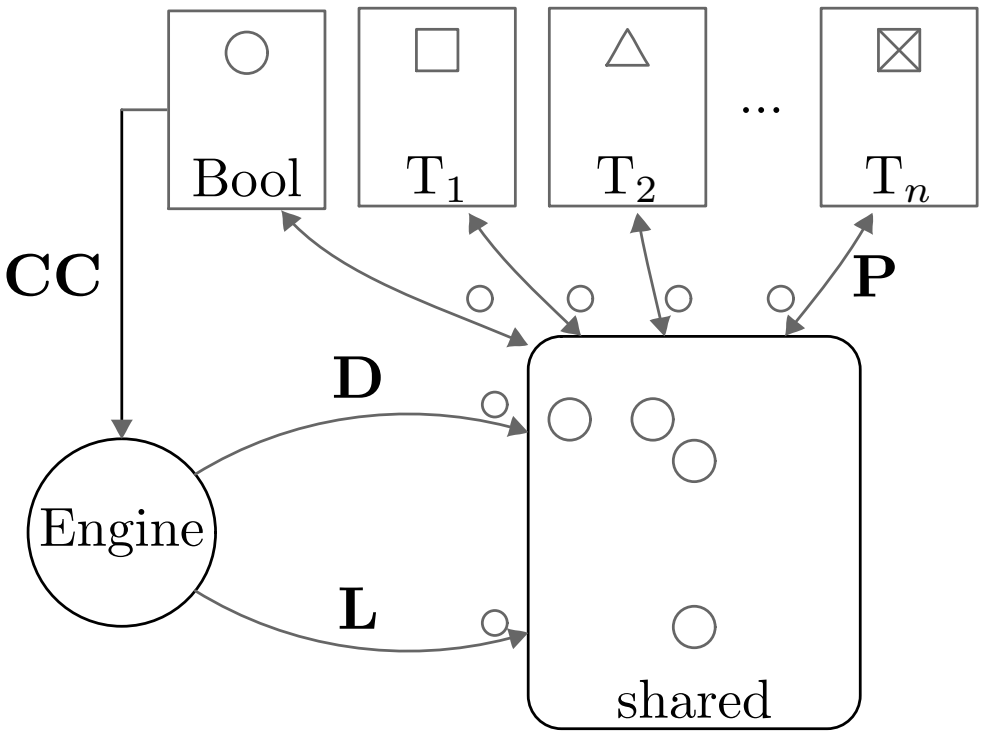
SMT-solver: Big picture and CDCL



$$\left(\begin{array}{c} 10 \leq x + t \vee 10 \leq y \\ \Longrightarrow\ 10 \leq z \end{array}\right)$$
$$\wedge\,(10 \leq z \implies t < 0)$$
$$\wedge\left(\begin{array}{c} t < 0 \wedge x \leq 0 \\ \Longrightarrow\ \neg 10 \leq x + t \end{array}\right)$$

$$\frac{\begin{array}{c} \neg(t < 0) \\ 10 \leq z \\ (10 \leq x + t)_{\mathcal{D}} \\ \vdots \end{array}}{(x \leq 0)}$$

SMT-solver: Big picture and CDCL



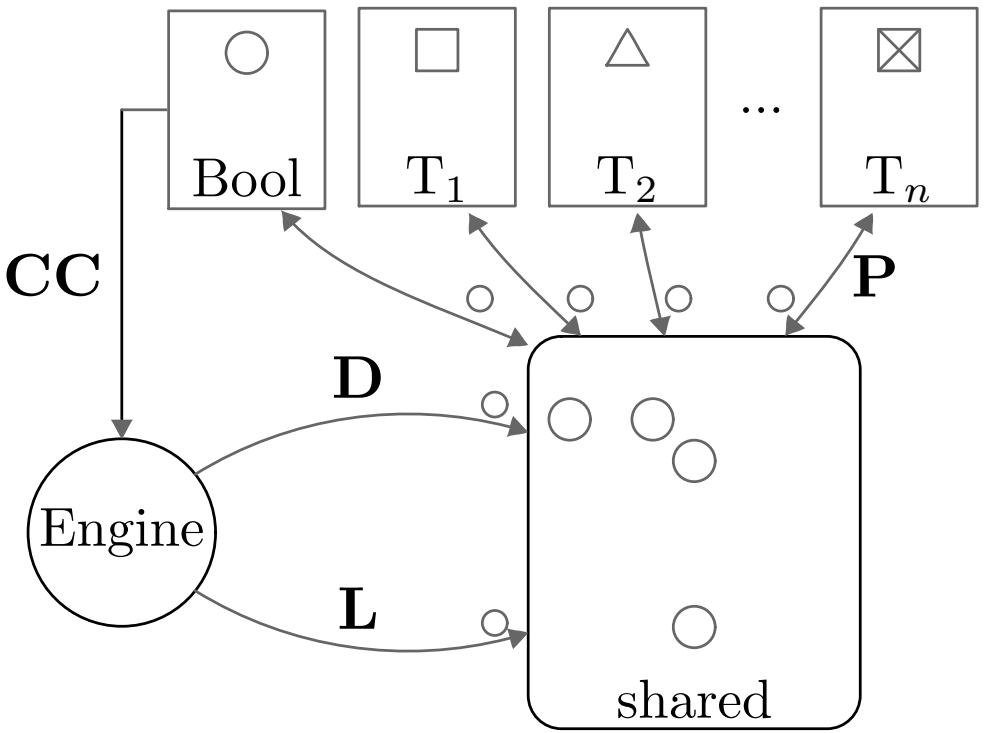
$$\left(\begin{array}{c} 10 \leq x + t \vee 10 \leq y \\ \Longrightarrow \; 10 \leq z \end{array}\right)$$

$$\wedge\,(10 \leq z \implies t < 0)$$

$$\wedge(\neg x \leq \; 0 \vee \neg 10 \leq x + t)$$

$$(x \leq 0)$$

SMT-solver: Big picture and CDCL



$$\left(\begin{array}{c} 10 \leq x + t \vee 10 \leq y \\ \Longrightarrow\ 10 \leq z \end{array}\right)$$

$$\wedge\,(10 \leq z \implies t < 0)$$

$$\wedge(\neg x \leq 10 \vee \neg 10 \leq x + t)$$

$$(x \leq 0)$$

SAT-solver: CDCL

- Decision: an unset variable is added to the partial model
- Propagation: propagate values using the constraints
- Conflict: a constraint is unsatisfiable in the partial model
- Learning: compute a constraint which:
  1. is implied by the constraints
  2. is unsatisfiable in the partial model
  3. disallows the last decision by propagation

SAT-solver: CDCL

- Decision: an unset variable is added to the partial model
- Propagation: propagate values using the constraints
- Conflict: a constraint is unsatisfiable in the partial model
- Learning: compute a constraint which:
  1. is implied by the constraints
  2. is unsatisfiable in the partial model
  3. disallows the last decision by propagation

SAT-solver: CDCL

- Decision: an unset variable is added to the partial model
- Propagation: propagate values using the constraints
- Conflict: a constraint is unsatisfiable in the partial model
- Learning: compute a constraint which:
  1. is implied by the constraints
  2. is unsatisfiable in the partial model
  3. disallows the last decision by propagation

SAT-solver: CDCL

- Decision: an unset variable is added to the partial model
- Propagation: propagate values using the constraints
- Conflict: a constraint is unsatisfiable in the partial model
- Learning: compute a constraint which:
  1. is implied by the constraints
  2. is unsatisfiable in the partial model
  3. disallows the last decision by propagation

SAT-solver: CDCL

- Decision: an unset variable is added to the partial model
- Propagation: propagate values using the constraints
- Conflict: a constraint is unsatisfiable in the partial model
- Learning: compute a constraint which:
  1. is implied by the constraints
  2. is unsatisfiable in the partial model
  3. disallows the last decision by propagation

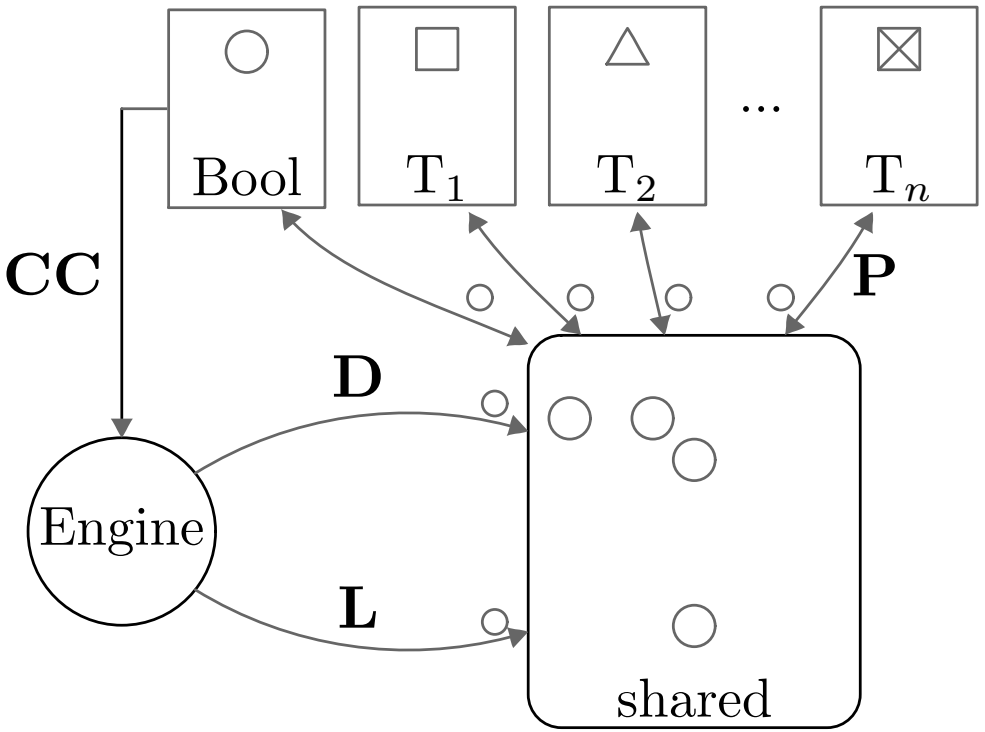
SAT-solver: CDCL

- Decision: an unset variable is added to the partial model
- Propagation: propagate values using the constraints
- Conflict: a constraint is unsatisfiable in the partial model
- Learning: compute a constraint which:
  1. is implied by the constraints
  2. is unsatisfiable in the partial model
  3. disallows the last decision by propagation

SAT-solver: CDCL

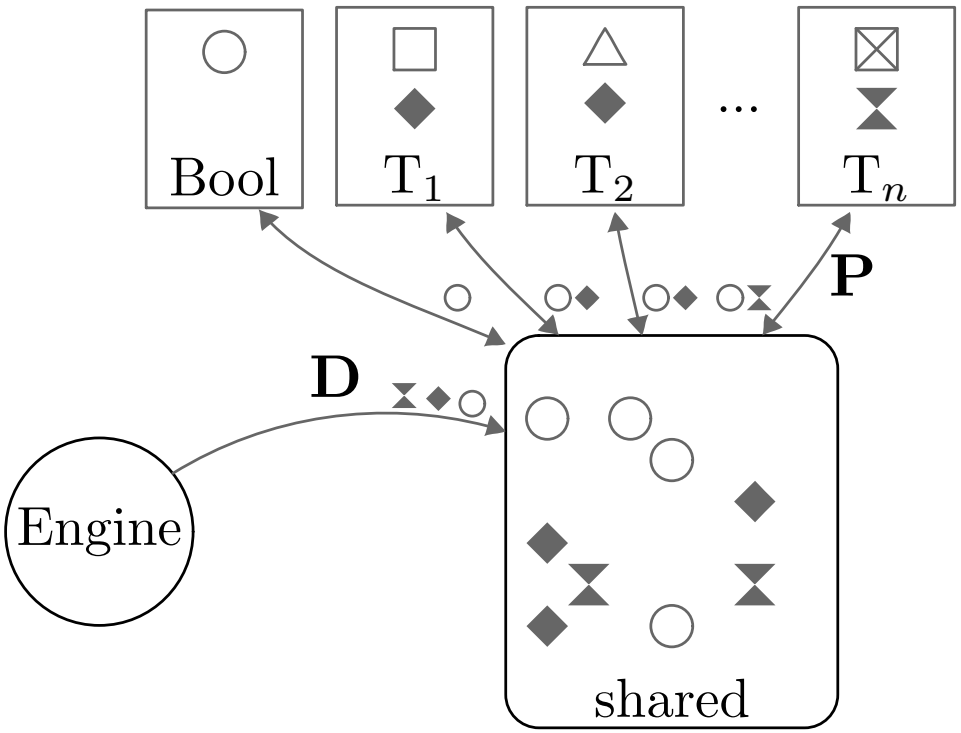
- Decision: an unset variable is added to the partial model
- Propagation: propagate values using the constraints
- Conflict: a constraint is unsatisfiable in the partial model
- Learning: compute a constraint which:
  1. is implied by the constraints
  2. is unsatisfiable in the partial model
  3. disallows the last decision by propagation

SMT-solver: Difficulties



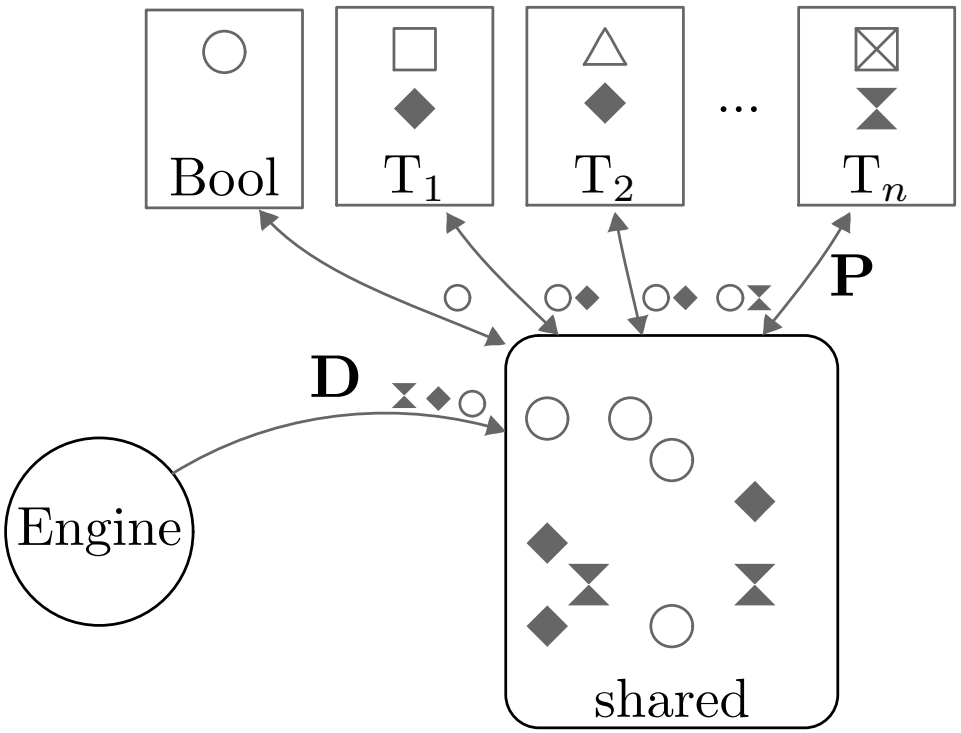
- communication using only booleans
- split-on-demand: $x \leq 11 \vee 11 < x$
- learning: no new literals

Big picture



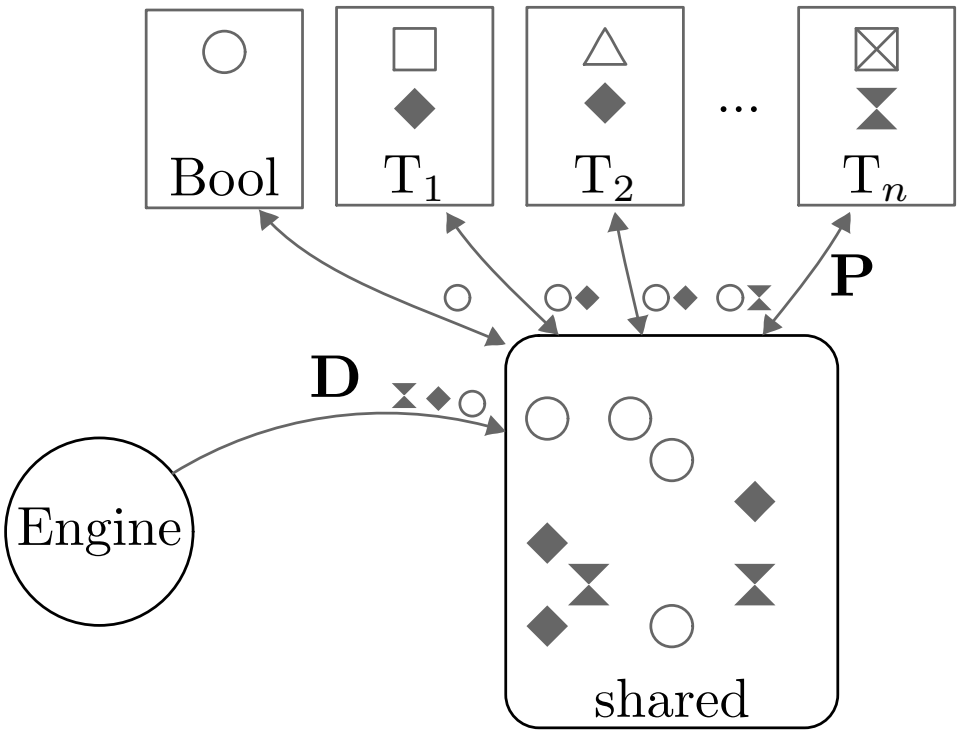
- Specific domains
- Precise and numerous propagator
- Easy model computation

Big picture



- Specific domains
- Precise and numerous propagator
- Easy model computation

Big picture



- Specific domains
- Precise and numerous propagator
- Easy model computation

Propagation and domains

- arithmetic intervals: $\mathcal{D}_I(x) = [\![1; 3]\!]$ and $\mathcal{D}_I(x + y) = [\![2; 4]\!]$ implies $\mathcal{D}_I(y) = [\![-1; 3]\!]$
- intervals and modulos: reduce product (AI)
- intervals and arrays: $\text{select}(x, \text{store}(\text{store}(\text{store}(\text{const}(4), y_3, 3), y_2, 2), y_1, 1)) = t$ $\mathcal{D}_I(t) = [\![1; 4]\!]$
- bitvectors and arithmetic intervals: $\mathcal{D}_I(x) = [\![0; +\infty[$ implies $\mathcal{D}_I(x \mid 31) = [\![2^{32}; +\infty]\!]$
- floating-points

Propagation and domains

- arithmetic intervals: $\mathcal{D}_I(x) = [\![1; 3]\!]$ and $\mathcal{D}_I(x + y) = [\![2; 4]\!]$ implies $\mathcal{D}_I(y) = [\![-1; 3]\!]$
- intervals and modulos: reduce product (AI)
- intervals and arrays: $\text{select}(x, \text{store}(\text{store}(\text{store}(\text{const}(4), y_3, 3), y_2, 2), y_1, 1)) = t$ $\mathcal{D}_I(t) = [\![1; 4]\!]$
- bitvectors and arithmetic intervals: $\mathcal{D}_I(x) = [\![0; +\infty[$ implies $\mathcal{D}_I(x \mid 31) = [\![2^{32}; +\infty]\!]$
- floating-points

Propagation and domains

- arithmetic intervals: $\mathcal{D}_I(x) = [\![1; 3]\!]$ and $\mathcal{D}_I(x + y) = [\![2; 4]\!]$ implies $\mathcal{D}_I(y) = [\![-1; 3]\!]$
- intervals and modulos: reduce product (AI)
- intervals and arrays:
  $\text{select}(x, \text{store}(\text{store}(\text{store}(\text{const}(4), y_3, 3), y_2, 2), y_1, 1)) = t$
  $\mathcal{D}_I(t) = [\![1; 4]\!]$
- bitvectors and arithmetic intervals: $\mathcal{D}_I(x) = [\![0; +\infty[$ implies $\mathcal{D}_I(x \mid 31) = [\![2^{32}; +\infty]\!]$
- floating-points

Propagation and domains

- arithmetic intervals: $\mathcal{D}_I(x) = [\![1; 3]\!]$ and $\mathcal{D}_I(x + y) = [\![2; 4]\!]$ implies $\mathcal{D}_I(y) = [\![-1; 3]\!]$
- intervals and modulos: reduce product (AI)
- intervals and arrays:
  $\text{select}(x, \text{store}(\text{store}(\text{store}(\text{const}(4), y_3, 3), y_2, 2), y_1, 1)) = t$
  $\mathcal{D}_I(t) = [\![1; 4]\!]$
- bitvectors and arithmetic intervals: $\mathcal{D}_I(x) = [\![0; +\infty[$ implies $\mathcal{D}_I(x \mid 31) = [\![2^{32}; +\infty]\!]$
- floating-points

Propagation and domains

- arithmetic intervals: $\mathcal{D}_I(x) = [\![1; 3]\!]$ and $\mathcal{D}_I(x + y) = [\![2; 4]\!]$ implies $\mathcal{D}_I(y) = [\![-1; 3]\!]$
- intervals and modulos: reduce product (AI)
- intervals and arrays:
  $\text{select}(x, \text{store}(\text{store}(\text{store}(\text{const}(4), y_3, 3), y_2, 2), y_1, 1)) = t$
  $\mathcal{D}_I(t) = [\![1; 4]\!]$
- bitvectors and arithmetic intervals: $\mathcal{D}_I(x) = [\![0; +\infty[$ implies $\mathcal{D}_I(x \mid 31) = [\![2^{32}; +\infty]\!]$
- floating-points

Global constraints

- DBM: Difference Constraint
- Simplex when slow convergence appears
- ...

Global constraints

- DBM: Difference Constraint
- Simplex when slow convergence appears
- ...

Global constraints

- DBM: Difference Constraint
- Simplex when slow convergence appears
- ...

Difficulties

- No backjumping
- No learning
- Costly propagation

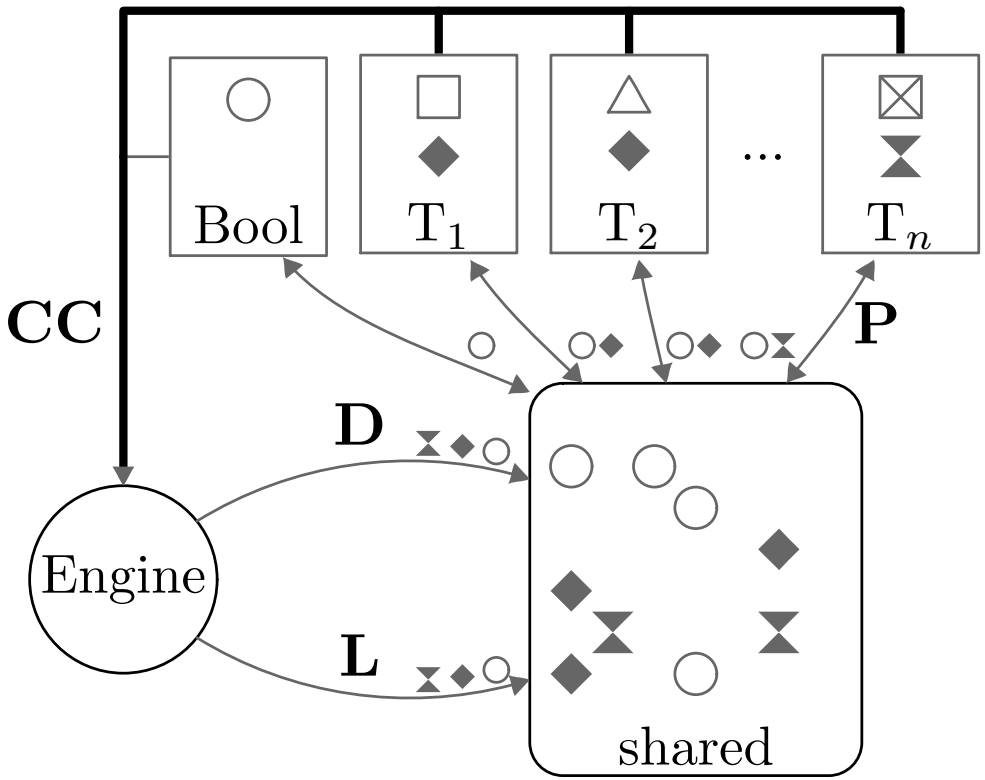
Difficulties

- No backjumping
- No learning
- Costly propagation

Difficulties

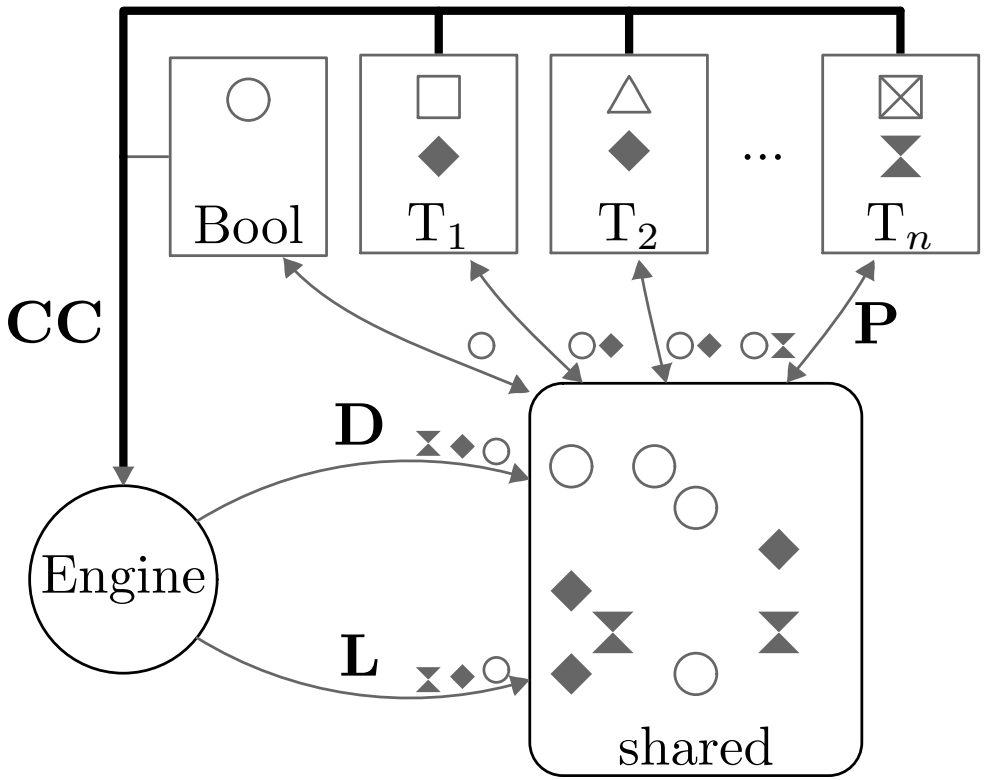
- No backjumping
- No learning
- Costly propagation

Ideas[Generalisation of de Moura (2013)]



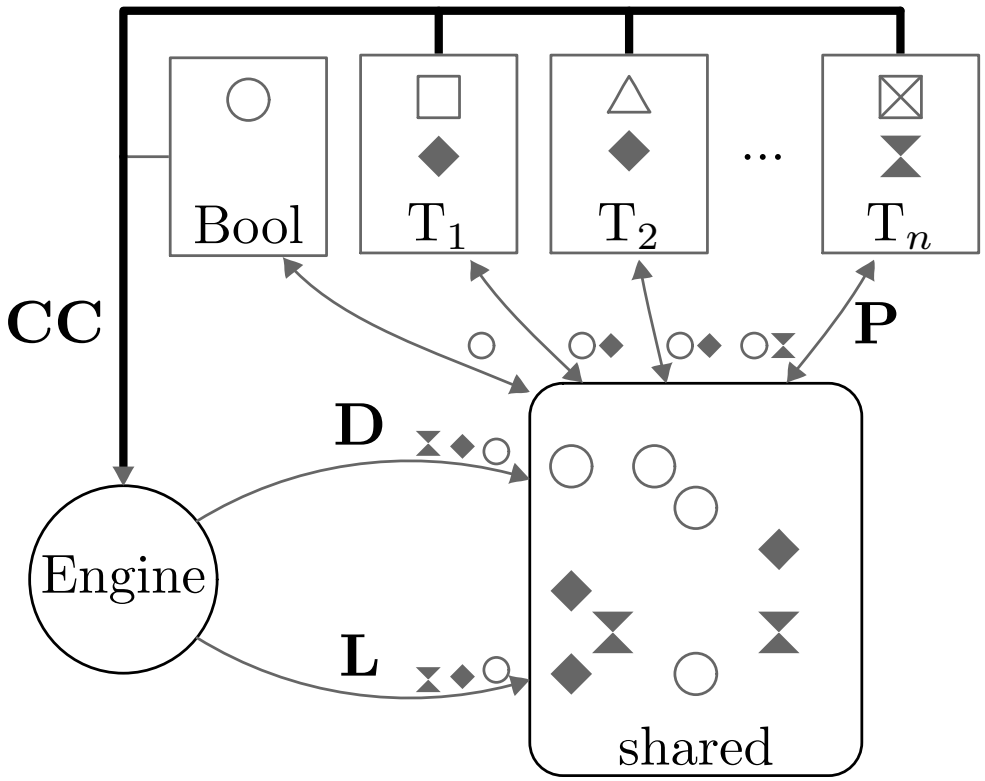
- Can decide on every variables
- Propagate using specific domains
- Learn new constraints

Ideas[Generalisation of de Moura (2013)]



- Can decide on every variables
- Propagate using specific domains
- Learn new constraints

Ideas[Generalisation of de Moura (2013)]



- Can decide on every variables
- Propagate using specific domains
- Learn new constraints

Example: Arithmetic

$$10 \leq x - y \quad \wedge \quad 10 < x + y + z$$

1. Decide $x = 0$
2. Propagate using $x - y$
3. Conflict in $x + y + z$
4. Compute clause conflict
5. Propagate

	$\mathcal{D}$	*reason*
x	$]-\infty;+\infty[$	
y	$]-\infty;+\infty[$	
z	$]-\infty;0]$	...
x-y	$[10;+\infty[$	
x+y+z	$]10;+\infty[$	

Example: Arithmetic

$$10 \leq x - y \quad \wedge \quad 10 < x + y + z$$

1. Decide $x = 0$
2. Propagate using $x - y$
3. Conflict in $x + y + z$
4. Compute clause conflict
5. Propagate

	$\mathcal{D}$	*reason*
x	$]-\infty;+\infty[$	
y	$]-\infty;+\infty[$	
z	$]-\infty;0]$	...
x-y	$[10;+\infty[$	
x+y+z	$]10;+\infty[$	

Example: Arithmetic

$$10 \leq x - y \quad \wedge \quad 10 < x + y + z$$

1. Decide $x = 0$
2. Propagate using $x - y$
3. Conflict in $x + y + z$
4. Compute clause conflict
5. Propagate

	$\mathcal{D}$	*reason*
x	$\{0\}$	*dec*
y	$]-\infty;+\infty[$	
z	$]-\infty;0]$	...
x-y	$[10;+\infty[$	
x+y+z	$]10;+\infty[$	

Example: Arithmetic

$$10 \leq x - y \quad \wedge \quad 10 < x + y + z$$

1. Decide $x = 0$
2. Propagate using $x - y$
3. Conflict in $x + y + z$
4. Compute clause conflict
5. Propagate

	$\mathcal{D}$	*reason*
x	$\{0\}$	*dec*
y	$]-\infty; -10]$	$x - y$
z	$]-\infty; 0]$	...
x-y	$[10; +\infty[$	
x+y+z	$]10; +\infty[$	

Example: Arithmetic

$$10 \leq x - y \quad \wedge \quad 10 < x + y + z$$

1. Decide $x = 0$
2. Propagate using $x - y$
3. Conflict in $x + y + z$
4. Compute clause conflict
5. Propagate

	$\mathcal{D}$	*reason*
x	$\{0\}$	*dec*
y	$]-\infty;-10]$	$x - y$
z	$]-\infty;0]$	...
x-y	$[10;+\infty[$	
x+y+z	$]10;+\infty[$	

Example: Arithmetic

$$10 \leq x - y \quad \wedge \quad 10 < x + y + z$$

1. Decide $x = 0$
2. Propagate using $x - y$
3. Conflict in $x + y + z$
4. Compute clause conflict
5. Propagate

	$\mathcal{D}$	*reason*
x	$\{0\}$	*dec*
y	$]-\infty;-10]$	$x - y$
z	$]-\infty;0]$	...
x-y	$[10;+\infty[$	
x+y+z	$]10;+\infty[$	

$$\dfrac{\dfrac{\dfrac{\texttt{dec}}{0 \leq -x} \quad \dfrac{\texttt{level0}}{10 \leq x - y}}{10 \leq -y} \quad \dfrac{\texttt{dec}}{0 \leq -x} \quad \dfrac{\vdots}{0 \leq -z} \quad \dfrac{\texttt{level0}}{10 < x + y + z}}{20 < 0}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{-x \leq -x}\,\texttt{gen} \quad \dfrac{}{10 \leq x - y}\,\texttt{level0}}{-x + 10 \leq -y} \quad \dfrac{\dfrac{}{-x \leq -x}\,\texttt{gen} \quad \dfrac{}{-z \leq -z}\,\texttt{gen}}{}\quad \dfrac{}{10 < x + y + }\,\texttt{level0}}{-z - 2x + 20 < 0}$$

Example: Arithmetic

$$10 \leq x - y \quad \wedge \quad 10 < x + y + z$$

1. Decide $x = 0$
2. Propagate using $x - y$
3. Conflict in $x + y + z$
4. Compute clause conflict
5. Propagate

	$\mathcal{D}$	*reason*
x	$\{0\}$	*dec*
y	$]-\infty; -10]$	$x - y$
z	$]-\infty; 0]$	...
x-y	$[10; +\infty[$	
x+y+z	$]10; +\infty[$	

Example: Arithmetic

$$10 \leq x - y \quad \wedge \quad 10 < x + y + z$$

1. Decide $x = 0$
2. Propagate using $x - y$
3. Conflict in $x + y + z$
4. Compute clause conflict
5. Propagate

	$\mathcal{D}$	*reason*
x	$\{0\}$	*dec*
y	$]-\infty; -10]$	$x - y$
z	$]-\infty; 0]$	...
x-y	$[10; +\infty[$	
x+y+z	$]10; +\infty[$	
2x+z	$[20; +\infty[$	

Example: Arithmetic

$$10 \leq x - y \quad \wedge \quad 10 < x + y + z$$

1. Decide $x = 0$
2. Propagate using $x - y$
3. Conflict in $x + y + z$
4. Compute clause conflict
5. Propagate

	$\mathcal{D}$	*reason*
x	$]10; +\infty]$	$2x + z$
y	$]-\infty; +\infty[$	
z	$]-\infty; 0]$	...
x-y	$[10; +\infty[$	
x+y+z	$]10; +\infty[$	
2x+z	$[20; +\infty[$	

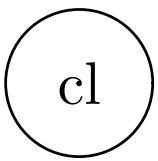
Popop

`git@git.frama-c.com:soprano/popop.git`

`specifically written: 8417 code, 1409 comment`

`(reuse Why3 STDLIB, alt-ergo parser, smtlib2 parser)`

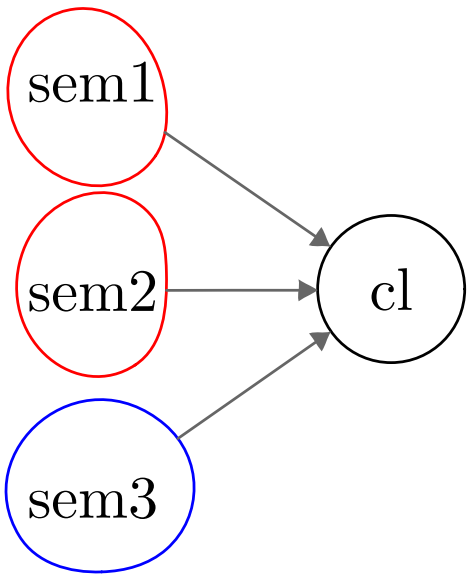
Framework: propagation



kind of semantic values:

- boolean formula $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$
- rational polynomial $4cl_1 + 3cl_2 + 5$
- equality $cl_1 = cl_2 = cl_3 = cl_4$
- ite $ite(b, cl_1, cl2)$
- uninterpreted symbols $App(cl_1, cl_2)$

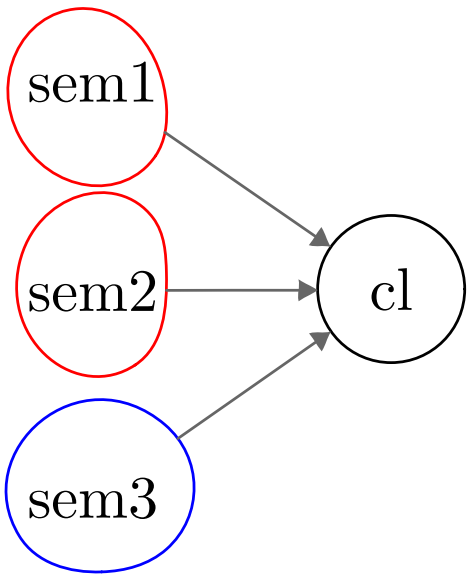
Framework: propagation



kind of semantic values:

- boolean formula $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$
- rational polynomial $4cl_1 + 3cl_2 + 5$
- equality $cl_1 = cl_2 = cl_3 = cl_4$
- ite $ite(b, cl_1, cl2)$
- uninterpreted symbols $App(cl_1, cl_2)$

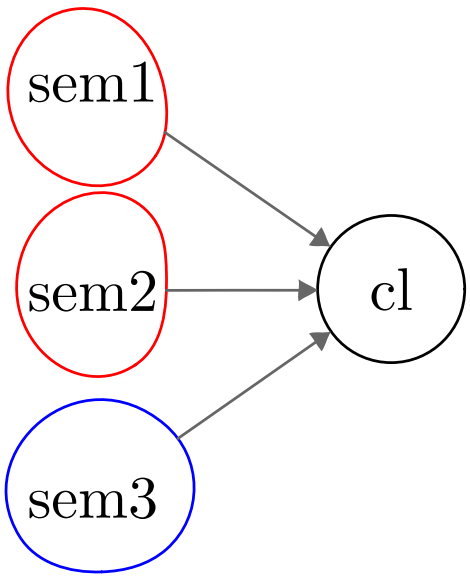
Framework: propagation



kind of semantic values:

- boolean formula $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$
- rational polynomial $4cl_1 + 3cl_2 + 5$
- equality $cl_1 = cl_2 = cl_3 = cl_4$
- ite $ite(b, cl_1, cl2)$
- uninterpreted symbols $App(cl_1, cl_2)$

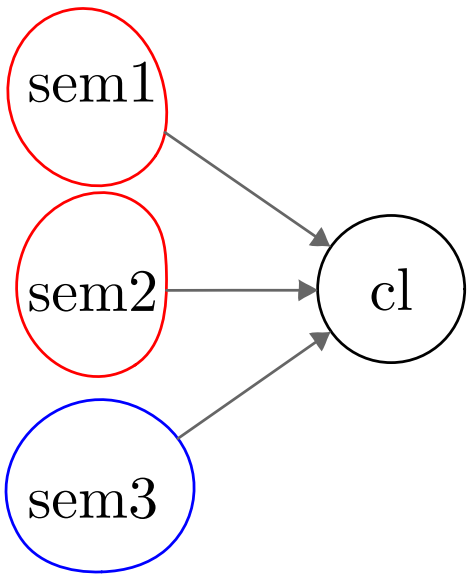
Framework: propagation



kind of semantic values:

- boolean formula $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$
- rational polynomial $4cl_1 + 3cl_2 + 5$
- equality $cl_1 = cl_2 = cl_3 = cl_4$
- ite $ite(b, cl_1, cl2)$
- uninterpreted symbols $App(cl_1, cl_2)$

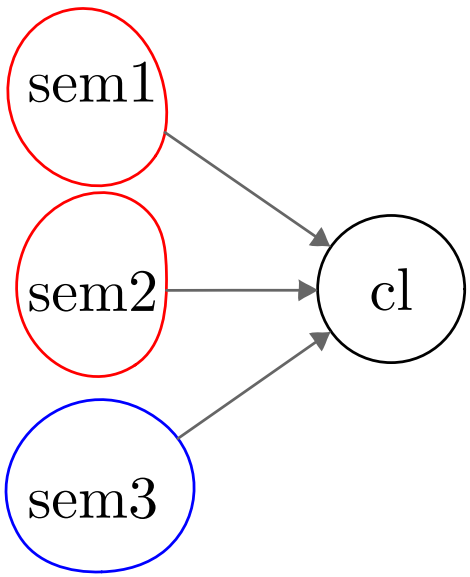
Framework: propagation



kind of semantic values:

- boolean formula $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$
- rational polynomial $4cl_1 + 3cl_2 + 5$
- equality $cl_1 = cl_2 = cl_3 = cl_4$
- ite $ite(b, cl_1, cl2)$
- uninterpreted symbols $App(cl_1, cl_2)$

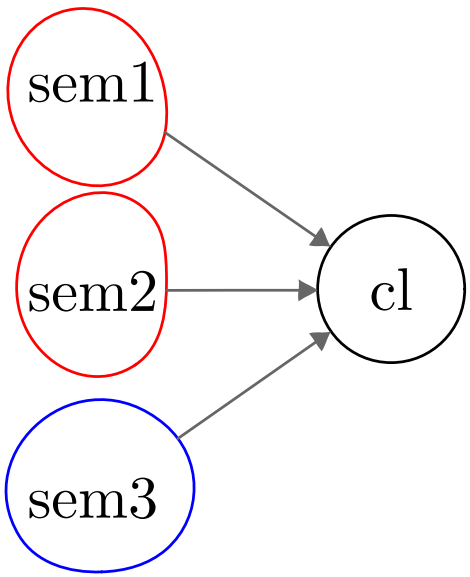
Framework: propagation



kind of semantic values:

- boolean formula $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$
- rational polynomial $4cl_1 + 3cl_2 + 5$
- equality $cl_1 = cl_2 = cl_3 = cl_4$
- ite $ite(b, cl_1, cl2)$
- uninterpreted symbols $App(cl_1, cl_2)$

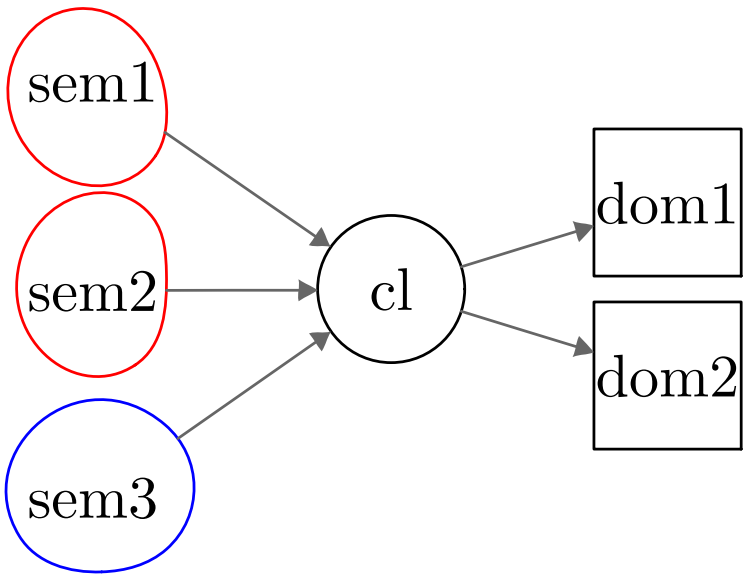
Framework: propagation



kind of semantic values:

- boolean formula $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$
- rational polynomial $4cl_1 + 3cl_2 + 5$
- equality $cl_1 = cl_2 = cl_3 = cl_4$
- ite $ite(b, cl_1, cl2)$
- uninterpreted symbols $App(cl_1, cl_2)$

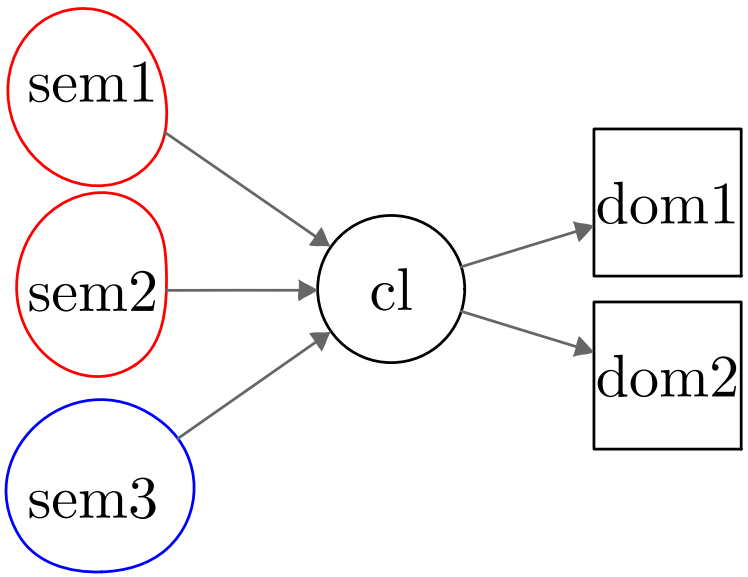
Framework: propagation



kind of domain values:

- boolean $\{\top, \bot\}$
- distinct: $\{1; 9; 10\}$
- normalized polynomial $4cl_1 + 3cl_2 + 5$
- interval rational $[a; b]$

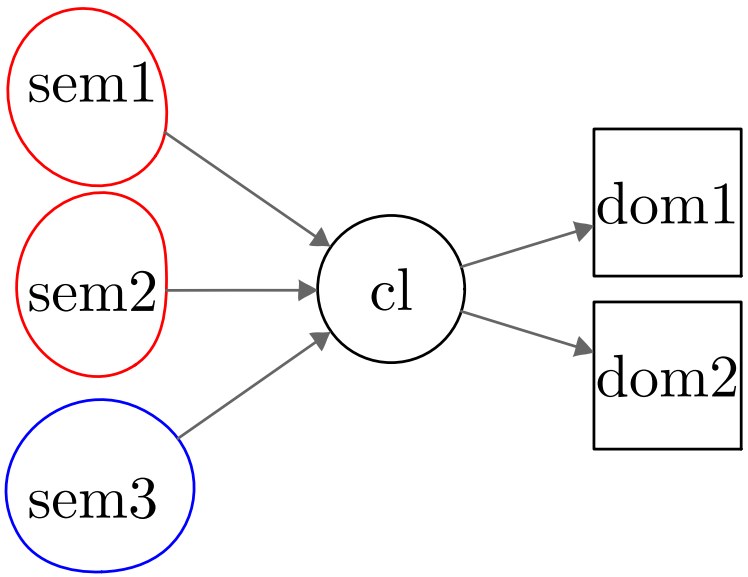
Framework: propagation



kind of domain values:

- boolean $\{\top, \bot\}$
- distinct: $\{1; 9; 10\}$
- normalized polynomial $4cl_1 + 3cl_2 + 5$
- interval rational $[a; b]$

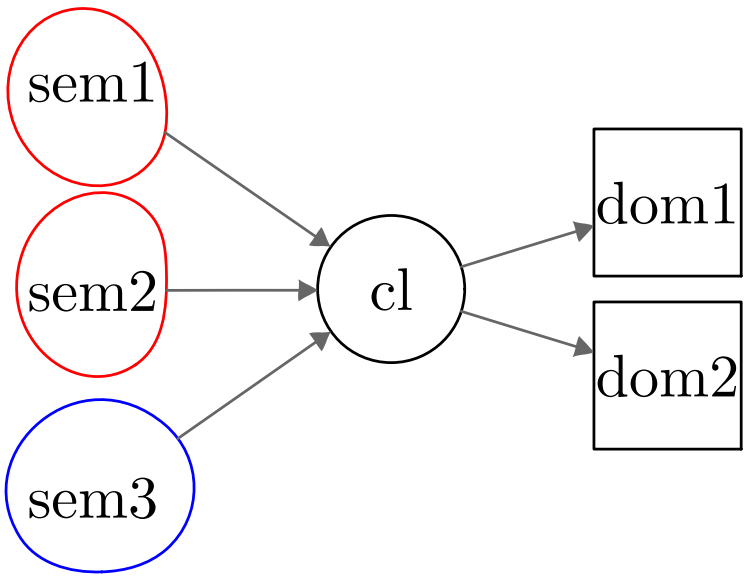
Framework: propagation



kind of domain values:

- boolean $\{\top, \bot\}$
- distinct: $\{1; 9; 10\}$
- normalized polynomial $4cl_1 + 3cl_2 + 5$
- interval rational $[a; b]$

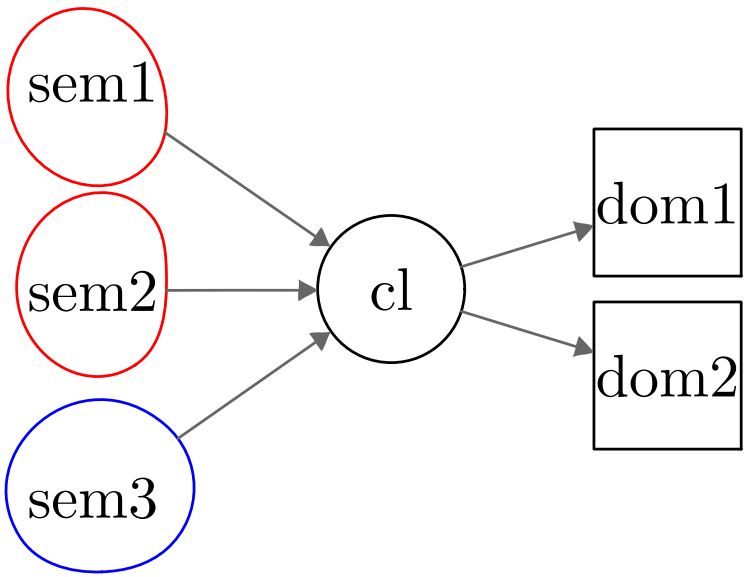
Framework: propagation



kind of domain values:

- boolean $\{\top, \bot\}$
- distinct: $\{1; 9; 10\}$
- normalized polynomial $4cl_1 + 3cl_2 + 5$
- interval rational $[a; b]$

Framework: propagation



kind of domain values:

- boolean $\{\top, \bot\}$
- distinct: $\{1; 9; 10\}$
- normalized polynomial $4cl_1 + 3cl_2 + 5$
- interval rational $[a; b]$

Sémantique Value

src/types.mli:

```
module Sem: sig
  type 'a k
  val create_key: string -> 'a k
  ...
end

type 'a sem = 'a Sem.k

module type Sem = sig
  include Datatype

  val key: t sem
end
```

```
module RegisterSem (D:Sem) : sig
  include Datatype

  val index: D.t -> Ty.t -> t

  val cl: t -> Cl.t

  val sem: t -> D.t

  ...
end
```

Domains

src/solver.mli:

```
module type Dom = sig
  type t

  val merged: t option -> t option -> bool

  val merge:
    Delayed.t -> pexp ->
    t option * Cl.t (* cl1 *) ->
    t option * Cl.t (* cl2 *) ->
    bool (** true: cl1 will be repr otherwise it is
    unit

  val print: Format.formatter  -> t  -> unit
  val key: t dom
```

Framework: propagation

Internal scheduler:

1. Set value of domains
2. Set semantic values
3. Immediate daemons (ex: polynomial normalization)
4. Merge domains: `merge:t -> exp -> cl -> cl -> unit`
5. Finalize merge
6. Start a pending merge

External scheduler:

1. Daemons (boolean propagation, congruence closure,...)
2. Decisions

Framework: Events

- Domain change
- A class start to be used
- A semantic value is used for the first time
- A class is not anymore the representative class

Framework: Learning

- Choice: a possible decision (propositional variable in sat)
  - Booleans: propositional variable
  - Equality: equalities between uninterpreted class
  - Arithmetic: arithmetic variable
- Explanation: the constraint responsible for the modification
  - Booleans: $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$ and the kind of propagation
  - Equality: $cl_1 = cl_2 = cl_3 = cl_4$ and idem
  - Arithmetic pivot: linear combination
- Conflict:
  - Booleans: map $\{cl_1 \mapsto \top; cl_2 \mapsto \bot\}$
  - Equality: depend on the type
  - Arithmetic: polynomial.

Framework: Learning

- Choice: a possible decision (propositional variable in sat)
  - Booleans: propositional variable
  - Equality: equalities between uninterpreted class
  - Arithmetic: arithmetic variable
- Explanation: the constraint responsible for the modification
  - Booleans: $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$ and the kind of propagation
  - Equality: $cl_1 = cl_2 = cl_3 = cl_4$ and idem
  - Arithmetic pivot: linear combination
- Conflict:
  - Booleans: map $\{cl_1 \mapsto \top; cl_2 \mapsto \bot\}$
  - Equality: depend on the type
  - Arithmetic: polynomial.

Framework: Learning

- Choice: a possible decision (propositional variable in sat)
  - Booleans: propositional variable
  - Equality: equalities between uninterpreted class
  - Arithmetic: arithmetic variable
- Explanation: the constraint responsible for the modification
  - Booleans: $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$ and the kind of propagation
  - Equality: $cl_1 = cl_2 = cl_3 = cl_4$ and idem
  - Arithmetic pivot: linear combination
- Conflict:
  - Booleans: map $\{cl_1 \mapsto \top; cl_2 \mapsto \bot\}$
  - Equality: depend on the type
  - Arithmetic: polynomial.

Framework: Learning

- Choice: a possible decision (propositional variable in sat)
  - Booleans: propositional variable
  - Equality: equalities between uninterpreted class
  - Arithmetic: arithmetic variable
- Explanation: the constraint responsible for the modification
  - Booleans: $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$ and the kind of propagation
  - Equality: $cl_1 = cl_2 = cl_3 = cl_4$ and idem
  - Arithmetic pivot: linear combination
- Conflict:
  - Booleans: map $\{cl_1 \mapsto \top; cl_2 \mapsto \bot\}$
  - Equality: depend on the type
  - Arithmetic: polynomial.

Framework: Learning

- Choice: a possible decision (propositional variable in sat)
  - Booleans: propositional variable
  - Equality: equalities between uninterpreted class
  - Arithmetic: arithmetic variable
- Explanation: the constraint responsible for the modification
  - Booleans: $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$ and the kind of propagation
  - Equality: $cl_1 = cl_2 = cl_3 = cl_4$ and idem
  - Arithmetic pivot: linear combination
- Conflict:
  - Booleans: map $\{cl_1 \mapsto \top; cl_2 \mapsto \bot\}$
  - Equality: depend on the type
  - Arithmetic: polynomial.

Framework: Learning

- Choice: a possible decision (propositional variable in sat)
  - Booleans: propositional variable
  - Equality: equalities between uninterpreted class
  - Arithmetic: arithmetic variable
- Explanation: the constraint responsible for the modification
  - Booleans: $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$ and the kind of propagation
  - Equality: $cl_1 = cl_2 = cl_3 = cl_4$ and idem
  - Arithmetic pivot: linear combination
- Conflict:
  - Booleans: map $\{cl_1 \mapsto \top; cl_2 \mapsto \bot\}$
  - Equality: depend on the type
  - Arithmetic: polynomial.

Framework: Learning

- Choice: a possible decision (propositional variable in sat)
  - Booleans: propositional variable
  - Equality: equalities between uninterpreted class
  - Arithmetic: arithmetic variable
- Explanation: the constraint responsible for the modification
  - Booleans: $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$ and the kind of propagation
  - Equality: $cl_1 = cl_2 = cl_3 = cl_4$ and idem
  - Arithmetic pivot: linear combination
- Conflict:
  - Booleans: map $\{cl_1 \mapsto \top; cl_2 \mapsto \bot\}$
  - Equality: depend on the type
  - Arithmetic: polynomial.

Framework: Learning

- Choice: a possible decision (propositional variable in sat)
  - Booleans: propositional variable
  - Equality: equalities between uninterpreted class
  - Arithmetic: arithmetic variable
- Explanation: the constraint responsible for the modification
  - Booleans: $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$ and the kind of propagation
  - Equality: $cl_1 = cl_2 = cl_3 = cl_4$ and idem
  - Arithmetic pivot: linear combination
- Conflict:
  - Booleans: map $\{cl_1 \mapsto \top; cl_2 \mapsto \bot\}$
  - Equality: depend on the type
  - Arithmetic: polynomial.

Framework: Learning

- Choice: a possible decision (propositional variable in sat)
  - Booleans: propositional variable
  - Equality: equalities between uninterpreted class
  - Arithmetic: arithmetic variable
- Explanation: the constraint responsible for the modification
  - Booleans: $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$ and the kind of propagation
  - Equality: $cl_1 = cl_2 = cl_3 = cl_4$ and idem
  - Arithmetic pivot: linear combination
- Conflict:
  - Booleans: map $\{cl_1 \mapsto \top; cl_2 \mapsto \bot\}$
  - Equality: depend on the type
  - Arithmetic: polynomial.

Framework: Learning

- Choice: a possible decision (propositional variable in sat)
  - Booleans: propositional variable
  - Equality: equalities between uninterpreted class
  - Arithmetic: arithmetic variable
- Explanation: the constraint responsible for the modification
  - Booleans: $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$ and the kind of propagation
  - Equality: $cl_1 = cl_2 = cl_3 = cl_4$ and idem
  - Arithmetic pivot: linear combination
- Conflict:
  - Booleans: map $\{cl_1 \mapsto \top; cl_2 \mapsto \bot\}$
  - Equality: depend on the type
  - Arithmetic: polynomial.

Framework: Learning

- Choice: a possible decision (propositional variable in sat)
  - Booleans: propositional variable
  - Equality: equalities between uninterpreted class
  - Arithmetic: arithmetic variable
- Explanation: the constraint responsible for the modification
  - Booleans: $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$ and the kind of propagation
  - Equality: $cl_1 = cl_2 = cl_3 = cl_4$ and idem
  - Arithmetic pivot: linear combination
- Conflict:
  - Booleans: map $\{cl_1 \mapsto \top; cl_2 \mapsto \bot\}$
  - Equality: depend on the type
  - Arithmetic: polynomial.

Framework: Learning

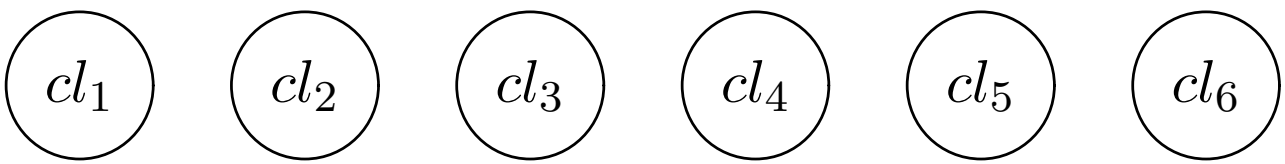
- Choice: a possible decision (propositional variable in sat)
  - Booleans: propositional variable
  - Equality: equalities between uninterpreted class
  - Arithmetic: arithmetic variable
- Explanation: the constraint responsible for the modification
  - Booleans: $\neg(cl_1 \vee \neg cl_2 \vee cl_3)$ and the kind of propagation
  - Equality: $cl_1 = cl_2 = cl_3 = cl_4$ and idem
  - Arithmetic pivot: linear combination
- Conflict:
  - Booleans: map $\{cl_1 \mapsto \top; cl_2 \mapsto \bot\}$
  - Equality: depend on the type
  - Arithmetic: polynomial.

Example in live

$$x = y - 1 \wedge f(2) = t \wedge y = 3 \implies f(x) = t$$

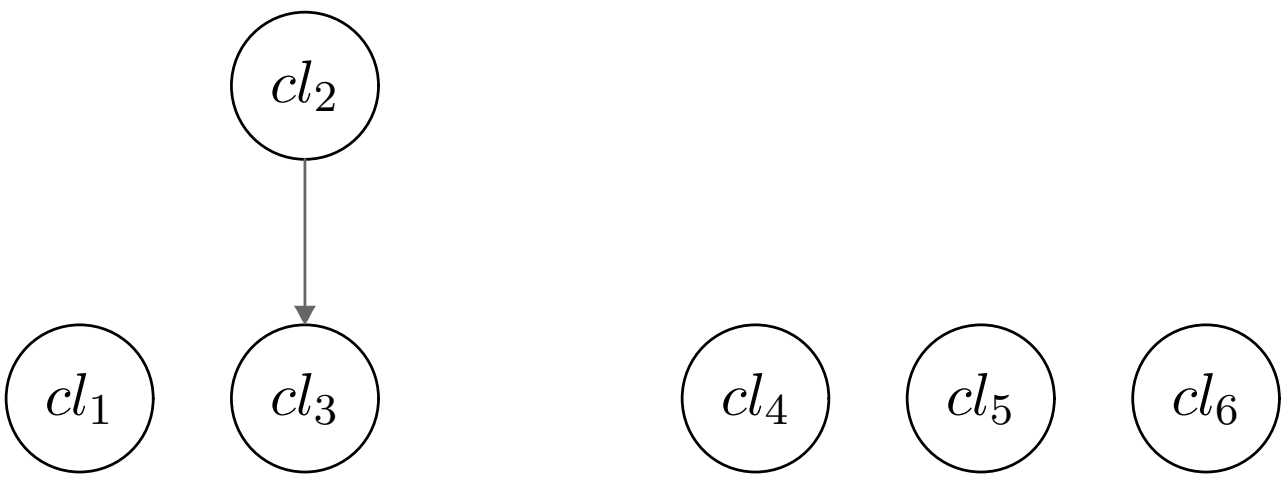
Representatives give bad explanation

$cl_2 = cl_3,\ cl_5 = cl_4,\ cl_5 = cl_3,\ cl_6 = cl_5$



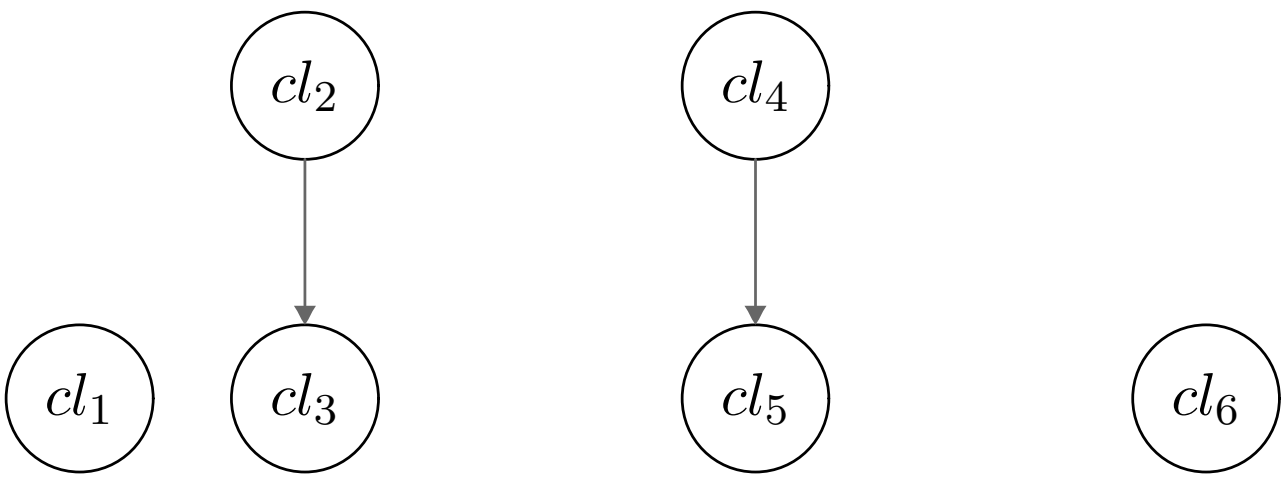
Representatives give bad explanation

$cl_2 = cl_3,\ cl_5 = cl_4,\ cl_5 = cl_3,\ cl_6 = cl_5$



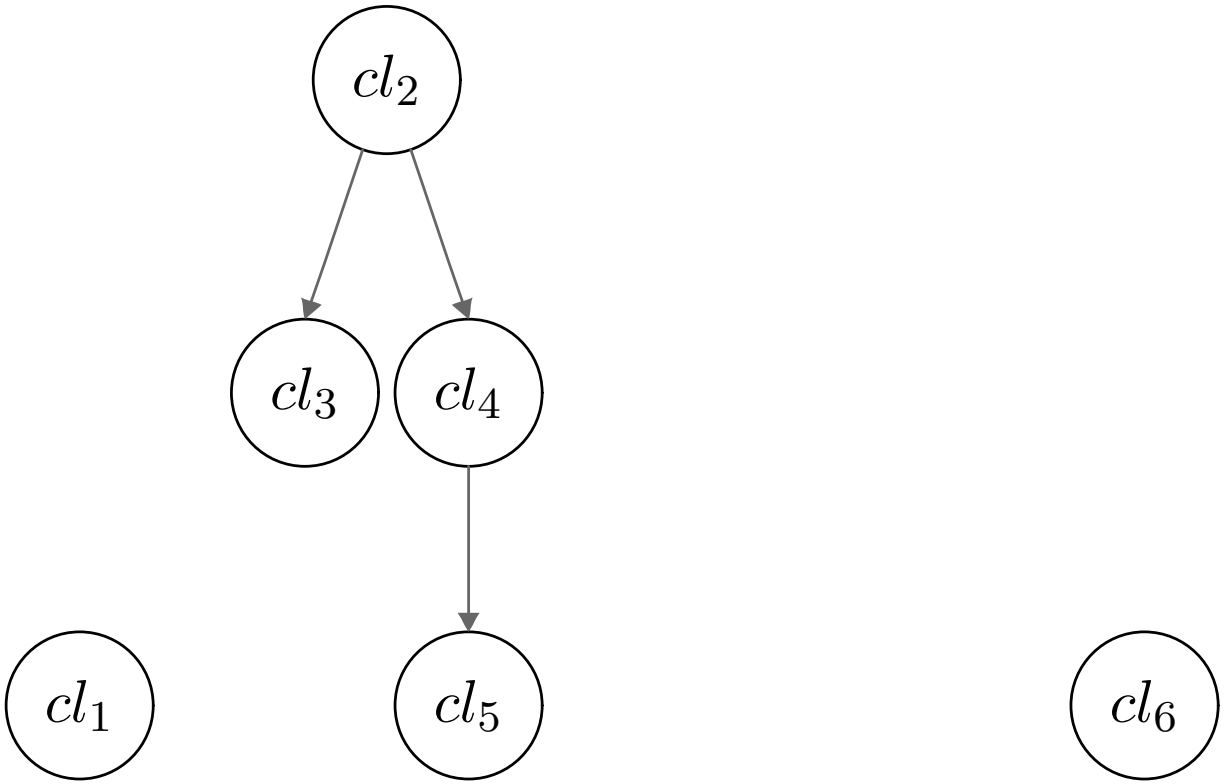
Representatives give bad explanation

$$cl_2 = cl_3,\ cl_5 = cl_4,\ cl_5 = cl_3,\ cl_6 = cl_5$$



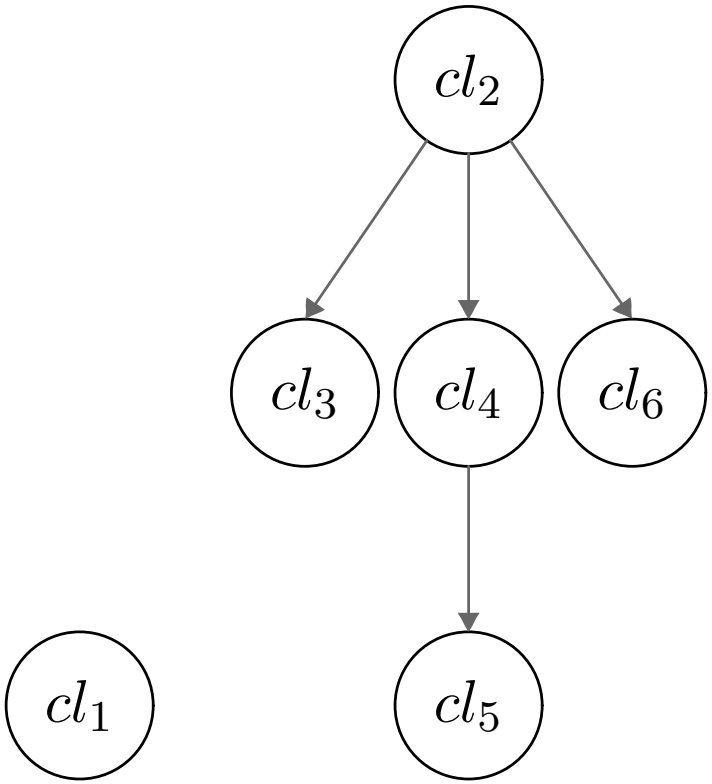
Representatives give bad explanation

$cl_2 = cl_3,\ cl_5 = cl_4,\ cl_5 = cl_3,\ cl_6 = cl_5$



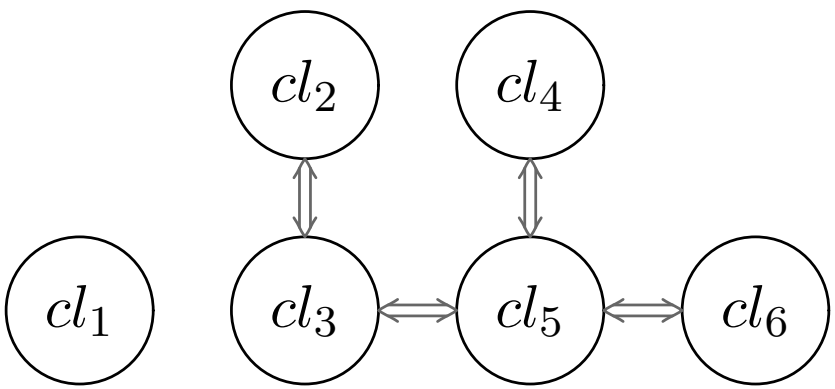
Representatives give bad explanation

$cl_2 = cl_3,\ cl_5 = cl_4,\ cl_5 = cl_3,\ cl_6 = cl_5$



Representatives give bad explanation

$$cl_2 = cl_3,\ cl_5 = cl_4,\ cl_5 = cl_3,\ cl_6 = cl_5$$



Future Work

many many many things:

1. linear arithmetic
2. quantifiers
3. bitvectors, algebraic datatypes
4. connect to Qed
5. SOPRANO
6. ...

Future Work

many many many things:

1. linear arithmetic
2. quantifiers
3. bitvectors, algebraic datatypes
4. connect to Qed
5. SOPRANO
6. ...

Future Work

many many many things:

1. linear arithmetic
2. quantifiers
3. bitvectors, algebraic datatypes
4. connect to Qed
5. SOPRANO
6. ...

Future Work

many many many things:

1. linear arithmetic
2. quantifiers
3. bitvectors, algebraic datatypes
4. connect to Qed
5. SOPRANO
6. ...

Future Work

many many many things:

1. linear arithmetic
2. quantifiers
3. bitvectors, algebraic datatypes
4. connect to Qed
5. SOPRANO
6. ...

Future Work

many many many things:

1. linear arithmetic
2. quantifiers
3. bitvectors, algebraic datatypes
4. connect to Qed
5. SOPRANO
6. ...

Related Work

MC-Sat implementation:

- Boolean a particular theory
- UF: generalized variable
- No sharing equivalence class
- Theoritical result

IntSat (CP 2014):

- bounded integer