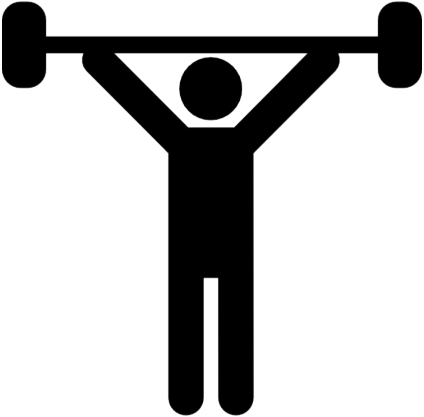
SCL: **C**lause **L**earning from **S**imple Models



Hi

Lifting CDCL to first-order logic

CDCL – quo vadis?

$$N = \{P \lor Q, \quad P \lor \neg Q, \quad \neg P \lor Q, \quad \neg P \lor \neg Q\}$$

State: $(T; N; U; k; D)$

$(\varepsilon; N; \emptyset; 0; \top)$

$\Rightarrow_{\text{Decide}} (P^1; N; \emptyset; 1; \top)$

$\Rightarrow_{\text{Propagate}} (P^1 Q^{\neg P \lor Q}; N; \emptyset; 1; \top)$

$\Rightarrow_{\text{Conflict}} (P \text{ '' } \quad \text{ '' } \quad \text{ '' }; \neg P \lor \neg Q)$

$\Rightarrow_{\text{Resolve}} (P^1; N; \emptyset; 1; \neg P \lor \neg P)$

$\Rightarrow_{\text{Factorize, Backtrack}} (\neg P^{1\,P}; N; \{\neg P\} \ldots)$

SCL Clause Learning from Simple Models

The basic idea of SCL is to lift the principles of CDCL, Section 2.9, to first-order logic:

1. operating wih respect to a partial model assumption represented by a trail,
2. learning only non-redundant clauses out of false clauses with respect to the trail,
3. finding models in case no conflict occurs.

It is called clause learning from simple models, because the trail is **restricted to ground literals.**

$\Gamma = P(x)\, R(x,y)$ ← do not this

$\Gamma = P(a)\, P(b)\, \neg R(a,b)$

SCL: *Simplified* Problem State

$$(\Gamma; N; U; k; D)$$

- Γ: Ground trail
- N: Initial clause set
- U: Learned clauses
- k: Level $\longleftrightarrow$ CDCL
- D: State
  - $\top$: Trail building
  - $\bot$: N is refuted
  - A conflict clause

Initially, the state for a first-order clause set N is $(\epsilon; N; \emptyset; 0; \top)$.

SCL – quo vadis?

ground → easy case!

$$N = \{P(a) \vee Q(b),\ \ P(a) \vee \neg Q(b),\ \ \neg P(x) \vee Q(x),\ \ \neg P(x) \vee \neg Q(x)\}$$

$(\varepsilon; N; \emptyset; 0; \top)$

$\Rightarrow_{\text{Decide}} (\neg P(a)^1; N; \emptyset; 1; \top)$

$\Rightarrow_{\text{Propagate}} (\neg P(a)\, \neg Q(b)^{P(a) \vee \neg Q(b) \cdot \{\}}; N; \emptyset; 1; \top)$

$\Rightarrow_{\text{conflict}} (\text{''}\ \text{''}; N; \emptyset; 1; P(a) \vee Q(b) \cdot \{\})$

$\Rightarrow_{\text{Resolve}} (\neg P(a)^1; N; \emptyset; 1; P(a) \vee P(a) \cdot \{\})$

$\Rightarrow$ Fact, Backtracking: $P(a)$

SCL: Motivation

SCL employs a trail consisting of **ground literals** only:

- deciding falsity of a first-order clause with variables can be done practically efficiently and
- different ground literals don't have common instances resulting in efficient trail operations. $P(a)\ \neg P(x)$ ....
- Still, non-redundant clauses **with variables** can be learned
  - Find falsified ground clause
  - Guide resolution on the clause level (with variables)

Resolution learns non-ground clauses

$$N = \{\underbrace{P(x) \lor Q(b), \quad P(x) \lor \neg Q(y)}, \quad \neg P(a) \lor Q(x), \quad \neg P(x) \lor \neg Q(b)\}$$

resolve $P(x) \lor Q(b)$ with $P(x) \lor \neg Q(y)\{y \mapsto b\}$

$\rightarrow P(x) \lor P(x)$

apply Factoring

$\rightarrow P(x)$

SCL: *Simplified* Problem State

$$(\Gamma; N; U; k; D)$$

- Γ: Ground trail
- N: Initial clause set
- U: Learned clauses
- k: Level
- D: States
  - $\top$: Trail building
  - $\bot$: N is refuted
  - A **closure** $C \cdot \sigma$: Conflict clause C with substitution σ

$C\sigma$

C is non-ground in general
σ which is a grd. subst.

Initially, the state for a first-order clause set N is $(\epsilon; N; \emptyset; 0; \top)$.

$(C \vee L)$ is non-grd.
σ is a grd. subst

SimpPropagate $(\Gamma; N; U; k; \top) \Rightarrow_{\text{SCL}} (\Gamma, L\sigma^{(C \vee L)\cdot\sigma}; N; U; k; \top)$

provided $C \vee L \in (N \cup U)$ *technicalities missing, see later...*, $(C \vee L)\sigma$ is ground, $C\sigma$ is false under Γ, and $L\sigma$ is undefined in Γ

Conflict $(\Gamma; N; U; \beta; k; \top) \Rightarrow_{\text{SCL}} (\Gamma; N; U; \beta; k; D \cdot \sigma)$

provided $D \in (N \cup U)$, $D\sigma$ false in Γ for a grounding substitution σ

Resolve $(\Gamma, L\delta^{(C\vee L)\cdot\delta}; N; U; \beta; k; (D \vee L')\cdot\sigma)$

$\Rightarrow_{\text{SCL}}$ $(\Gamma, L\delta^{(C\vee L)\cdot\delta}; N; U; \beta; k; (D \vee C)\eta\cdot\sigma\delta)$

provided $L\delta = \mathrm{comp}(L'\sigma)$, $\eta = \mathrm{mgu}(L, \mathrm{comp}(L'))$

SimpBacktrack $(\Gamma\,\mathrm{comp}(L\sigma)^k; N; U; \beta; k; (D \vee L)\cdot\sigma)$

$\Rightarrow_{\text{SCL}}$ $(\Gamma'; N; U \cup \{D \vee L\}; \beta; j; \top)$

provided *a lot of technicalities...*

SCL learns non-ground clauses

$$N = \{P(x) \lor Q(b),\ \ P(x) \lor \neg Q(y),\ \ \neg P(a) \lor Q(x),\ \ \neg P(x) \lor \neg Q(b)\}$$

$(\varepsilon;\ N;\ \emptyset;\ 0;\ \top)$

$\Rightarrow_{\text{Decide}} (\neg P(a)^1;\ N;\ \emptyset;\ 1;\ \top)$

$\Rightarrow_{\text{Propagate}} (\neg P(a)^1 \neg Q(b)^{P(x) \lor \neg Q(y) \cdot \{x \mapsto a,\, y \mapsto b\}};\ N;\ \emptyset;\ 1;\ \top)$

$\Rightarrow_{\text{Conflict}} (\ "\qquad"\ ;\ N;\ \emptyset;\ 1;\ P(x) \lor Q(b) \cdot \{x \mapsto a\})$

$\Rightarrow_{\text{Resolve}} (\neg P(a)^1 \neg Q(b)^{P(x) \lor \neg Q(y) \cdot \{x \mapsto a,\, y \mapsto b\}};\ N;\ \emptyset;\ 1;\ P(x) \lor P(x) \cdot \{x \mapsto a, [y \mapsto b]\})$

$\Rightarrow_{\text{Factorize}}\ "\qquad"\qquad"\ ;\ P(x) \cdot \{x \mapsto a\})$

$\Rightarrow_{\text{Skip}} (\neg P(a)^1;\ \ldots\ "\qquad"\qquad"\ )$

$\Rightarrow_{\text{Backtrack}} (\varepsilon;\ \ldots\ "\ N;\ \{P(x)\};\ \top)$

Recall: CDCL soundness

2.9.10 Lemma (CDCL Soundness)

In a reasonable CDCL derivation, CDCL can only terminate in two different types of final states: $(M; N; U; k; \top)$ where $M \models N$ and $(M; N; U; k; \bot)$ where N is unsatisfiable.

First problems in first-order

$$N = \{P(a), \quad \neg P(x) \lor P(f(x))\}$$

$$(P(a)^{P(a) \circ \{\}}; N; \emptyset; 0; \top)$$

$$\Rightarrow_{\text{Propagate}} (P(a)^{P(a) \circ \{\}}\ P(f(a))^{\neg P(x) \lor P(f(x)) \circ \{x \mapsto a\}}; \ldots)$$

$$\Rightarrow_{\text{Propagate}} \quad P(f(f(a)))$$

$$\Rightarrow_{\text{Propagate}}$$

this would not terminate!

- First-order Herbrand models are **infinite in general**
- In SCL:
  - **Restrict** the reasoning with respect to some ground literal β
  - Require that **any trail literal** is **smaller** than β
  - Use a well-founded, total, strict ordering $\prec_\beta$ (e.g. KBO)
- Goal: Achieve termination

SCL: Problem State

$$(\Gamma; N; U; \beta; k; D)$$

- Γ: Ground trail
- N: Initial clause set
- U: Learned clauses
- β: Limiting literal
- k: Level
- D: State
  - $\top$: Trail building
  - $\bot$: N is refuted
  - A **closure** $C \cdot \sigma$: Conflict clause C with substitution σ

Initially, the state for a first-order clause set N is $(\epsilon; N; \emptyset; \beta; 0; \top)$.

SCL: Stuck states

$$N = \{P(a), \quad \neg P(x) \vee P(f(x))\}$$

set $\beta = P(f(f(a)))$, hence exactly $P(a) \prec_\beta \beta$ and $P(f(a)) \prec_\beta \beta$

$$\Rightarrow_{\text{Propagate}} (P(a)^{\cdots} \; P(f(a))^{\cdots}; N; \ldots; \beta; \ldots)$$

"Stuck state"

Exhaustive Propagation vs. First-order logic

In propositional logic: Propagation instead of deciding is often a good idea.

In FOL: Exhaustively propagating all ground instances is a very bad idea:

$$N' = \{P(1, 0, x_1, \ldots, x_n),\ \underbrace{Q \vee \neg R,\ Q \vee R,\ \neg Q \vee R,\ \neg Q \vee \neg R}_{\text{unsat.}}\}$$

P(1,0,0,0, .. ,0)
(0 0,1)
(0 ,1,1,1)
(0 1,0,0)

3.16.23 Example (Comparing Proof Length Depending on Clause Propagation)

Let i be a positive integer and consider the clause set N^i with one predicate P of arity i consisting of the following clauses, where we write $\bar{x}, \bar{0}$ and $\bar{1}$ to denote sequences of the appropriate length of variables and constants to meet the arity of P:

$$P(\bar{0}) \qquad \neg P(\bar{1})$$

$P(0,0,0)$ $\neg P(1,1,1)$

and i clauses of the form

$$\neg P(\bar{x}, 0, \bar{1}) \vee P(\bar{x}, 1, \bar{0})$$

where the length of $\bar{1}$ varies between 0 and $i-1$. The example encodes an i-bit counter. An SCL run with exhaustive propagation on this clause set finds a conflict after $O(2^i)$ propagations without any application of Decide.

Example ctd.

$$N^4 = \left\{ \begin{array}{l} 1 : P(0,0,0,0) \\ 2 : \neg P(x_1,x_2,x_3,0) \vee P(x_1,x_2,x_3,1) \\ 3 : \neg P(x_1,x_2,0,1) \vee P(x_1,x_2,1,0) \\ 4 : \neg P(x_1,0,1,1) \vee P(x_1,1,0,0) \\ 5 : \neg P(0,1,1,1) \vee P(1,0,0,0) \\ 6 : \neg P(1,1,1,1) \end{array} \right\}$$

$P(0,0,0,0) \xrightarrow[\text{Propagate}]{(2)} P(0,0,0,1) \xrightarrow{(3)} P(0,0,1,0) \xrightarrow{(2)} \ldots$

2^4 applications of Propagate

+ conflict to clause (6)

2^4 applications of Resolve $\Rightarrow \bot$

Example ctd.

$$N^4 = \left\{ \begin{array}{l} 1 : P(0,0,0,0) \\ 2 : \neg P(x_1,x_2,x_3,0) \lor P(x_1,x_2,x_3,1) \\ 3 : \neg P(x_1,x_2,0,1) \lor P(x_1,x_2,1,0) \\ 4 : \neg P(x_1,0,1,1) \lor P(x_1,1,0,0) \\ 5 : \neg P(0,1,1,1) \lor P(1,0,0,0) \\ 6 : \neg P(1,1,1,1) \end{array} \right\}$$

2.2 Res 3.1	$7 : \neg P(x_1,x_2,0,0) \lor P(x_1,x_2,1,0)$
7.2 Res 2.1	$8 : \neg P(x_1,x_2,0,0) \lor P(x_1,x_2,1,1)$
8.2 Res 4.1	$9 : \neg P(x_1,0,0,0) \lor P(x_1,1,0,0)$
9.2 Res 8.1	$10 : \neg P(x_1,0,0,0) \lor P(x_1,1,1,1)$
10.2 Res 5.1	$11 : \neg P(0,0,0,0) \lor P(1,0,0,0)$
11.2 Res 10.1	$12 : \neg P(0,0,0,0) \lor P(1,1,1,1)$
12.1 Res 6.1	$13 : \perp$

$2 \cdot 4$ steps

$< 2^4$ steps

Can be simulated with SCL, but not with exhaustive propagation

Propagate $(\Gamma; N; U; \beta; k; \top) \Rightarrow_{\text{SCL}}$
$(\Gamma, L\sigma^{(C_0 \vee L)\delta \cdot \sigma}; N; U; \beta; k; \top)$

provided $C \vee L \in (N \cup U)$, $C = C_0 \vee C_1$, $C_1\sigma = L\sigma \vee \cdots \vee L\sigma$, $C_0\sigma$ does not contain $L\sigma$, δ is the mgu of the literals in C_1 and L, $(C \vee L)\sigma$ is ground, $(C \vee L)\sigma \prec_\beta \{\beta\}$, $C_0\sigma$ is false under Γ, and $L\sigma$ is undefined in Γ

$\underbrace{P}_{C_0} \vee Q(x) \vee Q(5)$ $\quad \sigma = \{x \to 5\}$

The rule Propagate applies exhaustive factoring to the propagated literal with respect to the grounding substitution σ and annotates the factored clause to the propagation literal on the trail.

Decide $(\Gamma; N; U; \beta; k; \top) \Rightarrow_{\text{SCL}}$
$(\Gamma, L\sigma^{k+1}; N; U; \beta; k+1; \top)$

provided $L \in C$ for a $C \in (N \cup U)$, $L\sigma$ is a ground literal undefined in Γ, and $L\sigma \prec_\beta \beta$

closure

Conflict $(\Gamma; N; U; \beta; k; \top) \Rightarrow_{\text{SCL}} (\Gamma; N; U; \beta; k; D \cdot \sigma)$

provided $D \in (N \cup U)$, $D\sigma$ false in Γ for a grounding substitution σ

These rules construct a (partial) model via the trail Γ for $N \cup U$ until a conflict, i.e., a false clause with respect to Γ is found or all ground atoms smaller β are defined in M and $M \models \mathrm{grd}(N)^{\prec_\beta}$.

we don't know if is

- $M \not\models N$
- M is extendable to a model for N

- Guaranteed Termination
  - Only ground literals $\prec_\beta \beta$ considered
  - There are only finitely many.
- Choosing the right β is crucial
  - For some fragments, this gives completeness: Bernays-Schoenfinkel
  - In general: Every fragment with finite models

Next up: Conflict resolution rules

Before any conflict resolution step, we assume that the respective clauses are renamed such that they do not share any variables and that the grounding substitutions of closures are adjusted accordingly.

Skip $(\Gamma, L; N; U; \beta; k; D \cdot \sigma) \ \Rightarrow_{\text{SCL}}$
$(\Gamma; N; U; \beta; k - i; D \cdot \sigma)$

provided $\mathtt{comp}(L)$ does not occur in $D\sigma$, if L is a decision literal then $i = 1$, otherwise $i = 0$

Factorize $(\Gamma; N; U; \beta; k; (D \lor L \lor L') \cdot \sigma) \ \Rightarrow_{\text{SCL}}$
$(\Gamma; N; U; \beta; k; (D \lor L)\eta \cdot \sigma)$

provided $L\sigma = L'\sigma$, $\eta = \text{mgu}(L, L')$

Resolve $(\Gamma, L\delta^{(C \lor L)\cdot\delta}; N; U; \beta; k; (D \lor L') \cdot \sigma)$
$\Rightarrow_{\text{SCL}}$ $(\Gamma, L\delta^{(C \lor L)\cdot\delta}; N; U; \beta; k; (D \lor C)\eta \cdot \sigma\delta)$
provided $L\delta = \text{comp}(L'\sigma)$, $\eta = \text{mgu}(L, \text{comp}(L'))$

Backtrack $(\Gamma_0, K, \Gamma_1, \text{comp}(L\sigma)^k; N; U; \beta; k; (D \lor L) \cdot \sigma)$
$\Rightarrow_{\text{SCL}}$ $(\Gamma_0; N; U \cup \{D \lor L\}; \beta; j; \top)$
provided $D\sigma$ is of level $i' < k$, and Γ_0, K is the minimal trail subsequence such that there is a grounding substitution τ with $(D \lor L)\tau$ is false in Γ_0, K but not in Γ_0, and Γ_0 is of level j

The clause $D \lor L$ added by the rule Backtrack to U is called a *learned clause*.

- $\perp$ can only be derived by Resolve (or be present already in N)
  $\Longrightarrow$ The generation of $\perp$ is a resolution refutation
- Freedom with respect to decisions and factorizations
- Literals are not removed during resolution (eventually, Skip removes the literal from Γ)