Logic for AI : Part II

Marius

December 2021

Davis-Putman-Loveland-Logmann

DPLL Algorithm

- Guess the assignment of a variable
- Propagate the decision through unit propagation
- In case of conflict, switch the last decision

CDCL Algorithm

- Guess the assignment of a variable
- Propagate the decision through unit propagation
- In case of conflict, switch the last decision and add a cause preventing the partial assignment that led to the conflict.

Conflit Driven Clause Learning II

Definitions

A *cut clause* is a cut that separates the decisions from the conflict in an oriented graph where edges are (partial) implication of unit propagation.
A *Unique Implication Point (UIP)* is a literal l propagated at decision level d occuring in each path from decision d to the conflict

Optimisations

Any cut clause may replace the clause learnt.
A way to do it is to iteratively replace a decision with an UIP, preferably the closest one to the conflict
Then, We may add *back clauses* to keep informations about the relations between UIPs and decisions.

Conflit Driven Clause Learning in theory $\mathcal{T}$

Definitions

A *boolean encoder* e replaces atoms of the theories with new boolean (e.g $x < 2 \rightarrow x_0$)
A *unsat core* is a minimal subset of the atoms of the formula that is unsat.

CDCL($\mathcal{T}$) Algorithm

- Apply CDCL Algorithm
- At each decision, apply Theory Propagation Algorithm.

Theory Propagation Algorithm

- Check Satisfiability of $e^{-1}(m)$
- If UNSAT, return an unsat core and a decision level.
- If SAT returns atoms of the input formula implied by $e^{-1}(m)$.

Theory 1 : Equality and Function symbols (basics)

EUF

Signature : $\{\mathcal{F}, =\}$
Deduction : Symmetry, Transitivity, Congruence

Remark

Constant may be removed (There is an equisatisfiable formula).

Theory 1 : Equality and Function symbols (implementation of TheoryPropagation)

Ackermann's reduction

- Encode terms with new constants
- Encode atoms with variables (boolean encoder)
- Solve the satisfiability of $F \wedge Rules$

This is possible because applicable rules are finite and determined by the subterms of the initial formula.

A quick example

$x = y \Rightarrow f(x) = f(y)$

$t_1 = t_2 \Rightarrow t_3 = t_4, t_1 = t_2 \Rightarrow t_2 = t_1 ...$

$p_1 \Rightarrow p_2, p_1 \Rightarrow p_3 ...$

Theory 1 : Equality and Function symbols (implementation of TheoryPropagation II)

Lazy-QF_EUF

- Process each subformula $t_1 \otimes t_2$.
- Keep tracks of equivalence class among terms and subterms.

A quick example

$f(x) = x \wedge f(f(x)) \neq x$
$f(x) = x \rightarrow C = [\{x, f(x)\}]$
$f(f(x)) \neq x \rightarrow C = [\{x, f(x)\}, \{f(f(x))\}], D = [(C_0, C_1)]$
$f(x), f(f(x)) \in T \wedge x \sim f(x) \rightarrow$ merge C_0 and C_1.
As $(C_0, C_1) \in D$, return UNSAT.

LRA

Signature : $\{\mathbb{N}, <, +\}$
Deduction : Non-negative linear combination of inequalities

Fundamental Theorem of Linear Inequality

Let $(a_i)_{0<m}$ and $b \in \mathbb{R}^n$. Then :
• Either $\exists \lambda_0 ... \lambda_{m-1}\ b = \lambda_0 a_0 ... \lambda_{m-1} a_{m-1}$ and $a_1 ... a_m$ linearly independent.
• Or there exists a hyperplane $\{c | cx = 0\}$ such that $ca_i \geq 0$ and $cb < 0$

Remarks

This is a fancy way to say that either b is in the cone generated by $(a_i)_{0\leq i<m}$ or it is not.
A *cone* generated by $(a_i)_{0\leq i<m}$ is $\{\lambda_0 a_0 ... \lambda_{m-1} a_{m-1}\}$