Mathematical Logics
PL - Reasoning via Truth Tables*

Fausto Giunchiglia and Mattia Fumagalli

University of Trento

**Originally by Luciano Serafini and Chiara Ghidini*
Modified by Fausto Giunchiglia and Mattia Fumagalli

Lecture index

1. Truth tables – summary
2. Decision problems
3. Deciding satisfiability (SAT)
   a. CNF
   b. The DPLL SAT decision procedure
4. Satisfiability in practice: MiniSat

Davis-Putnam (DP) Algorithm

- In 1960, Davis and Putnam published a SAT algorithm.

  Davis, Putnam. A Computing Procedure for Quantification Theory. Journal of the ACM, 7(3):2012˘013215, 1960.

- In 1962, Davis, Logemann, and Loveland improved the DP algorithm.

  Davis, Logemann, Loveland. A Machine Program for Theorem-Proving. Communications of the ACM, 5(7):3942˘013397, 1962.

- The DP algorithm is often confused with the more popular DLL algorithm. In the literature you often find the acronym DPLL.

- Basic framework for most current SAT solvers.
- We consider the DP algorithm …

Satisfiability of a set of clauses

- Let $N = C_0, \ldots, C_n = CNF(\varphi)$
  - $I \vDash \varphi$ if and only if $I \vDash C_i$ for all $i = 0..n$;
  - $I \vDash C_i$ if and only if for some literal $l \in C$, $I \vDash l$
- To check if a model I satisfies N we do not need to know the truth values that I assigns to all the literals appearing in N.
- For instance, if $I(p) =$ *true* and $I(q) =$ *false*, we can say that $I \vDash \{\{p, q, \neg r\}, \{\neg q, s\}\}$, without considering the evaluations of $I(r)$ and $I(s)$.

Partial evaluation

A partial evaluation is a partial function that associates to some propositional variables of the alphabet PROP a truth value (either true or false) and can be undefined for the other elements of PROP.

Partial Valuation

- Partial evaluations allow us to construct models for a set of clauses $N = \{C_1, \ldots, C_n\}$ incrementally
- DPLL starts with an empty valuation (i.e., the truth values of all propositional letters are not defined) and tries to extend it step by step to all propositional letters occurring in $N = \{C_1, \ldots, C_n\}$.
- Under a partial valuation I the literals and clauses can be true, false or undefined;
  - A clause is true under I if one of its literals is true;
  - A clause is false (or conflicting) if all its literals are false;
  - otherwise C it is undefined (or unresolved).

DPLL

Simplification of a formula by an evaluated literal

For any CNF formula φ and atom p, $\varphi|_p$ stands for the formula obtained from φ by replacing all occurrences of p by $\top$ and simplifying the result by removing

- all clauses containing the disjunctive term $\top$, and the
- literals $\neg\top = \bot$ in all remaining clauses

Similarly, $\varphi|_{\neg p}$ is the result of replacing p in φ by $\bot$ and simplifying the result, according to the process dual to above.

Example

For instance,

$$\{\{p, q, \neg r\}, \{\neg p, \neg r\}\}|_{\neg p} = \{\{q, \neg r\}\}$$

DPLL (cont'd)

Unit clause

If a CNF formula φ contains a clause $C = \{l\}$ that consists of a single literal l, it is a unit clause

Unit propagation

If φ contains unit clause $\{l\}$ then, to satisfy φ we have to satisfy $\{l\}$ and therefore the literal l must be evaluated to True. As a consequence φ can be simplified using the procedure called UnitPropagation

```
UnitPropagation(φ, I)
    while φ contains a unit clause {l}
        φ := φ|l
        if l = p, then I(p) := true
        if l = ¬p, then I(p) := false
    end
```

DPLL (cont'd)

Example

$$\text{UnitPropagation}(\{p\}, \{\neg p, \neg q\}, \{\neg q, r\}\}, I)$$

$\{\{p\}, \{\neg p, \neg q\}, \{\neg q, r\}\}$

$\{\{p\},\{\neg p,\neg q\},\{\neg q,r\}\}|_p$ $\quad$ $I(p) = \mathit{true}$

$\{\{T\}, \{\neg T, \neg q\}, \{\neg q, r\}\}$

$\{\{\neg q\}, \{\neg q, r\}\}$

$\{\{\neg q\}, \{\neg q, r\}\}$

$\{\{\neg q\},\{\neg q,r\}\}|_{\neg q}$ $\quad$ $I(q) = \mathit{false}$

$\{\{T\}, \{T, r\}\}$

$\{\}$

Exercize

Use unit propagation to decide whether the formula

$$p \wedge (p \vee q) \wedge (\neg p \vee \neg q) \wedge (q \vee r) \wedge (\neg q \vee \neg r)$$

is satisfiable.

DPLL (cont'd)

Example

$\mathrm{UnitPropagation}(\{p\}, \{\neg p\}, \{\neg q, r\}\}, I)$

$\{\{p\}, \{\neg p\}, \{\neg q, r\}\}$

$\{\{p\}, \{\neg p\}, \{\neg q, r\}\}|_p$ $\quad$ $\boldsymbol{I(p) = true}$

$\{\{T\}, \{\neg T\}, \{\neg q, r\}\}$

$\{\{\}, \{\neg q, r\}\}$

$\{\ldots, \{\}, \ldots\}$ $\quad$ Initial clause unsatisfiable

DPLL (cont'd)

Remark

Unit propagation is enough to decide the satisfiability problem when it terminates with the following two results:

- $\{\}$ as in the example above, then the initial formula is satisfiable,
- $\{\ldots\{\}\ldots\}$, with $\{\}$ the empty clause, then the initial formula is unsatisfiable

There are cases in which $\mathtt{UnitPropagation}$ does not terminate, i.e., when there is no unit clause and the CNF is not empty and doesn't contain empty clauses. e.g.,

$$\{\{p, q\}, \{\neg q, r\}\}$$

In this case we have to guess . . . .

DPLL definition

The Davis-Putnam-Logemann-Loveland procedure

DPLL is an extension of the unit propagation method that can solve the satisfiability

```
DPLL(φ,I)
    UnitPropagation(φ,I)
    if φ contains the empty clause { }
        then return False
    if φ = { }
        then exit with I
    select a literal l∈C ∈φ
    DPLL(φ|l, I ∪ (I(l) = true) ) or
    DPLL(φ|ln, I ∪ (I(l) = false) )
```

where: if l= p, ln $= \neg p$ and if l$=\neg p$ then ln = p

Note: heuristic choice of literal *l* in order to achieve efficiency

Mathematical Logics
PL - Reasoning via Truth Tables*

Fausto Giunchiglia and Mattia Fumagalli

University of Trento

**Originally by Luciano Serafini and Chiara Ghidini*
Modified by Fausto Giunchiglia and Mattia Fumagalli