2-Propositional Logic

2.3-The DPLL decision procedure

UNIVERSITÀ
DI TRENTO

DataScientia
Unitas per Varietatem

2.3-The DPLL decision procedure

1. Basic notions
2. Conjunctive Normal Form (CNF)
3. Satisfiability of a CNF formula
4. The decision procedure

2.3.1-Basic notions

Lecture index

The DPLL SAT decision procedure (remark)

- *Davis, Putnam. A Computing Procedure for Quantification Theory. Journal of the ACM, 7(3), 1960.*
- *Davis, Logemann, Loveland. A Machine Program for Theorem-Proving. Communications of the ACM, 1962.*
- Huge amount of work based on these ideas. In the literature you often find the acronym **DPLL** (for Davis, Putnam, Longemann, Loveland). Lots of work still on going, with applications in virtually all the domains where there is a need of automated reasoning (e.g., hardware and software verification, scheduling, planning, space, ..).
- **DPLL** is the *de-facto* reference standard for the implementation of SAT-based reasoning.
- **https://datascientia.education/logictools/**: here you can find various inference procedures for PL (including truth tables/ DPLL).
- **http://minisat.se/**: here you can find binaries, sources, documentation and projects related to **MiniSat**. MiniSat is a minimalistic, open-source DPLL based SAT solver. Released under the MIT licence.

Why DPLL SAT (remark)

- SAT (UNSAT) is a key property, as it amounts to checking whether a certain theory can be instantiated in practice (think, e.g., of scheduling);
- PL SAT is **NP-complete**: all the NP-complete problems can be encoded in PL SAT;
- **Deduction theorem**: If $\Gamma, \phi \models \psi$ then $\Gamma \models \phi \supset \psi$, (with Γ possibly empty). This allows to reduce checking logical consequence to a PL SAT problem;
- PL SAT can be reduced to CNF PL SAT (for Conjunctive Normal form, see later, i.e., conjunctions of disjunctions of possibly negated atomic propositions);
- CNF SAT can be implemented very efficiently by exploiting smart **heuristics** (e.g. strategies for selecting the "best" truth assignment);
- State of the art SAT solvers, called **CDCL** (for Conflict-Driven Clause-Learning) solve industrial problem up to a few million atomic propositions;
- ... Last year, IBM and Google announced a quantum computer solving SAT problems.

- Basic notion
- Properties of CNF formulas
- Computing the CNF of a formula

2.3.2-Conjunctive Normal Form (CNF)

Lecture index

Conjunctive normal form (CNF) - (notion)

Definition 1 (Literal) A literal is either a propositional variable or the negation of a propositional variable, two examples being the formulas p, $\neg q$

Definition 2 (Clause) A clause is a disjunction of literals, an example being the formula $(p \vee \neg q \vee r)$

Conjunctive normal form (CNF) - (notion)

Definition 3 (Conjunctive normal form (CNF)) A formula is in conjunctive normal form, if it is a conjunction of clauses, an example being the formula

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

Conjunctive Normal Form (CNF) (remark)

A CNF formula has the following shape:

$$(L_{(1,1)} \vee \ldots \vee L_{(1,n_1)}) \wedge \ldots \wedge (L_{(m,1)} \vee \ldots \vee L_{(m,n_m)})$$

equivalently written as:

$$\bigwedge_{i=1}^{m} (\bigvee_{j=1}^{n_j} L_{ij})$$

where L_{ij} is the j-th literal of the i-th clause

Example 1 (Conjunctive Normal Form, special cases)

- $\{\}$
- p
- $\neg p$
- $p \wedge q \wedge r$
- $p \vee q \vee r$

Properties of clauses 1 (notion)

Order of literals does not matter

- If a clause is obtained by reordering the literals of a clause C' then the two clauses are equivalent.

Example 2 $(p \vee q \vee r \vee \neg r) \equiv (\neg r \vee q \vee p \vee r)$

Remark

Commutativity of $\vee$: $\phi \vee \psi \equiv \psi \vee \phi$

Properties of clauses 2 (notion)

Multiple literals can be merged

- If a clause contains more than one occurrence of the same literal then it is equivalent to the clause obtained by deleting all but one of these occurrences

Example 3 $(p \vee q \vee r \vee q \vee \neg r) \equiv (p \vee q \vee r \vee \neg r)$

Remark

Absorption of $\vee$: $\phi \vee \phi \equiv \phi$

Properties of clauses 3 (notion)

Clauses as sets of literals

- From these properties we can represent a clause as a set of literals, by leaving disjunction implicit and by ignoring replication and order of literals

Example 4 $(p \vee q \vee r \vee \neg r)$ is represented by the set $\{p, q, r, \neg r\}$

Properties of CNF formulas 1 (notion)

Order of clauses does not matter

- If a CNF formula ϕ is obtained by reordering the clauses of a CNF formula ϕ' then ϕ and ϕ' are equivalent

Example 5 $(p \vee q) \wedge (r \vee \neg q) \wedge (\neg q) \equiv (r \vee \neg q) \wedge (\neg q) \wedge (p \vee q)$

Remark

Commutativity of $\wedge$: $\phi \wedge \psi \equiv \psi \wedge \phi$

Properties of CNF formulas 2 (notion)

Multiple clauses can be merged

- If a CNF formula contains more than one occurrence of the same clause then it is equivalent to the formula obtained by deleting all but one of the duplicated occurrences

Example 6 $(p \vee q) \wedge (r \vee \neg q) \wedge (p \vee q) \equiv (p \vee q) \wedge (r \vee \neg q)$

Remark

Absorption of $\wedge$: $\phi \wedge \phi \equiv \phi$

Properties of CNF formulas (notion)

A CNF formula can be seen as a set of clauses

- A CNF formula can be represented as a set of sets of literals

Example 7 $(p \vee q) \wedge (r \vee \neg q) \wedge (\neg \vee q)$ is represented by $\{\{p, q\}, \{r, \neg q\}, \{\neg q\}\}$

CNF formulas (main properties)

Proposition (Existence) Every formula can be rewritten into Conjunctive Normal Form

Proposition (Equivalence) $\models CNF(\phi) \equiv \phi$

CNF of a formula (notion)

Definition 4 (The CNF function) Given a PL formula ϕ the function CNF, which transforms ϕ in its CNF form, called CNF(ϕ) is recursively defined as follows:

$$
\begin{array}{lcl}
\text{CNF(p)} & = & \text{p if p} \in \textbf{PROP} \\
\text{CNF}(\neg\ \text{p}) & = & \neg\ \text{p if p} \in \textbf{PROP} \\
\text{CNF}(\phi \supset \psi) & = & \text{CNF}(\neg\phi) \otimes\ \text{CNF}(\psi) \\
\text{CNF}(\phi \wedge \psi) & = & \text{CNF}(\phi) \wedge \text{CNF}(\psi) \\
\text{CNF}(\phi \vee \psi) & = & \text{CNF}(\phi) \otimes\ \text{CNF}(\psi) \\
\text{CNF}(\phi \equiv \psi) & = & \text{CNF}(\phi \supset \psi) \wedge \text{CNF}(\psi \supset \phi) \\
\text{CNF}(\neg\neg\phi) & = & \text{CNF}(\phi) \\
\text{CNF}(\neg(\phi \supset \psi)) & = & \text{CNF}(\phi) \wedge \text{CNF}(\neg\psi) \\
\text{CNF}(\neg(\phi \wedge \psi)) & = & \text{CNF}(\neg\phi) \otimes\ \text{CNF}(\neg\psi) \\
\text{CNF}(\neg\ (\phi \vee \psi)) & = & \text{CNF}(\neg\phi) \wedge \text{CNF}(\neg\psi) \\
\text{CNF}(\neg\ (\phi \equiv \psi)) & = & \text{CNF}(\phi \wedge \neg\psi) \otimes\ \text{CNF}(\psi \wedge \neg\phi)
\end{array}
$$

where $(C_1 \wedge \ldots \wedge C_n) \otimes\ (D_1 \wedge \ldots \wedge D_m)$ is defined as:

$$(C_1 \vee D_1) \wedge \ldots \wedge (C_1 \vee D_m) \wedge \ldots \wedge (C_n \vee D_1) \wedge \ldots \wedge (C_n \vee D_m)$$

with C_i being a conjunction (possibly single formula) and D_j being a disjunction (possibly single formula)

CNF of a formula (example)

Example 8 (CNF transformation)

$$\begin{aligned} CNF((a \wedge b) \vee \neg(c \supset d)) &= \\ CNF(a \wedge b) \otimes\ CNF(\neg(c \supset d)) &= \\ (CNF(a) \wedge CNF(b)) \otimes\ (CNF(c) \wedge CNF(\neg d)) &= \\ (a \wedge b) \otimes\ (c \wedge \neg d) &= \\ (a \vee c) \wedge (a \vee \neg d) \wedge (b \vee c) \wedge (b \vee \neg d) &= \end{aligned}$$

Example 9 (Exponential explosion) Compute the CNF of

$$p1 \equiv (p2 \equiv (p3 \equiv (p4 \equiv (p5 \equiv p6))))$$

Cost of CNF (remark)

- In the second exammple above, the formula resulting from the first step is:

$$CNF(p1 \supset (p2 \equiv (p3 \equiv (p4 \equiv (p5 \equiv p6))))) \wedge CNF((p2 \equiv (p3 \equiv (p4(p5 \equiv p6)))) \supset p1)$$

- Continuing, the formula will keep growing exponentially;
- In the worst case the formula CNF(ϕ) is exponentially longer than ϕ (construct a formula where this is the case using only conjunction, disjunction and negation);
- Deciding validity/ unsatisfiability of a CNF formula can be done in linear time (prove it);
- CNF formulas are simpler to handle and checking satisfiability/validity of a formula in CNF is easier.

Computing the CNF of a formula (example)

Example 10 Compute the CNF of $(q \land p) \lor \neg p$

$$
\begin{aligned}
&CNF((q \land p) \lor \neg p) \\
&CNF((q \land p)) \otimes\ CNF(\neg p) \\
&(CNF(q) \land CNF(p)) \otimes\ \neg p \\
&(q \land p) \otimes\ \neg p \\
&(q \lor \neg p) \land (p \lor \neg p)
\end{aligned}
$$

Computing the CNF of a formula (example)

Example 11 Compute the CNF of $(p \supset q) \equiv (\neg q \supset \neg p)$

- $\mathrm{CNF}((p \supset q) \equiv (\neg q \supset \neg p))$
- $\mathrm{CNF}((p \supset q) \supset (\neg q \supset \neg p)) \wedge \mathrm{CNF}((\neg q \supset \neg p) \supset (p \supset q))$
- $\mathrm{CNF}(\neg(p \supset q)) \otimes \mathrm{CNF}(\neg q \supset \neg p) \wedge \mathrm{CNF}(\neg(\neg q \supset \neg p)) \otimes \mathrm{CNF}(p \supset q)$
- $(\mathrm{CNF}(p) \wedge \mathrm{CNF}(\neg q)) \otimes (\mathrm{CNF}(q) \otimes \mathrm{CNF}(\neg p)) \wedge (\mathrm{CNF}(\neg q) \wedge \mathrm{CNF}(p)) \otimes (\mathrm{CNF}(\neg p) \otimes \mathrm{CNF}(q))$
- $(p \wedge \neg q) \otimes (q \otimes \neg p) \wedge (\neg q \wedge p) \otimes (\neg p \otimes q)$
- $(p \wedge \neg q) \otimes (q \vee \neg p) \wedge (\neg q \wedge p) \otimes (\neg p \vee q)$
- $(p \vee q \vee \neg p) \wedge (\neg q \vee q \vee \neg p) \wedge (\neg q \vee q \vee \neg p) \wedge (\neg p \vee q \vee p)$

Computing the CNF of a formula (example)

Example 12 Compute the CNF of $(p \wedge r) \supset q$

$$
\begin{aligned}
& CNF((p \wedge r) \supset q) \\
& CNF((\neg(p \wedge r))) \otimes CNF(q) \\
& (CNF((\neg p)) \otimes CNF(\neg r)) \otimes q \\
& (\neg p \otimes \neg r) \otimes q \\
& (\neg p \vee \neg r) \otimes q \\
& (\neg p \vee q) \wedge (\neg r \vee q)
\end{aligned}
$$

Disjunctive Normal Form (remark)

- A formula can also be translated in Disjunctive Normal Form (DNF), namely in a disjunction of conjunction of literals;
- A DNF formula has the following shape:

$$(L_{(1,1)} \wedge \ldots \wedge L_{(1,n_1)}) \vee \ldots \vee (L_{(m,1)} \wedge \ldots \wedge L_{(m,n_m)})$$

  equivalently written as:

$$\bigvee_{i=1}^{m} \left(\bigwedge_{j=1}^{n_j} L_{ij} \right)$$

  where L_{ij} is the j-th literal of the i-th clause
- DNF(ϕ) can be exponentially longer than ϕ (construct a formula where this is the case using only conjunction, disjunction and negation);
- Deciding satisfiability of a DNF formula can be done in linear time (prove it);
- Deciding validity and unsatisifiability of a DNF formula is NP complete;
- Question: why people translate in CNF and not in DNF being the latter exponentially simpler to check for satisfiability?

- Satisfiability of a set of clauses
- Formula simplification
- Satisfiability of a CNF formula

2.3.3-Satisfiability of a CNF formula

Lecture index

Satisfiability of a set of clauses (notion)

Proposition (Satisfiability of a set of clauses) Let $CNF(\phi) = C_0, \ldots C_n$, where $C_0, \ldots C_n$ are the clauses in CNF(ϕ). Then we have the following:

- $I \models \phi$ if and only if $I \models C_i$ for all $i = 0 \ldots n$
- $I \models C_i$ if and only if for some literal $lit \in C_i$, $I \models lit$

Satisfiability of a set of clauses (remarks)

- To check if a model I satisfies a formula ϕ we **do not need to know the truth values that I assigns to all the literals appearing in ϕ**.
- For instance, if $I(p)$ = true and $I(q)$ = false, we can say that $I \models \{\{p, q, \neg r\}, \{\neg q, s\}\}$
- 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**
- Under a partial evaluation I, the literals and clauses can be true, false or undefined
- We have the following:
  - A clause is true under I if at least one of its literals is true;
  - A clause is false (or conflicting) if all literals are false;
  - In all the other cases, a clause C is undefined (or unresolved)
  - A clause is left undefined when the truth value of its literals is irrelevant to the computation of the current interpretation

Formula simplification via literal evaluation (notion)

Definition 5 (Formula simplification by positive literal) For any CNF formula ϕ and atom p, $\phi|_p$ stands for the formula obtained from ϕ by

- replacing all occurrences of p by the truth value $\top$ and
- by simplifying the result by removing:
  - the clauses containing the disjunctive term $\top$;
  - the literals $\neg\top = \bot$ in all remaining clauses.

Definition 6 (Formula simplification by negative literal) For any CNF formula ϕ and atom $\phi|_{\neg p}$ stands for the formula obtained from ϕ by

- replacing all occurrences of p by the truth value $\bot$ and
- by simplifying the result by removing:
  - the clauses containing the disjunctive term $\neg\bot = \top$;
  - the literals $\top$ in all remaining clauses.

Formula simplification via literal evaluation (example)

Example 13 (Simplification of a formula by an evaluated literal (example))

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

- The second clause is verified because it contains $\neg p$ which we assume to be true($\top$) from the $|_{\neg p}$ notation.
- The first clause which contains p isn't verified by assuming $\neg p$ as $\top$, so we leave there the clause and we try to verify it by using the remaining literals.

Satisfiability of a CNF formula (notion)

Proposition Let $CNF(\phi) = C_0, \ldots C_n$, where $C_0, \ldots C_n$ are the clauses in CNF(ϕ). Let us assume that we iterate the process of literal evaluation.

Then the process will terminate with one of two possible situations:

- {}, that is, with an **empty set of clauses**, in which case ϕ is satisfiable;
- $\{\ldots\{\}\ldots\}$, that is, with a **non empty set of clauses containing one empty clause**, in which case ϕ this unsatisfiable

Remark

- The first situation arises when, within CNF(ϕ), all the clauses have been progressively eliminated because of multiple occurrences of $\top$;
- The second situation arises when, within one clause in CNF(ϕ), all the literals have been progressively eliminated because of multiple occurrences of $\bot$;
- The process will terminate, independently of the order of selection of the literals being evaluated.

Checking the satisfiability a CNF formula (example)

Example 14 Check the satisfiability of the following formula

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

1. $(\neg p \vee q) \wedge (\neg r \vee q)$
2. $\{\{\neg p, q\}, \{\neg r, q\}\}$
3. $\{\{\neg p, \top\}, \{\neg r, \top\}\}|_q$
4. $\{\}$

Checking the satisfiability a CNF formula (example)

Example 15 Check the satisfiability of the following formula

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

1. $(p \vee q) \wedge (p \vee \neg\, p) \wedge (\neg q \vee q) \wedge (\neg\, q \vee \neg\, p) \wedge (\neg q \vee \neg\, p) \wedge (\neg\, q \vee \neg\, p) \wedge (\neg\, q \vee q\,) \wedge (p \vee \neg\, p) \wedge (p \vee q)$
2. $\{\{p, q\}, \{p, \neg p\}, \{\neg q, q\}, \{\neg q, \neg p\}, \{\neg q, \neg p\}, \{\neg q, \neg p\}, \{\neg q,q\}, \{p, \neg p\}, \{p,q\}\}$
3. $\{\{\top, q\}, \{\top, \bot\}, \{\neg q, q\}, \{\neg q, \bot\}, \{\neg q, \bot\}, \{\neg q, \bot\}, \{\neg q,q\}, \{\top, \bot\}, \{\top,q\}\}\,|_p$
4. $\{\{\neg q, q\}, \{\neg q\}, \{\neg q\}, \{\neg q\}, \{\neg q,q\}\}$
5. $\{\{\top, \bot\}, \{\top\}, \{\top\}, \{\top\}, \{\top,\bot\}\}|_{\neg q}$
6. $\{\}$

Checking the satisfiability a CNF formula (example)

Example 16 Check the satisfiability of the following formula

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

1. $(q \vee \neg p) \wedge (q \vee \neg p)$
2. $\{\{q, \neg p\}, \{q, \neg p\}\}$
3. $\{\{q, \top\}, \{q, \top\}\}|_{\neg p}$
4. $\{\}$

Checking the satisfiability a CNF formula (example)

Example 17 Select one the four possible interpretations and check whether it is a model for the formula below:

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

1. $(q \vee \neg p) \wedge (\neg q \vee p) \wedge (p \vee q)$
2. $\{\{q, \neg p\}, \{\neg q, p\}, \{p,q\}\}$
3. $\{\{q, \top\}, \{\neg q, \bot\}, \{\bot, q\}\}|_{\neg p}$
4. $\{\{\neg q\}, \{q\}\}$
5. $\{\{\bot\}, \{\top\}\}|_{q}$
6. $\{\{\}\}$

NOTE: Note however that the above result does not mean that the formula is unsatisfiable. It only means that the chosen interpretation function (p=F, q=T) is not a model for the formula. Still this formula is satisfiable. To prove this it is sufficient to take p=T, q=T.

- Unit clauses
- DPLL

2.3.4-The decision procedure

Lecture index

Simplification of a formula by unit propagation (notion)

Definition 7 (Unit clause) If a CNF formula ϕ contains a clause $C = \{l\}$ that consists of a single literal l, it is a unit clause.

Remark

A formula ϕ containing a unit clause $\{l\}$

- is satisfiable only if k is evaluated as $\top$.
- it allows to simplify ϕ appyling the rules above for evaluated literals

Simplification of a formula by unit propagation (notion)

Proposition

while ϕ *contains a unit clause* $\{l\}$ **do**
    ϕ = $\phi|_l$
    if $l = p$, then $I(p)$ =true
    if $l = \neg p$, then $I(p)$ =false
end

Simplification of a formula by unit propagation (example) I

Example 18 (Unit propagation) Consider the following CNF formula ϕ, taking $\phi = \{\{p\}, \{\neg p, \neg q\}, \{\neg q, r\}\}$. Check whether ϕ is satisfiable by unit propagation. If so, find an interpretation I so that $I \models \phi$.

$$
\begin{aligned}
&\{\{p\}, \{\neg p, \neg q\}, \{\neg q, r\}\} \\
&\{\{p\}, \{\neg p, \neg q\}, \{\neg q, r\}\} \,|_p \\
&\{\{\top\}, \{\bot, \neg q\}, \{\neg q, r\}\} \\
&\{\{\neg q\}, \{\neg q, r\}\} \\
&\{\{\neg q\}, \{\neg q, r\}\} \,|_{\neg q} \\
&\{\{\top\}, \{\top, r\}\} \\
&\{\}
\end{aligned}
\qquad \phi
$$

is satisfiable, and $I = \{p, \neg q\}$. The literal r is left undefined, because in order to satisfy the formula there is no need to evaluate it. This is an example of partial evaluation.

Simplification of a formula by unit propagation (example) II

Example 19 (Unit propagation) Consider the following CNF formula ϕ, taking $\phi = \{\{p\}, \{\neg p\}, \{\neg q, r\}\}$ is satisfiable and if so, find an interpretation I so that $I \models \phi$.

$\{\{p\}, \{\neg p\}, \{\neg q, r\}\}$
$\{\{p\}, \{\neg p\}, \{\neg q, r\}\} \mid_p$
$\{\{\top\}, \{\bot\}, \{\neg q, r\}\}$
$\{\{\}, \{\neg q, r\}\}$
$\{\ldots, \{\}, \ldots\}$

The second clause cannot be verified, and thus the entire formula is not satisfiable.

Unit propagation (remark)

- There are cases in which Unit Propagation does not generate one of the two termination conditions;
- In this case, we have to guess and assign truth values of literals;
- each literal will generate a branch of two cases, one for each truth value.
- each branch doubles the number of interpretation functions to be analyzed. This is where the exponential explosion arises.

Example 20 $\{\{p, q\}, \{\neg q, r\}\}$

The DPLL decision procedure (notion)

DPLL(ϕ, I)
if ϕ *contains the empty clause* {} **then**
  return False;
end
if ϕ = {} **then**
  exit with I;
end
select a literal $l \in C \in \phi$
DPLL($\phi|_k, I \cup (I(k) = true)$) or DPLL($\phi|_{\neg k}, I \cup (I(k) = true)$)

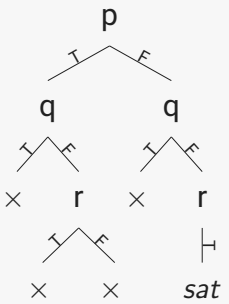
In order to achieve efficiency, we have to make an heuristic choice of literals. Problem of **backtracking** in case of wrong decisions.

Backtracking in DPLL (example)

Example 21 (Backtracking) Consider the following formula

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

The search for an assignment can be represented by the following tree:



The DPLL decision procedure (question)

- Merge in the most effective and minimal way the code for Unit propagation into the code of DPLL.

Example 22 Consider the following examples

1. $(p \supset q \supset r) \land p \land \neg q$
2. $(p \land q) \lor \neg p \supset r$
3. $(p \land r) \lor (\neg q \land p) \lor (\neg r \land \neg p)$

Execute DPLL first without and then with your modification. Then compute how much iterations you saved

The DPLL decision procedure (question)

- Assume that a literal occurs only positively or only negatively;
- How would you modify the algorithm produced in the previous step to take into account this situation?
- When do you check this information?

Example 23 Consider the following examples

1. $(p \supset q \supset r) \wedge p \wedge q$
2. $(p \wedge q) \supset r \wedge (p \supset r)$
3. $(p \wedge \neg r) \vee (q \wedge p) \vee (\neg r \wedge q)$

Execute DPLL first without and then with your modification. Then compute how much iterations you saved

The DPLL decision procedure (question)

- Assume that you count the number of time each single literal occurs in a formula;
- How would you modify the algorithm produced in the previous step to take into account this additional information?
- When do you compute this information?

Example 24 Consider the following examples

1. $(p \supset q \supset r) \wedge p \wedge \neg q$
2. $(p \wedge q) \vee \neg p \supset r$
3. $(p \wedge r) \vee (\neg q \wedge p) \vee (\neg r \wedge \neg p)$

Execute DPLL first without and then with your modification. Then compute how much iterations you saved

The DPLL decision procedure - final (notion)

Algorithm DPLL

Input: A set of clauses ϕ.

Output: A truth value indicating whether ϕ is satisfiable.

function DPLL(ϕ)

while there is a unit clause $\{l\}$ in ϕ do

$\phi \leftarrow$ unit-propagate(l, ϕ);

while there is a literal l that occurs pure in ϕ do

$\phi \leftarrow$ pure-literal-assign(l, ϕ);

if ϕ is empty then return true;

if ϕ contains an empty clause then return false;

$l \leftarrow$ select-literal(ϕ);

DPLL($\phi \wedge \{l\}$) or **DPLL**($\phi \wedge \{\neg(l)\}$);

2-Propositional Logic

2.3-The DPLL decision procedure

UNIVERSITÀ
DI TRENTO

DataScientia
Unitas per Varietatem