## Logik für Informatiker Logic for computer scientists

Till Mossakowski

WiSe 2013/14

# Resolution

### Recall: Conjunctive Normal Form (CNF)

For each propositional sentence, there is an equivalent sentence of form

$$(\varphi_{1,1} \vee \ldots \vee \varphi_{1,m_1}) \wedge \ldots \wedge (\varphi_{n,1} \vee \ldots \vee \varphi_{n,m_n}) \ (n \ge 1, m_i \ge 1)$$

where the  $\varphi_{i,j}$  are literals, i.e. atomic sentences or negations of atomic sentences.

Note that *n* may be 1, e.g.  $A \lor B$  is in CNF. Note that the  $m_i$  may be 1, e.g. *A* as well as  $A \land B$  are in CNF.

A sentence in CNF is called a Horn sentence, if each disjunction of literals contains at most one positive literal.

 $\neg Home(claire) \land (\neg Home(max) \lor Happy(carl))$  $Home(claire) \land Home(max) \land \neg Home(carl)$  $Home(claire) \lor \neg Home(max) \lor \neg Home(carl)$  $Home(claire) \land Home(max) \land$  $(\neg Home(max) \lor \neg Home(max))$ 

## $\neg$ Home(claire) $\land$ (Home(max) $\lor$ Happy(carl)) (Home(claire) $\lor$ Home(max) $\lor \neg$ Happy(claire)) $\land$ Happy(carl)

 $Home(claire) \lor (Home(max) \lor \neg Home(carl))$ 

Resolution

# Alternative notation for the conjuncts in Horn sentences

$$\neg A_1 \lor \ldots \lor \neg A_n \lor B \quad \Leftrightarrow \quad (A_1 \land \ldots \land A_n) \to B$$
$$\neg A_1 \lor \ldots \lor \neg A_n \quad \Leftrightarrow \quad (A_1 \land \ldots \land A_n) \to \bot$$
$$B \qquad \Leftrightarrow \qquad \top \to B$$
$$\bot \qquad \Leftrightarrow \qquad \Box$$

Any Horn sentence is equivalent to a conjunction of conditional statements of the above four forms.

### Satisfaction algorithm for Horn sentences

- For any conjunct  $\top \rightarrow B$ , assign TRUE to B.
- ② If for some conjunct  $(A_1 \land ... \land A_n) \rightarrow B$ , you have assigned TRUE to  $A_1, ..., A_n$  then assign TRUE to B.
- Seperat step 2 as often as possible.
- If there is some conjunct (A<sub>1</sub> ∧ ... ∧ A<sub>n</sub>) → ⊥ with TRUE assigned to A<sub>1</sub>,..., A<sub>n</sub>, the Horn sentence is not satisfiable. Otherwise, assigning FALSE to the yet unassigned atomic sentences makes all the conditionals (and hence also the Horn sentence) true.

### Correctness of the satisfaction algorithm

Theorem The algorithm for the satisfiability of Horn sentences is correct, in that it classifies as tt-satisfiable exactly the tt-satisfiable Horn sentences.

To ask whether this database entails B, Prolog adds  $\perp \leftarrow B$  and runs the Horn algorithm. If the algorithm fails, Prolog answers "yes", otherwise "no".



A clause is a finite set of literals. Examples:

$$C_1 = \{Small(a), Cube(a), BackOf(b, a)\}$$
  

$$C_2 = \{Small(a), Cube(b)\}$$
  

$$C_3 = \emptyset \text{ (also written } \Box\text{)}$$

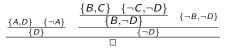
Any set  $\mathcal{T}$  of sentences in CNF can be replaced by an equivalent set  $\mathcal{S}$  of clauses: each conjunct leads to a clause.

A clause R is a resolvent of clauses  $C_1, C_2$  if there is an atomic sentence A with  $A \in C_1$  and  $(\neg A) \in C_2$ , such that

 $R = (C_1 \setminus \{A\}) \cup (C_2 \setminus \{\neg A\}).$ 

**Resolution algorithm**: Given a set S of clauses, systematically add resolvents. If you add  $\Box$  at some point, then S is not satisfiable. Otherwise (i.e. if no further resolution steps are possible and  $\Box$  has not been added), it is satisfiable.

We start with the CNF sentence:  $\neg A \land (B \lor C \lor B) \land (\neg C \lor \neg D) \land (A \lor D) \land (\neg B \lor \neg D)$ In Clause form:  $\{\neg A\}, \{B, C\}, \{\neg C, \neg D\}, \{A, D\}, \{\neg B, \neg D\}$ Apply resolution:



Theorem Resolution is sound and complete. That is, given a set S of clauses, it is possible to arrive at  $\Box$  by successive resolutions if and only if S is not satisfiable.

This gives us an alternative sound and complete proof calculus by putting

#### $\mathcal{T}\vdash S$

iff with resolution, we can obtain  $\square$  from the clausal form of  $\mathcal{T}\cup\{\neg S\}.$