

COMP2411 Lecture 9: An algorithmic approach to reasoning with Horn formulas

Reading: Huth and Ryan, Section 1.5.3

Example: A Medical Knowledge Base

If the weather is cold (C) and the patient has been exposed to influenza virus then the patient will get influenza (I).

If the patient has influenza (I) then the patient will have runny nose (R).

If the patient has influenza (I) then the patient will have a temperature (T).

If the patient is a newborn (N) and has been exposed to the measles virus (EM) then the patient will get measles (M).

If the patient has measles (M) then the patient will have a temperature (T).

If the patient has measles then the patient will have spots. (S)

If the patient has a temperature (T) and the weather is cold (C) then the patient should have a hot rum drink (D).

The knowledge base as a set of Horn clauses

- 1: $C \wedge EF \longrightarrow I$ 2: $I \longrightarrow R$
3: $I \longrightarrow T$ 4: $N \wedge EM \longrightarrow M$
5: $M \longrightarrow T$ 6: $M \longrightarrow S$
7: $T \wedge C \longrightarrow D$

Some facts about the patient, also Horn Clauses:

- 8: $\top \longrightarrow C$
9: $\top \longrightarrow EF$

The query, “Should the patient drink rum?” as a Horn clause:

- 10: $D \longrightarrow \perp$

An efficient algorithm for satisfiability of Horn Clauses

Input: a set S of Horn clauses.

output: An answer to the question “is S satisfiable?”

marked:= $\{p \mid \top \longrightarrow p \text{ is a clause in } S\}$

while there exist clauses $p_1 \wedge \dots \wedge p_n \longrightarrow q$ in S such that
all the p_i are in **marked**, but q is not in **marked**,
do add such q to **marked**
 if $\perp \in$ **marked** **then** return “unsatisfiable”
 end do
return “satisfiable”

Applying the algorithm to the medical knowledge-base:

(Initialize) $\text{marked} := \{C, EF\}$

Loop:

1. While test: identifies clause 1
 $\text{marked} := \{C, EF, I\}$
2. While test: identifies clause 2,3
 $\text{marked} := \{C, EF, I, R, T\}$
3. While test: identifies clause 7
 $\text{marked} := \{C, EF, I, R, T, D\}$
4. While test: identifies clause 10
 $\text{marked} := \{C, EF, I, R, T, D, \perp\}$
if test evaluates to true, return “unsatisfiable”

Theorem:

1. If n is the number of atoms in S , the algorithm always terminates with no more than n passes through the while loop.
2. The algorithm returns “satisfiable” iff S is satisfiable.

A rough calculation of the number of steps of computation taken by the algorithm: ($|S|$ is the size of the input, in number of symbols.)

- Initialization $\approx |S|$ (need to scan input)
- Loop test $\approx |S|$ (scan through all clauses)
- DO section: $\approx |S| + 1$ (could need to add about $|S|$ atoms)

Time needed for algorithm

$$\begin{aligned} &\approx \text{Initialization} + \text{No of iterations} * (\text{Loop test} + \text{DO section}) \\ &\approx |S| + |S|(|S| + |S| + 1) \\ &= 2|S|^2 + |S| \end{aligned}$$

Remarks

1. This is better than the exponential amount of computation that seems to be required for deciding satisfiability of a set of clauses in general.
2. This result is for a *special case* (Horn Clauses) of the satisfiability problem. It does not prove that the general problem can be solved with complexity $2|S|^2 + |S|$!