10

A formula composed from boolean variables and the boolean connectives $\neg$, $\vee$, $\wedge$, $\rightarrow$ and $\leftrightarrow$ is called a **propositional formula**

The problem to determine whether a given propositional formula is satisfiable is called

SAT(isfiability)

So the method of truth tables is a method for SAT

However, the complexity of this method is always exponential in the number of variables, by which the method is unsuitable for formulas over many variables

Many practical problems can be expressed as SAT-problems over hundreds or thousands of variables

Hence we need other methods than truth tables

43

Resolution

This is a method for SAT, being the basis of the best current SAT-solvers

Resolution is only applicable to formulas of a particular shape, namely CNF

A **conjunctive normal form (CNF)** is a conjunction of clauses

A **clause** is a disjunction of literals

A **literal** is either a variable or the negation of a variable

Hence a CNF is of the shape

$$\bigwedge_i (\bigvee_j \ell_{ij})$$

where ℓ_{ij} are literals

For example, the pigeon hole formula PF_n is a CNF

44

Arbitrary formulas can be transformed to CNFs in a clever way maintaining satisfiability

This makes resolution applicable to arbitrary formulas

Basic idea of resolution:

Add new clauses in such a way that the conjunction of all clauses remains equivalent to the original CNF

The empty clause $\perp$ is equivalent to *false*

If the clause $\perp$ is created in the resolution process then the conjunction of all clauses is equivalent to *false*, and hence the same holds for the original CNF

45

Intuitively:

Clauses are properties that you know to be true

From these clauses you derive new clauses, trying to derive a contradiction: the empty clause

Surprisingly here we need only one rule, the **resolution rule**

This rule states that if there are clauses of the shape $V \vee p$ and $W \vee \neg p$, then the new clause $V \vee W$ may be added

This is correct since

$$(V \vee p) \wedge (W \vee \neg p) \Rightarrow (V \vee W)$$

(apply case analysis p and $\neg p$)

46

Order of literals in a clause does not play a role

Double occurrences of literals will be removed

Think of a clause as a **set** of literals

Think of a CNF as a **set** of clauses

Example

We prove that

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

is unsatisfiable

47

1	$p \vee q$	
2	$\neg r \vee s$	
3	$\neg q \vee r$	
4	$\neg r \vee \neg s$	
5	$\neg p \vee r$	
6	$p \vee r$	$(1, 3, q)$
7	r	$(5, 6, p)$
8	s	$(2, 7, r)$
9	$\neg r$	$(4, 8, s)$
10	$\perp$	$(7, 9, r)$

Remarks:

- Lot of freedom in choice

  Other first steps in the example could have been $(3, 4, r)$ or $(2, 4, s)$ or $(1, 5, p)$ or $\cdots$

- Resolution steps in which V contains q and W contains $\neg q$ for some q (or conversely) are allowed but useless

  In that case the new clause $V \vee W$ is of the shape $q \vee \neg q \vee \cdots$ and hence equivalent to *true*, not containing fruitful information

- If a clause consists of a single literal ℓ then by the resolution rule the literal $\neg\ell$ may be removed from every clause containing $\neg\ell$

  This is called **unit resolution**

57

A strategy that can always be applied is

Davis-Putnam's procedure (1960):

Repeat until either no clauses are left or the empty clause has been derived:

> Choose a variable p
>
> Apply resolution on every pair of clauses for which the one contains p and the other contains $\neg p$
>
> Remove all clauses containing both q and $\neg q$ for some q
>
> Remove all clauses containing either p or $\neg p$

58

Example:

Consider the CNF consisting of the following nine clauses

$$\begin{array}{lll} \neg p \vee \neg s & p \vee r & \neg s \vee t \\ \neg p \vee \neg r & s \vee p & q \vee s \\ \neg q \vee \neg t & r \vee t & \neg r \vee q \end{array}$$

First do all resolution steps w.r.t. p

After removing all clauses containing p, $\neg p$ or the shape $A \vee \neg A$ the following clauses remain:

$$\underbrace{r \vee \neg s,\ \ \neg r \vee s}_{\text{new}},\ \ \neg s \vee t,\ \ q \vee s,\ \ \neg q \vee \neg t,\ \ r \vee t,\ \ \neg r \vee q$$

Next do all resolution steps w.r.t. q

After removing all clauses containing q, $\neg q$ the following clauses remain:

$$r \vee \neg s,\ \ \neg r \vee s,\ \ \neg s \vee t,\ \ r \vee t,\ \ \underbrace{s \vee \neg t,\ \ \neg r \vee \neg t}_{\text{new}}$$

59

Next do all resolution steps w.r.t. r

After removing all clauses containing r, $\neg r$ or the shape $A \vee \neg A$ the following clauses remain:

$$\underbrace{\neg t \vee \neg s,\ \ s \vee t}_{\text{new}},\ \ \neg s \vee t,\ \ s \vee \neg t$$

Next do all resolution steps w.r.t. s

After removing all clauses containing s, $\neg s$ or the shape $A \vee \neg A$ the following clauses remain:

$$t,\ \ \neg t$$

Finally we do resolution on t and obtain the empty clause, proving that the original CNF is unsatisfiable

66

Summarizing Davis-Putnam's procedure:

- Procedure to establish satisfiability of any CNF
- Complete: it always ends and always gives the right answer
- One long repeat loop doing at most n steps if there are n variables
- In every step of the loop clauses are added and removed
- Worst case exponential: intermediate CNF may blow up exponentially (and often does in practice $\cdots$)
- By keeping intermediate CNFs in case of satisfiability a satisfying assignment can be constructed from the run of the procedure, as we show by an example

67

$$\begin{array}{cc}
p \vee q,\ \neg p \vee \neg q,\ p \vee \neg q,\ \neg p \vee r & \\
\downarrow p & \\
q \vee r,\ \neg q,\ \neg q \vee r & \\
\downarrow q & \\
r & r = \mathit{true} \\
\downarrow r & \\
\{\} &
\end{array}$$

find value for the last variable that was removed

68

$$\begin{array}{cc}
p \vee q,\ \neg p \vee \neg q,\ p \vee \neg q,\ \neg p \vee r & \\
\downarrow p & \\
q \vee r,\ \neg q,\ \neg q \vee r & q = \mathit{false} \\
\downarrow q & \uparrow \\
r & r = \mathit{true} \\
\downarrow r & \\
\{\} &
\end{array}$$

evaluate in formula, find next value

69

$$
\begin{array}{cc}
p \lor q,\ \neg p \lor \neg q,\ p \lor \neg q,\ \neg p \lor r & p = \mathit{true} \\
\downarrow p & \uparrow \\
q \lor r,\ \neg q,\ \neg q \lor r & q = \mathit{false} \\
\downarrow q & \uparrow \\
r & r = \mathit{true} \\
\downarrow r & \\
\{\} &
\end{array}
$$

evaluate all values in formula, find next value, until finished