

Resolution

Note Title

Topics today: (1) The resolution rule, (2) A resolution algorithm

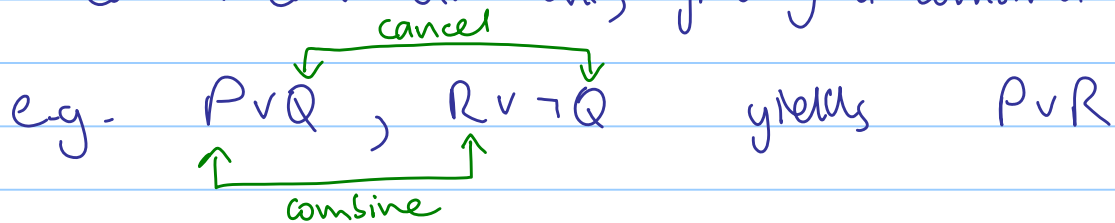
(3) Efficiency of SAT solving

(1) Revision from last time: The Resolution inference rule

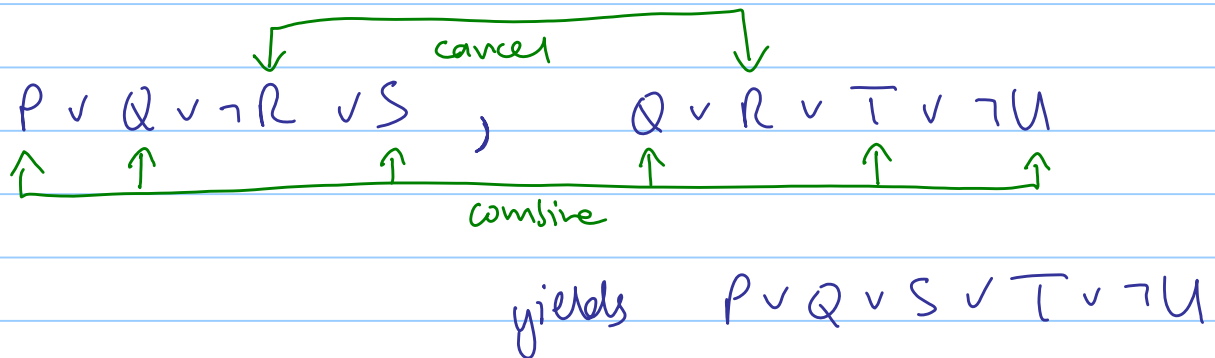
Resolution is an important inference rule.

Basic idea is that opposite literals in separate clauses cancel each other out, yielding a combined clause.

eg. $P \vee Q, R \vee \neg Q$ yields $P \vee R$



$P \vee Q \vee \neg R \vee S, Q \vee R \vee T \vee \neg U$ yields $P \vee Q \vee S \vee T \vee \neg U$



Exercise: Apply resolution to the KB
 $\{P \vee \neg Q, \neg P \vee R \vee \neg S, S \vee T\}$

The resolution rule is important because it can be used as part of an algorithm that infers entailment. i.e. it decides whether $KB \models \alpha$ for any KB, α .

We study this next.

2 Our Simple resolution algorithm for entailment

We want to determine whether $KB \models \alpha$.
Equivalently, is $KB \Rightarrow \alpha$ valid?
Equivalently, is $KB \wedge \neg \alpha$ unsatisfiable?

This alg is guaranteed to terminate. The book has a proof, but we don't study it.

- Algorithm:
- Convert $KB \wedge \neg \alpha$ to CNF
 - Apply resolution repeatedly
 - If you ever get an empty clause, conclude that $KB \models \alpha$.
 - If can't make any more clauses, conclude that $KB \not\models \alpha$.

Why? Because you've derived the empty clause, equiv to 'false', meaning $KB \wedge \neg \alpha$ is unsatisfiable

Why? Because you can now satisfy $KB \wedge \neg \alpha$. Detailed proof in book (not required) but basically just fill in the values.

Exercise:

$$KB = \{ P \Rightarrow Q, Q \vee R \vee S, S \Rightarrow P \vee Q \}$$

- (i) Does KB entail $Q \vee R$?
- (ii) Does KB entail $\neg Q \wedge S$?

③ Efficiency of SAT-solvers

- Note that the resolution algorithm above is just a particular method of determining satisfiability, also known as SAT-solving.
- Satisfiability (or just "SAT") is of central importance in the theory of algorithms. It was the first problem to be proved NP-complete, in the early 1970s.
 - NP-complete means:
 - no efficient algorithm is known to solve all instances
 - if we did find an efficient alg, that alg would solve most other "hard" problems in CS
 - therefore, efficient alg for all instances probably doesn't exist.
(worst case)

i.e. polynomial time
- So, our resolution algorithm takes exponential time (in the number of variables and/or clauses)
- Better algorithms are known (e.g. Davis-Putnam), but still exponential in the worst case
- Still, "modern solvers handle problems with tens of millions of variables" - see last paragraph of book 7.6-1, p262.

- An interesting special case where a linear time solution exists: if the KB consists completely of Horn clauses

→ clause with at most one positive literal
eg. $\neg B \vee \neg C \vee D$, $\neg P \vee \neg Q$