Testing satisfiability: resolution method

- Compactness: an inf. set of wffs S is sat'able iff every finite $S_0 \subseteq S$ is sat'able.
  - ↳ Q: is there an algorithm for testing whether a given $S_0 = \{A_1, \ldots, A_n\}$ is sat'able?
  - Note: S_0 sat'able iff $A = A_1 \wedge \cdots \wedge A_n$ is sat'able
  - ↳ Q: is there an algorithm for testing whether a given A is sat'able?
  - Ans: Yes: compute truth table for A

- Problem is: for wffs A w/ lots of variables, computing truth table is inefficient
  - ↳ Q: is there a more efficient test of satisfiability?
  - Ans: yes! ... at least sometimes called "method of resolution"

- method only works on A's in CNF:
  $$A = (\ell_{11} \vee \cdots \vee \ell_{1n_1}) \wedge \cdots \wedge (\ell_{k1} \vee \cdots \vee \ell_{kn_k})$$

- on HW4 you'll show: given A, can efficiently find A^* in CNF s.t.

A is satisfiable iff A^* is taut

(A^* need not be logically equiv to A)

- so for now we'll work w/ A's in cnf:
  $A = (l_{11} \vee \cdots \vee l_{1n_1}) \wedge \cdots \wedge (l_{k1} \vee \cdots \vee l_{kn_k})$
  recall: each l_{ij} a literal: either p or $\neg p$ for some p

- A is sat'able iff there is a valuation v satisfying every conjunct $(l_{i1} \vee \cdots \vee l_{in_i})$
- write $c_i = \{l_{i1}, \ldots, l_{in_i}\}$ for set of literals in ith conjunct
  c_i is called a "clause"
- write $C = \{c_1, \ldots, c_k\} = \{\{l_{11}, \ldots, l_{1n_1}\}, \ldots, \{l_{k1}, \ldots, l_{kn_k}\}\}$ for set of clauses

- note: A is sat'able iff there is v making every conjunct true
  iff there is v s.t. for every i, $v(l_{ij}) = 1$ for at least one literal $l_{ij} \in c_i$

Ex: if $A = (p_1 \vee p_2 \vee \neg p_3) \wedge (p_3 \vee p_4)$ ← in cnf
then $c_1 = \{p_1, p_2, \neg p_3\}$, $c_2 = \{p_3, p_4\}$
$C = \{c_1, c_2\} = \{\{p_1, p_2, \neg p_3\}, \{p_3, p_4\}\}$

- testing sat'ability of A in cnf reduces to testing sat'ability of C.

- now we just work w/ sets of clauses C
- we allow $C = \{c_1, c_2, \ldots\}$ to be infinite (tho if C corresponds to a wff A, will be finite)
- method of resolution applies to such C.

<u>Notation</u>: - for a clause $c = \{\ell_1, \ldots, \ell_n\}$ write $v \models c$ if $v \models \ell_1 \vee \ldots \vee \ell_n$

- for a set of clauses $C = \{c_1, c_2, \ldots\}$ write $v \models C$ iff $v \models c_i$ for all i
- for u a literal, write $\bar{u}$ for <u>conjugate</u> of u, i.e.

if $u = p$ $\quad \bar{u} = \neg p$

if $u = \neg p$ $\quad \bar{u} = p$

<u>Def'n</u> Sps c, c_1, c_2 are clauses. Say: c is a <u>resolvent</u> of c_1, c_2 if there is a literal u s.t.

$u \in c_1$, $\bar{u} \in c_2$, and

$$c = (c_1 \setminus \{u\}) \cup (c_2 \setminus \{\bar{u}\})$$

Write:

$$\begin{array}{ccc} & c & \\ \diagup & & \diagdown \\ c_1 & & c_2 \end{array}$$

<u>Note</u>: we allow c to be the <u>empty clause</u> $\emptyset$, also denoted $\square$

↳ by convention, $\square$ is unsatisfiable

(why: a clause c is satisfied iff one of its literals is true — $\square$ has no literals!)

<u>Ex:</u> (i)

$$\begin{tikzcd}[column sep=small] & \{p, r\} & \\ \{p, \neg q, r\} \arrow[ur, no head] & & \{q, r\} \arrow[ul, no head] \end{tikzcd}$$

$\{p, r\} = \{p, r\} \cup \{r\}$ ← removed $\neg q$ ← removed q

(ii)

$$\begin{tikzcd}[column sep=small] & \{q, \neg q\} & \\ \{p, \neg q\} \arrow[ur, no head] & & \{\neg p, q\} \arrow[ul, no head] \end{tikzcd}$$

<u>also</u>:

$$\begin{tikzcd}[column sep=small] & \{p, \neg p\} & \\ \{p, \neg q\} \arrow[ur, no head] & & \{\neg p, q\} \arrow[ul, no head] \end{tikzcd}$$

(two diff. resolutions)

(iii)

$$\begin{tikzcd}[column sep=small] & \square & \\ \{p\} \arrow[ur, no head] & & \{\neg p\} \arrow[ul, no head] \end{tikzcd}$$

<u>Prop'n</u> if c is resolvent of $\{c_1, c_2\}$ then $\{c_1, c_2\} \models c$

<u>Pf</u>: Sps $u \in c_1$ and $\bar{u} \in c_2$ are literals removed in the resolution and $v \models \{c_1, c_2\}$

— if $v(u) = 1$ (so $v(\bar{u}) = 0$) then must have $v(w) = 1$ since $v \models c_2$ and $v(\bar{u}) = 0$ for some $w \in c_2 \setminus \{\bar{u}\}$

— if $v(\bar{u}) = 1$ must have $v(w) = 1$ for some $w \in c_1 \setminus \{u\}$

— in either case: $v \models c \; (= c_1 \setminus \{u\} \cup c_2 \setminus \{\bar{u}\})$

$$\begin{array}{ccc} & C & \\ \swarrow & & \searrow \\ C_1 & & C_2 \end{array}$$

Ex Sps $C_1 = \{p, \neg q, r\}$, $C_2 = \{q, r\}$

$C = \{p, r\}$

if $v \models C_1, C_2$ and $v(q) = 1$ must have either $v(p) = 1$ or $v(r) = 1$

so $v \models C$

if $v(q) = 0$ must have $v(r) = 1$

so again $v \models C$.

Def'n Sps C is a set of clauses (poss. infinite)

A proof by resolution from C is a finite sequence $C_1, C_2, \ldots, C_n$ of clauses from C s.t. for every i

either (i) $C_i \in \mathbf{C}$

or (ii) C_i is a resolvant of some C_j, C_k for $j, k < i$

If $C_n = \square$, we say the proof is a refutation of C

Ex Sps $C = \{\{p, q, \neg r\}, \{\neg p\}, \{p, q, r\}, \{p, \neg q\}\}$

Then:

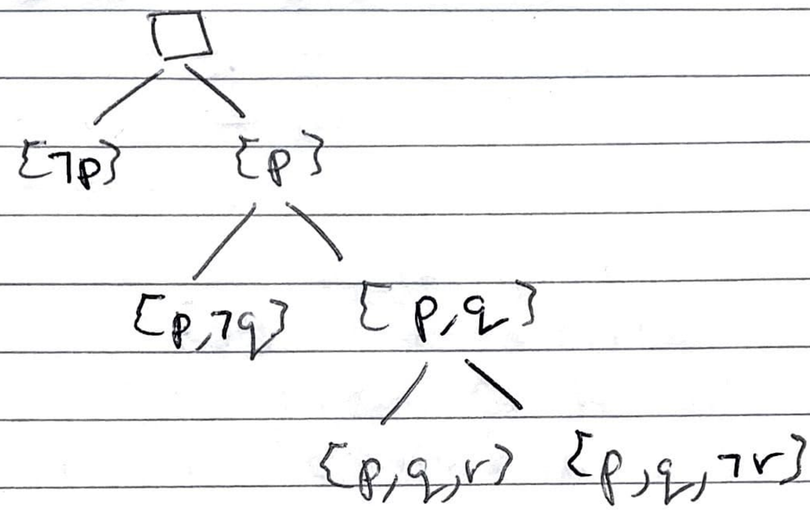
$C_1 =$	$\{p, q, \neg r\}$	in C
$C_2 =$	$\{p, q, r\}$	in C
$C_3 =$	$\{p, q\}$	resolvant of C_1, C_2
$C_4 =$	$\{p, \neg q\}$	in C
$C_5 =$	$\{p\}$	resolvant of C_3, C_4

$c_6 = \{\neg p\}$ in C

$c_7 = \square$ resolvent of c_5, c_6

is a refutation of C

Pic:



Ex: Sps $C = \{\{\neg p, s\}, \{p, \neg q, s\}, \{p, q, \neg r\}, \{p, r, s\}, \{\neg s\}\}$

Then:

$c_1 = \{p, \neg q, s\}$ in C

$c_2 = \{p, q, \neg r\}$ in C

$c_3 = \{p, s, \neg r\}$ c_1, c_2 res.

$c_4 = \{p, r, s\}$ in C

$c_5 = \{p, s\}$ c_3, c_4 res.

$c_6 = \{\neg p, s\}$ in C

$c_7 = \{s\}$ c_5, c_6 res

$c_8 = \{\neg s\}$ in C

$c_9 = \square$

is a refutation of C.

Note: resolution proofs are not unique in general (can you find a different refutation of C above?)

- So: what does a refutation of C show us, anyway?
- that C is unsat'able! ... and in fact the converse is also true.

Theorem Sps $C = \{c_1, c_2, \ldots\}$ is a set of clauses. Then C is unsatisfiable iff there is a refutation of C.

- will need:

Lemma: if $d_1, \ldots, d_n$ is any resolution proof from C, then for all $i \le n$ we have $C \models d_i$ (i.e. if $\tau \models C$ then $\tau \models d_i$)

Pf: induction on i
- if $i = 1$, must have $d_1 \in C$ (can't be a resolvent - nothing before d_1)
- then clearly $\tau \models C$ implies $\tau \models d_1$
- Sps $C \models \{d_1, \ldots, d_i\}$
- if $c_{i+1} \in C$ then $C \models c_{i+1}$ as above
- otherwise c_{i+1} is resolvent of c_j, c_k for some $j, k \le i$
- have $C \models \{c_j, c_k\}$ by IH

- hence $C \models c_{i+1}$ by our prev. prop'n ✓

Pf of theorem: ($\Leftarrow$) Sps $d_1, \dots, d_n = \square$ is a refutation of C
- if there is ν s.t. $\nu \models C$ then by lemma $\nu \models \{d_1, \dots, d_n\}$ hence $\nu \models \square = d_n$, contradiction as $\square$ is unsatisfiable
- hence no such ν, i.e. C unsat'able ✓

($\Rightarrow$) hard direction. WTS: if C unsat'able then there is a refutation of C.

- may assume: C contains no clause c_i containing u and $\bar{u}$ for some literal u
  (such c_i satisfied by any ν, so C sat'able iff $C \setminus \{c_i\}$ is)

- given a literal u, define $C(u) :=$ set of clauses obtained by:
  (i) deleting u from any clauses in C
  (ii) deleting any clauses from C that contain $\bar{u}$.

Ex: if $C = \{\{p, q, \neg r\}, \{p, \neg q\}, \{p, q, r\}, \{q, r\}\}$ and $u = r$, then:

$$C(r) = \{\{p, \neg q\}, \{p, q\}, \{q\}\}$$
$$C(\bar{r}) = \{\{p, q\}, \{p, \neg q\}\}$$

- Observe: if $v(u) = 0$ then $v \models C$ iff $v \models C(u)$
- Also: $u, \bar{u}$ don't occur in $C(u), C(\bar{u})$
- So: if C unsat'able, so are $C(u)$, $C(\bar{u})$

Why: if $v \models C(u)$ define v' by $v' = v$ except $v'(u) = 0$

then $v' \models C(u)$ still (haven't changed truth values: $u, \bar{u}$ don't appear in $C(u)$)

and so $v' \models C$ by above, contradiction

(likewise for $C(\bar{u})$)

- Now: sps C is unsatisfiable
- By compactness there is a finite $C_0 \subseteq C$ unsat'able (view C, C_0 as sets of fmlas by identifying each clause w/ associated disjunction of literals)
- we'll get a refutation from C_0 so may as well assume $C = C_0$ (i.e. C is finite)
- Let $p_1, \ldots, p_n$ be variables appearing in C
- we induct on n

If $n=1$: only p_1 occurs in clauses in C
- since C unsat'able, must have

$C = \{\{p_1\}, \{\neg p_1\}\}$

- but then

$\{p_1\}$
$\{\neg p_1\}$
$\square$ a refutation ✓

- So sps thm. true for all C's involving $p_1, \ldots, p_n$
- Now sps C involves $p_1, \ldots, p_n, p_{n+1}$
- by our observation above: $C(u)$ and $C(\bar{u})$ are also unsat'able, where $u = p_{n+1}$
- also: $C(u)$, $C(\bar{u})$ only involve $p_1, \ldots, p_n$
- by induction, have refutations

$d_1, \ldots, d_m, d_{m+1} = \square$ from $C(u)$
$e_1, \ldots, e_k, e_{k+1} = \square$ from $C(\bar{u})$

- we will turn these into resolution pfs from C
- idea is: for $d_1, \ldots, d_{m+1}$, put $u = p_{n+1}$ back everywhere it was deleted in $C(u)$ and carry it along when resolving

- more explicitly define $d_1', \ldots, d_m', d_{m+1}'$ recursively:

<u>case 1</u>: if $d_i \in C(u)$, then either $d_i \in C$ too, in which case $d_i' = d_i$ or $d_i \cup \{p_{n+1}\} \in C$, in which case $d_i' = d_i \cup \{p_{n+1}\}$

<u>case 2</u>: if d_i a resolvent of

d_j, d_k for $j, k < i$
then this resolution applies to
some var $p_1, .., p_n$ (not p_{n+1})
and also applies to d_j', d_k' (already
defined) to get d_i'.

- then $d_1', .., d_{m+1}'$ is a proof from C
- since $d_{m+1} = \square$ (all of $p_1, .., p_n$
eliminated), must have either $d_{m+1}' =$
$\square$ or $d_{m+1}' = \{p_{n+1}\}$
- if $d_{m+1}' = \square$ done (have a refutation
of C)
- so may assume $d_{m+1}' = \{p_{n+1}\}$

- define $e_1', .., e_{k+1}'$ likewise
(inserting $\bar{a} = \neg p_{n+1}$ wherever deleted)
- then either $e_{k+1}' = \square$ (done) or
$= \{\neg p_{n+1}\}$
- in second case
$d_1', .., \underbrace{d_{m+1}'}_{= \{p_{n+1}\}}, e_1', .., \underbrace{e_{k+1}'}_{= \{\neg p_{n+1}\}}$
is a pf from C
... which we can resolve to get
$\square$, i.e. get a refutation
- thus in all cases we get a
refutation of C ✓

- Theorem expresses a kind of "completeness" for proofs by resolution: if $\mathcal{C}$ is unsatisfiable (semantic notion)

then we can prove it (i.e. get a refutation) (syntactic notion)

Ex: Sps $C = \{\{p, q, \neg r\}, \{\neg p\}, \{p, q, r\}, \{p, \neg q\}\}$

and let $u = r$

then $C(r) = \{\{\neg p\}, \{p, q\}\}$

$C(\bar{r}) = C(\neg r) = \{\{p, q\}, \{\neg p\}, \{p, \neg q\}\}$

idea of theorem is: if we can get refutations from $C(r)$, $C(\bar{r})$, can get a refutation from C:

Ref'n from $C(r)$	Proof in C
$\{p, q\}$	$\{p, q, r\}$
$\{p, \neg q\}$	$\{p, \neg q\}$
$\{p\}$	$\{p, r\}$
$\{\neg p\}$	$\{\neg p\}$
$\{\}$	$\{r\}$

Ref'n from $C(\bar{r})$	Proof in C
$\{p, q\}$	$\{p, q, \neg r\}$
$\{p, \neg q\}$	$\{p, \neg q\}$

$\{p\}$ $\qquad\qquad$ $\{p, \neg r\}$
$\{\neg p\}$ $\qquad\qquad$ $\{\neg p\}$
$\Box$ $\qquad\qquad$ $\{\neg r\}$

Combining the righthand proofs (in C) and resolving $\{r\}, \{\neg r\} \to \Box$ gives a refutation of C!

Hillbert style formal proofs:

- can view resolution as a method of proof
- ie. a method for showing (given S, A) that $S \models A$

- point is: $S \models A$ iff there is $S_0 = \{A_1, \ldots, A_n\}$ s.t. $S_0 \models A$ (compactness v2)
  iff $A_1 \wedge \ldots \wedge A_n \models A$
  iff $A_1 \wedge \ldots \wedge A_n \wedge \neg A$ is unsat'able
  iff A^* is unsat'able

  in cnf and sat'able iff $A_1 \wedge \ldots \wedge A_n \wedge \neg A$ is (HW4)

  iff there is a refutation of A^* using resolution (last time)

- method only applies to cnf fmlas
- one "deduction rule": resolution.