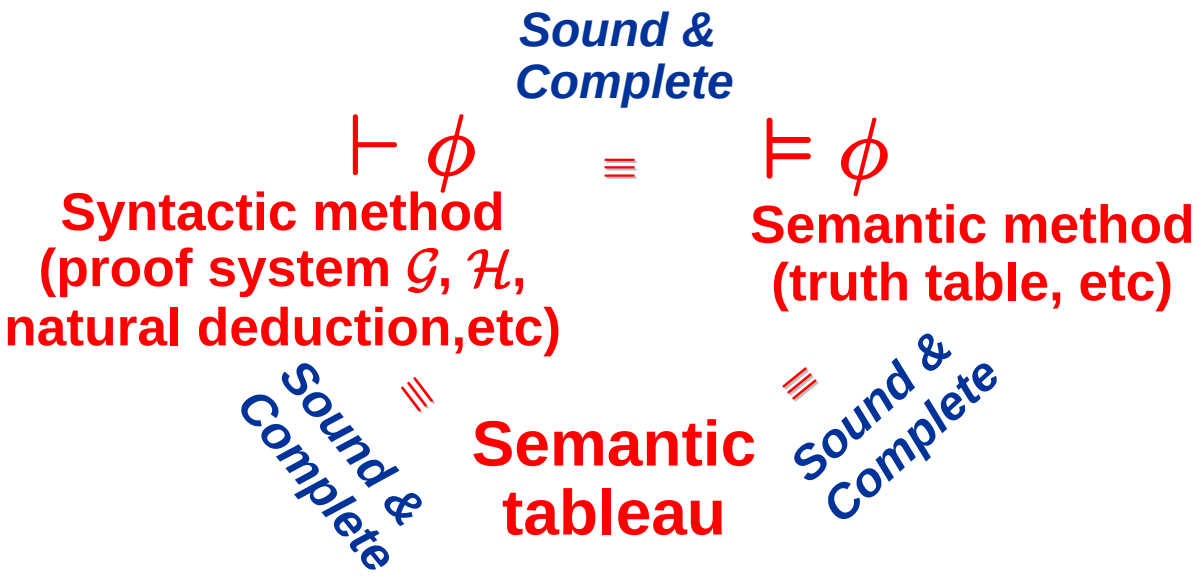
Propositional Calculus
-*Variant forms of the deductive systems*
- Resolution (1/2)

Moonzoo Kim
CS Division of EECS Dept.
KAIST

moonzoo@cs.kaist.ac.kr
http://pswlab.kaist.ac.kr/courses/cs402-07

Review

- Goal of logic
  - To check whether given a formula ϕ is valid
  - To prove a given formula ϕ



Variants of $\mathcal{H}$ (1/2)

- Variant Hilbert systems almost invariably have MP as the only rule. They differ in the choice of primitive operators and axioms
- $\mathcal{H}$' replace Axiom 3 by
  - Axiom 3' $\vdash (\neg B \rightarrow \neg A) \rightarrow ((\neg B \rightarrow A) \rightarrow B)$
- Thm 3.44 $\mathcal{H}$ and $\mathcal{H}$' are equivalent
  - A proof of Axiom 3' in $\mathcal{H}$
  - The other direction?

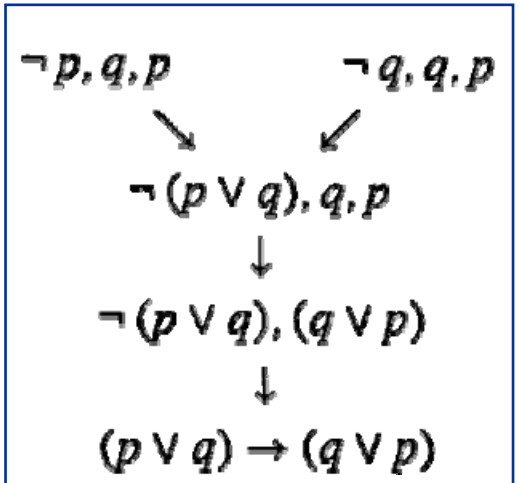
1.	$\{\neg B \rightarrow \neg A, \neg B \rightarrow A, \neg B\} \vdash \neg B$	Assumption
2.	$\{\neg B \rightarrow \neg A, \neg B \rightarrow A, \neg B\} \vdash \neg B \rightarrow A$	Assumption
3.	$\{\neg B \rightarrow \neg A, \neg B \rightarrow A, \neg B\} \vdash A$	MP 1,2
4.	$\{\neg B \rightarrow \neg A, \neg B \rightarrow A, \neg B\} \vdash \neg B \rightarrow \neg A$	Assumption
5.	$\{\neg B \rightarrow \neg A, \neg B \rightarrow A, \neg B\} \vdash A \rightarrow B$	Contrapositive 4
6.	$\{\neg B \rightarrow \neg A, \neg B \rightarrow A, \neg B\} \vdash B$	MP 3,5
7.	$\{\neg B \rightarrow \neg A, \neg B \rightarrow A\} \vdash \neg B \rightarrow B$	Deduction 7
8.	$\{\neg B \rightarrow \neg A, \neg B \rightarrow A\} \vdash (\neg B \rightarrow B) \rightarrow B$	Theorem 3.29
9.	$\{\neg B \rightarrow \neg A, \neg B \rightarrow A\} \vdash B$	MP 8,9
10.	$\{\neg B \rightarrow \neg A\} \vdash (\neg B \rightarrow A) \rightarrow B$	Deduction 9
11.	$\vdash (\neg B \rightarrow \neg A) \rightarrow ((\neg B \rightarrow A) \rightarrow B)$	Deduction 10

Variants of $\mathcal{H}$ (2/2)

- $\mathcal{H}$" has the same MP rule but a set of axioms as
  - Axiom 1 $\vdash A \vee A \rightarrow A$
  - Axiom 2 $\vdash A \rightarrow A \vee B$
  - Axiom 3 $\vdash A \vee B \rightarrow B \vee A$
  - Axiom 4 $\vdash B \rightarrow C \rightarrow (A \vee B \rightarrow A \vee C)$
  - Note that it is also possible to consider $\vee$ as the primitive binary operator. Then, $\rightarrow$ is defined by $\neg A \vee B$.
- Yet another variant of Hilbert system $\mathcal{H}$"' has only one axiom with MP
  - Meredith's axiom
    - $(\{[A \rightarrow B) \rightarrow (\neg C \rightarrow \neg D)] \rightarrow C\} \rightarrow E) \rightarrow [(E \rightarrow A) \rightarrow (D \rightarrow A)]$

Subformula property

- Def 3.48 A deductive system has the subformula property if any formula appearing in a proof of A is either a subformula of A or the negation of a subformula of A
- $\mathcal{G}$ has the subformula property while $\mathcal{H}$ obviously does not since MP 'erase' formulas
  - That is why a proof in $\mathcal{H}$ is harder than a proof in $\mathcal{G}$
- If a deductive system has the subformula property, then mechanical proof may be possible since there exists only
  - there exist only a finite number of subformulas for a finite formula ϕ
  - there exist only a finite number of inference rules



A proof of $(p \vee q) \rightarrow (q \vee p)$ in $\mathcal{G}$

Automated proof

- One desirable property of a deductive system is to generate an automated/mechanical proof
  - We have decision procedure to check validity of a propositional formula automatically (i.e., truth table and semantic tableau)
    - Note that decision procedure requires knowledge on all interpretations (i.e., infinite number of interpretations in general) which is not feasible except propositional logic
- A deductive proof requires only a finite set of sets of formulas, because a deductive proof system analyzes the target formula only, not its interpretations.
  - Many research works to develop (semi)automated theorem prover
- No obvious connection between the formula and its proof in $\mathcal{H}$ makes a proof in $\mathcal{H}$ difficult ($\equiv$ no mechanical proof)
  - A human being has to rely on his/her brain to select proper axioms

Resolution

- The resolution method was invented by J.A.Robinson in 1965
- The resolution method is frequently an efficient method for searching for a proof.
  - The resolution method uses only syntactic information of the target formula in CNF to check its unsatisfiability
- Def 4.1 A formula is conjunctive normal form (CNF) iff it is a conjunction of disjunctions of literals (atoms or negated atoms/propositions).
  - ex. $(\neg p \vee q \vee r) \wedge (\neg q \vee r) \wedge (\neg r)$ is in CNF
  - ex. $(\neg p \vee q \vee r) \wedge ((p \wedge \neg q) \vee r) \wedge (\neg r)$ is not in CNF
  - ex. $(\neg p \vee q \vee r) \wedge \neg (\neg q \vee r) \wedge (\neg r)$ is not in CNF

Transformation into CNF

- Thm 4.3 Every formula in the propositional calculus can be transformed into an equivalent formula in CNF
  1. Eliminate all operators except for $\neg$, $\vee$, $\wedge$ using the equivalence in Figure 2.6
  2. Push all negations inward using De Morgan's laws
     - $\neg (A \wedge B) \equiv (\neg A \vee \neg B)$
     - $\neg (A \vee B) \equiv (\neg A \wedge \neg B)$
  3. Eliminate double negation ($\neg\neg A \equiv A$)
  4. The formula now consists of $\vee$ and $\wedge$ of literals. Use the distributed laws to eliminate $\wedge$ within $\vee$
     - $A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$
     - $(A \wedge B) \vee C \equiv (A \vee B) \wedge (A \vee C)$

Notations

- Def 4.5
  - A clause is a set of literals which is considered to be an implicit disjunction
    - ex. {p, q, ¬r } ≡ p ∨ q ∨ ¬r
  - A unit clause is a clause consisting of exactly one literal
    - ex. {¬ r}
  - A formula in clausal form is a set of clauses which is considered to be an implicit conjunction
    - ex. {{¬q, ¬p, q}, {p, ¬q, q, ¬p}} ≡ (¬q ∨ ¬p ∨ q) ∧ (p ∨ ¬q ∨ q ∨ p ∨ ¬p)
- We use an abbreviated notation,
  - removing the set delimiters '{' and '}' from a clause
  - using ' to denote negation
  - ex. {q'p'q, pp'q} ≡ {{¬q, ¬p, q}, {p, ¬p, q}} ≡ (¬q ∨ ¬p ∨ q) ∧ (p ∨ ¬q ∨ q ∨ p ∨ ¬p)
- *S* for a set of clauses (that is, a formula in clausal form)
- *C* for a clause and *l* for a literal. l^c is its complement
  - ex. Let l be p, then l^c is p'. Let l be p', then l^c is p.

The resolution rule

- Resolution is a refutation procedure which is used to show that a formula in clausal form is unsatisfiable
- The resolution procedure consists of a sequence of applications of the resolution rule to a set of clauses
  - The rule maintains satisfiability: if a set of clauses is satisfiable, so is the set of clauses produced by an application of the rule
  - Therefore, if the unsatisfiable empty clause (called refutation □) is ever obtained, the original set of clauses must have been unsatisfiable

The resolution rule

- A given a formula $\phi = C_1 \wedge C_2$, we add a new clause (say R_1) obtained by analyzing C_1 and C_2 to ϕ s.t. satisfiability of $C_1 \wedge C_2$ is same to the satisfiability of $C_1 \wedge C_2 \wedge R_1$
  - If R_1 is unsatisfiable, ϕ is unsatisfiable
  - Otherwise, we repeat the same procedure to $C_1 \wedge C_2 \wedge R_1$ and add another new clause R_2 (i.e. now our target formula $C_1 \wedge C_2 \wedge R_1 \wedge R_2$) and so on until we find an unsatisfiable R_i
- Rule 4.22 Let C_1, C_2 be clauses s.t. $l \in C_1$, $l^c \in C_2$. The clauses C_1, C_2 are said to be clashing clauses and to clash on the complementary literals l, l^c. C, the resolvent of C_1 and C_2 is the clause:
  - $Res(C_1,C_2) = (C_1 - \{l\}) \cup (C_2 - \{l^c\})$

Example

- Let S be the set of clause {p, p'q, r', p'q'r}
  - I.e. a target formula $\phi = p \wedge (\neg p \vee q) \wedge \neg r \wedge (\neg p \vee \neg q \vee r)$
- The final formula obtained through resolution rule is
  - $p \wedge (\neg p \vee q) \wedge \neg r \wedge (\neg p \vee \neg q \vee r) \wedge (\neg p \vee \neg q) \wedge \neg p \wedge \square$
  - We obtain a refutation $\square$, thus ϕ is unsatisfiable