FUNDAMENTALS OF LOGIC
NO.12 RESOLUTION PRINCIPLE

Tatsuya Hagino

hagino@sfc.keio.ac.jp

lecture URL

https://vu5.sfc.keio.ac.jp/slide/

So Far

- Propositional Logic
  - Logical connectives ($\wedge$, $\vee$, $\rightarrow$, $\neg$)
  - Truth table
  - Tautology
  - Normal form
  - Axiom and theorem
  - LK framework
  - Soundness and completeness

- Predicate Logic
  - Logical Formulas (language, term)
  - Quantifiers ($\forall x\ P(x)$, $\exists x\ P(x)$)
  - Closed formulae (bound and free variables)
  - Semantics of predicate logic (domain, interpretation, structure)
  - Valid formulae
  - Prenex formulae
  - LK framework for predicate logic
  - Soundness and completeness
  - Skolemization
  - Herbrand Theorem

Skolemization and Herbrand Theorem

- *Skolemization*
  - The followings are equivalent:
    - $\forall x_1 \cdots \forall x_n \exists y\, A$ is satisfiable.
    - $\forall x_1 \cdots \forall x_n\, A[f(x_1, \dots, x_n)/y]$ is satisfiable.
  - To check the satisfiability of a formula, it can be transformed into $\forall x_1 \cdots \forall x_n\, A$ (where A does not contain any quantifiers) and check its satisfiability.

- *Herbrand Theorem*
  - Let $\forall x_1 \cdots \forall x_n\, A$ be a universal prenex normal form in language L (A does not contain any quantifiers). The followings are equivalent:
    - $\forall x_1 \cdots \forall x_n\, A$ is unsatisfiable.
    - There exists a natural number m and H_L terms $t_{i1}, \dots, t_{in}$ ($i = 1, \dots, m$),

$$A[t_{11}/x_1, \dots, t_{1n}/x_n] \wedge \cdots \wedge A[t_{m1}/x_1, \dots, t_{mn}/x_n]$$

is unsatisfiable in any Herbrand structure $\langle H_L, J \rangle$.

Resolution Principle for Propositional Logic

- *Complementary Literal*
  - For literals L and L', L and L' are complementary if $L' = \neg L$ or $L = \neg L'$.

- *Resolvent*
  - For clauses $L_1 \vee \cdots \vee L_n$ and $L'_1 \vee \cdots \vee L'_m$, when L_i and L'_j are complementary, the clause connecting the two and removing the complementary ones is called its *resolvent*.

$$L_1 \vee \cdots \vee L_{i-1} \vee L_{i+1} \vee \cdots \vee L_n \vee L'_1 \vee \cdots \vee L'_{j-1} \vee L'_{j+1} \vee \cdots \vee L'_m$$

- Example:
  - Resolvent of $p \vee \neg q \vee r$ and $\neg p \vee q$
    - $\neg q$ and q are complementary $\Rightarrow$ $p \vee r \vee \neg p$
    - p and $\neg p$ are complementary $\Rightarrow$ $\neg q \vee r \vee q$

- Given a set of clauses, repeatedly adding a resolvent of clauses and putting it back to the set is called *resolution principle*.

Exercises: Resolvent

1. Resolvent of $p \lor \neg q$ and $\neg p \lor r$

2. Resolvent of $p \lor \neg p \lor q$ and $p \lor \neg q \lor r$

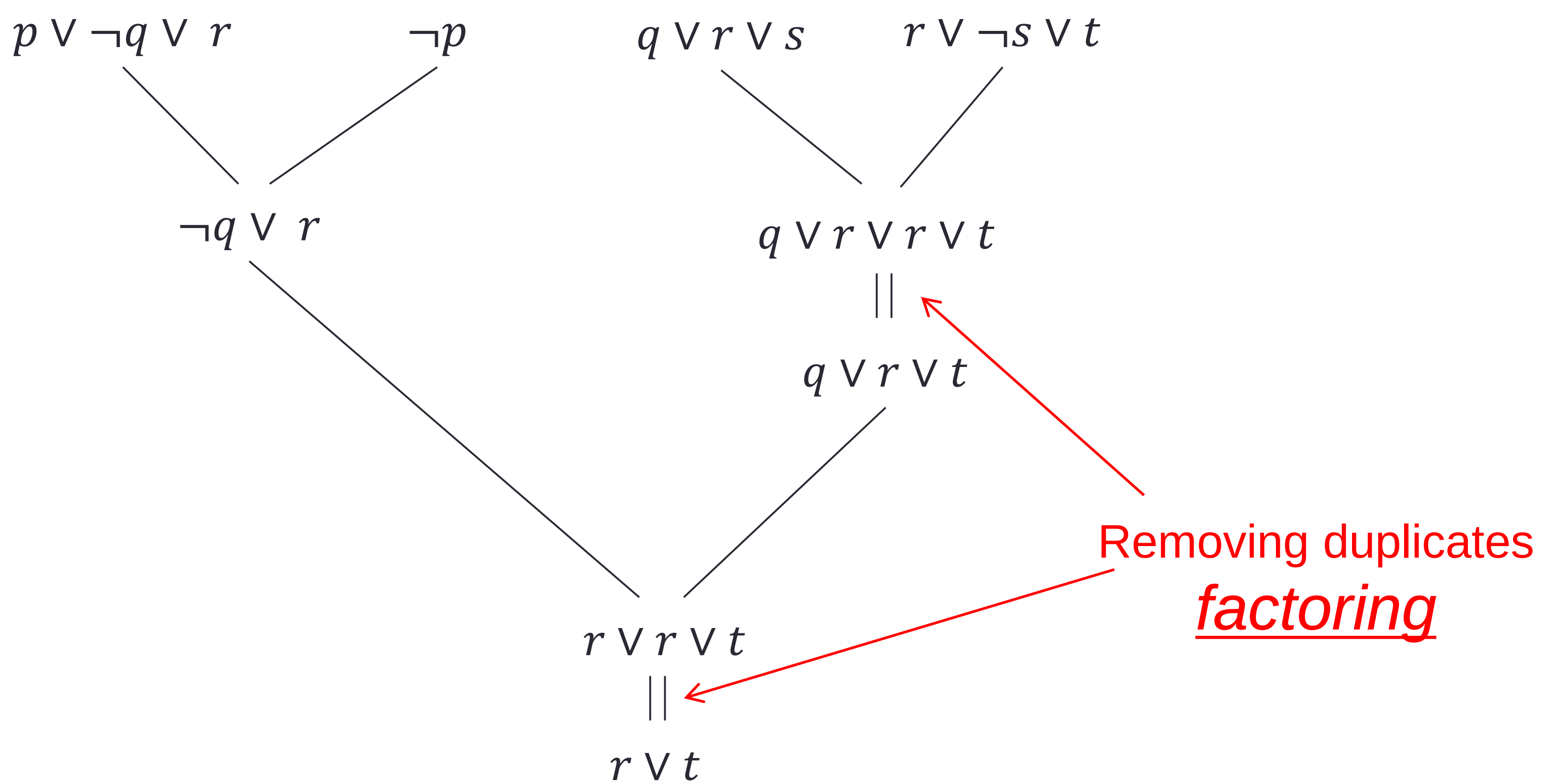
3. Resolvent of $p \lor \neg q$ and $\neg p \lor q$

4. Resolvent of $p \lor \neg q$ and $\neg p$

5. Resolvent of p and $\neg p$

Resolution Proof Tree

- $\{p \vee \neg q \vee r,\ \ \neg p,\ q \vee r \vee s,\ r \vee \neg s \vee t\}$



Exercise: Resolution Proof Tree

- $\{p \vee \neg q \vee r, \ \neg r, \ q \vee \neg r, \ \neg p \vee r, \ q \vee r\}$

Theorem about Resolution Principle

- Let C be a resolvent of C_1 and C_2.
  - If an assignment v makes both C_1 and C_2 true, it also makes C true.
    - If $v(p \vee A) = T$ and $v(\neg p \vee B) = T$, then $v(A \vee B) = T$.

- **Theorem**: If a set of clauses S is satisfiable, S with its resolvent is also satisfiable.

- *Empty Clause*
  - A clause without literals.
  - Use □ to represent the empty clause.
  - It means false or contradiction.

- **Theorem**: For a set of clauses S, if there is a resolution proof tree which contains □, S is unsatisfiable.
  - In order to show A is a tautology, convert $\neg A$ to clauses and find a resolution proof tree of □.

Predicate Logic

- $P(c)$ and $\neg P(z)$ are not complementary (where c is a constant and z is a variable).
  - Replace z with c (i.e. assigning c to z)
  - $P(c)$ and $\neg P(c)$ are complementary.

- $P(x, f(y))$ and $\neg P(z, z)$ are not complementary
  - Let $\theta = [f(y)/x]$ and $\mu = [f(y)/z]$ be two assignments.
  - $P(x, f(y))\theta = P(z, z)\mu$

- From Herbrand theorem,
  - In order to show the unsatisfiability of a set of clauses C, it is enough to show that their ground clauses are unsatisfiable.
  - Apply resolution principle to ground clauses.

Unification

- Atomic formulae $P(t_1, \ldots, t_n)$ and $Q(s_1, \ldots, s_m)$ are *unifiable* when
  - P and Q are the same predicate symbol,
  - n and m are equal, and
  - an assignment θ makes $t_1\theta = s_1\theta, \ldots, t_n\theta = s_n\theta$.

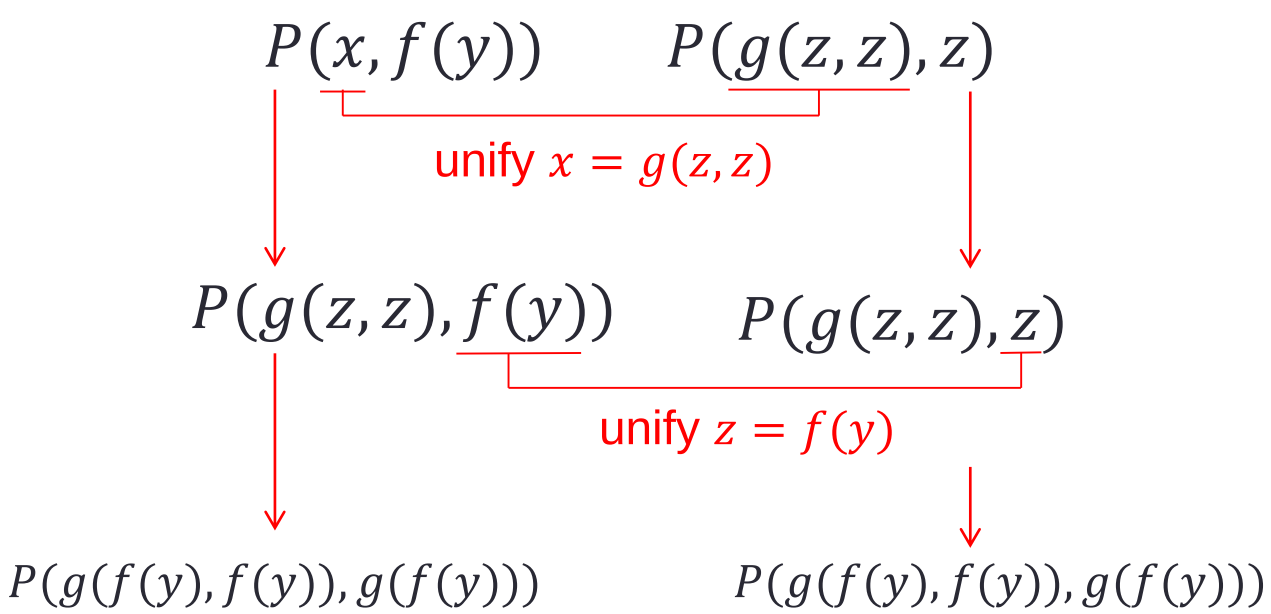
θ is called *unifier*.

- *Most General Unifier (mgu)*
  - θ is a unifier, and
  - for any unifier μ, there is a θ' and $\mu = \theta' \circ \theta$.

- Calculate mgu: compare two terms t and t' from left to right, and find unequal point.
  - If the unequal point is not variable, there is no unifier.
  - If the unequal point is a variable x and a term s,
    - if x appears inside s, there is no unifier.
    - otherwise, let $\theta = [s/x]$ and find an mgu θ' of $t\theta$ and $t'\theta$, then $\theta' \circ \theta$ is an mgu of t and t'.

Example of Unification

- Calsulate an mgu of $P(x, f(y))$ and $P(g(z,z), z)$.
  - The first unequal point is x and $g(z,z)$. Let $\theta = [g(z,z)/x]$.
  - Applying θ gives $P(g(z,z), f(y))$ and $P(g(z,z), z)$.
  - The next unequal point is $f(y)$ and z. Let $\theta' = [f(y)/z]$.
  - θ' makes both formulae $P(g(f(y), f(y)), g(f(y)))$.
  - Therefore, the mgu is $\theta' \circ \theta = [g(f(y), f(y))/x, f(y)/z]$.



Example of MGU

1. Find an mgu of $P(x)$ and $P(f(c))$ where c is a constant.

2. Find an mgu of $P(x, y)$ and $P(z, z)$.

3. Find an mgu of $P(x, c)$ と $P(z, z)$ where c is a constant.

4. Find an mgu of $P(g(x), f(y))$ and $P(z, f(g(z)))$.

Example of Resolution Principle (1)

- Prove Socrates problem.
  - $P(x)$ = "x is a man."
  - $Q(x)$ = "x is mortal."
  - Let s be an object constant for Socrates.

- $P(s) \wedge \forall x\big(P(x) \rightarrow Q(x)\big) \rightarrow Q(s)$

Example of Resolution Principle (2)

- Show the following formula is valid.
  - $\forall x\, R(x,x) \land \forall x \forall y \forall z \left(\left(R(x,y) \land R(z,y) \right) \to R(x,z) \right) \to$
    $\forall x \forall y \forall z \left(\left(R(x,y) \land R(y,z) \right) \to R(x,z) \right)$

Example of Resolution Principle (3)

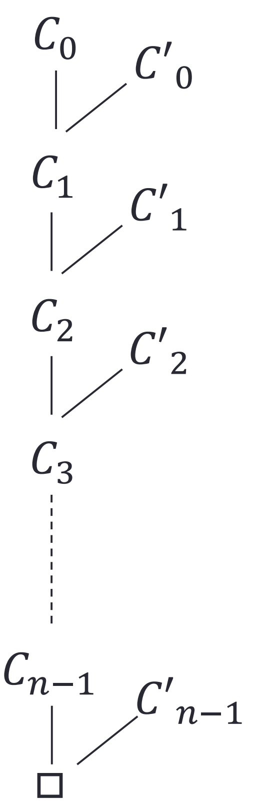
- Show the following formula is valid.

$$\forall x\, \forall y\big(R(x,y) \rightarrow R(y,x)\big)$$
$$\land\, \forall x \forall y \forall z\big(R(x,y) \land R(y,z) \rightarrow R(x,z)\big)$$
$$\land\, \forall x \exists y\, R(x,y)$$
$$\rightarrow \forall z\, R(z,z)$$

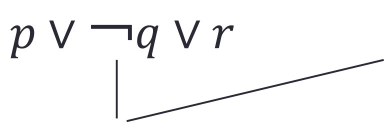
Linear Resolution

- In general, the resolution allows any order of combinations of clauses to get the empty clause.

- *Linear Resolution*
  - A set of clauses: S
  - A linear resolution: $C_0, C_1, \ldots, C_n$
  - $C_0 \in S$, $C_n = \square$
  - C_{k+1} is a resolvent of C_k and a clause of S or C_j $(j \leq k)$.



- Example: $S = \{\, p \vee \neg q \vee r,\ \ \neg r,\ q \vee \neg r,\ \ \neg p \vee r,\ \ q \vee r \,\}$
- Find a linear resolution of $p \vee \neg q \vee r$.



Logic Programming

- Logic Programming
  - Restrict to Horn clauses.
  - Starting from goal clause and using linear resolution to deduce the empty clause.

- *Horn Clause*
  - A clause $L_1 \vee \cdots \vee L_m$ where at most one literal is an atomic formula (others are negation of atomic formulae).
  - Program Clause: a clause where one literal is an atomic formula.
    - $A \vee \neg B_1 \vee \cdots \vee \neg B_n$
    - $A \leftarrow B_1, \ldots, B_n$
  - Goal Clause: a clause where all the literals are negation of atomic formulae.
    - $\neg B_1 \vee \cdots \vee \neg B_n$
    - $\leftarrow B_1, \ldots, B_n$

SWI-Prolog

- You may use SWI-Prolog on ccx01.
  - The command is `pl'.
- or, you may download SWI-Prolog to your machine.

```
% pl
Welcome to SWI-Prolog (Multi-threaded, 64 bits, Version 5.10.5)
Copyright (c) 1990-2015 University of Amsterdam, VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

1 ?- ['user'].
human(socrates).
|: mortal(X):-human(X).
|:
% user://1 compiled 0.00 sec, 1,976 bytes
true.

2 ?- mortal(socrates).
true.

3 ?- halt.
%
```

Summary

- Resolution Principle
  - resolvent of two clauses
  - a resolution proof tree with empty clause
- Unification
  - Unify two predicates by assigning terms to variables
  - mgu: most general unifier
- Logic Programming
  - Horn clause
  - Linear resolution