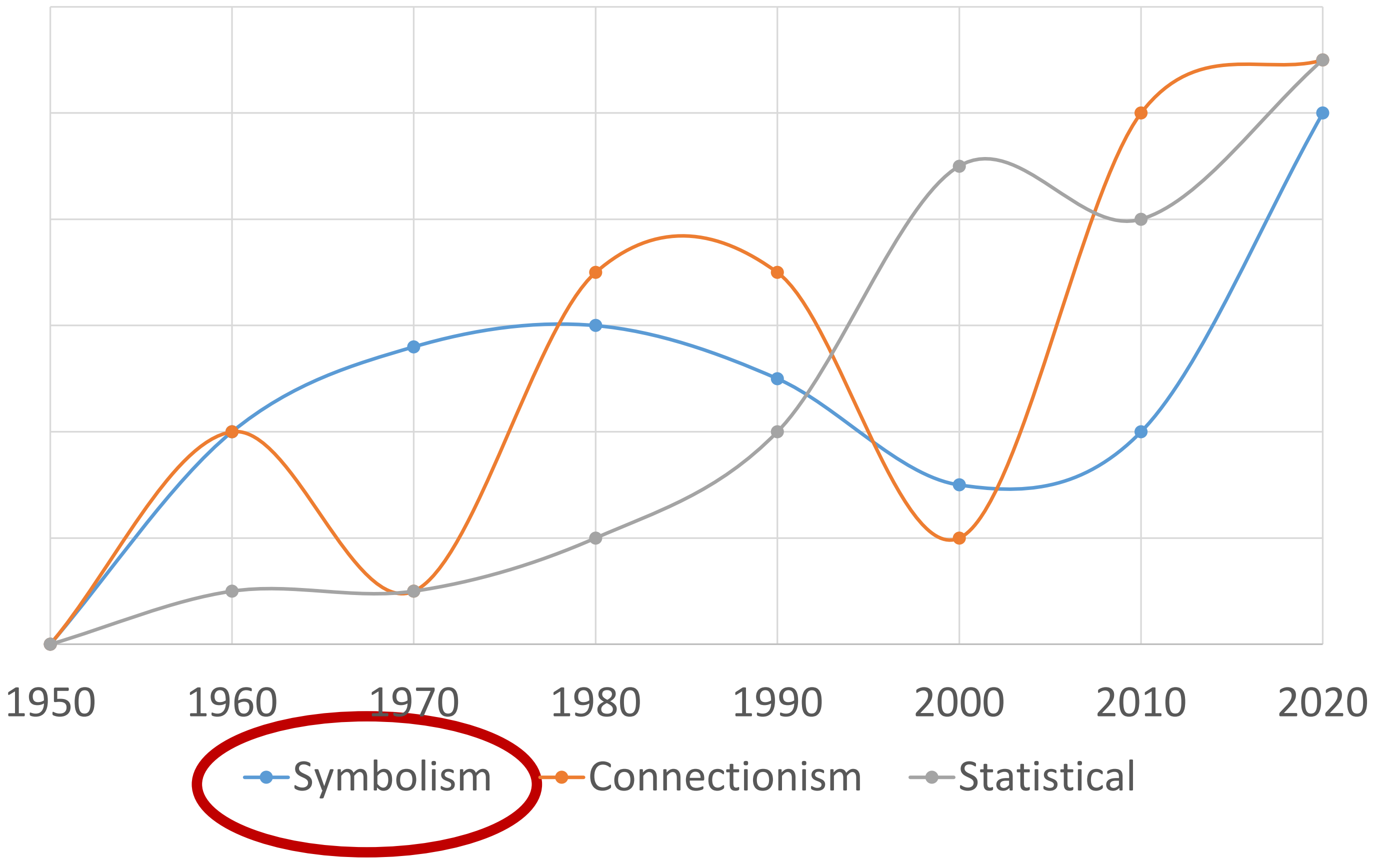
Three types of (strong) AI approaches



命题

Propositional Logic

AIMA Chapter 7

Outline

- Logic
- Propositional logic
  - Syntax
  - Semantics
  - Inference
- Horn logic
  - Inference
- An example application

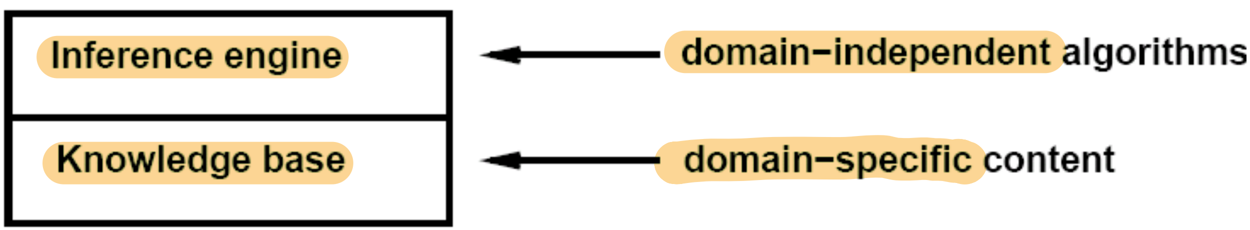
Logic-based Symbolic AI

- Logic
  - Formal language in which knowledge can be expressed
  - A means of carrying out reasoning in the language

Logic-based Symbolic AI

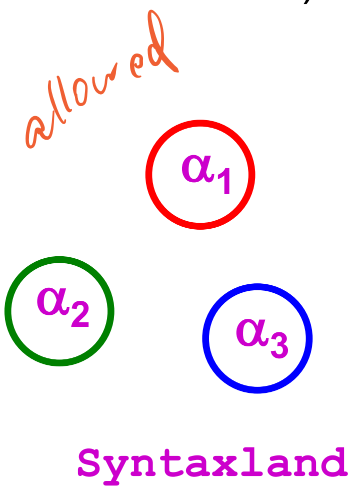
- Logic (Knowledge-Based) AI
  - Knowledge base
    - set of sentences in a formal language to represent knowledge about the "world"
  - Inference engine
    - answers any answerable question following the knowledge base

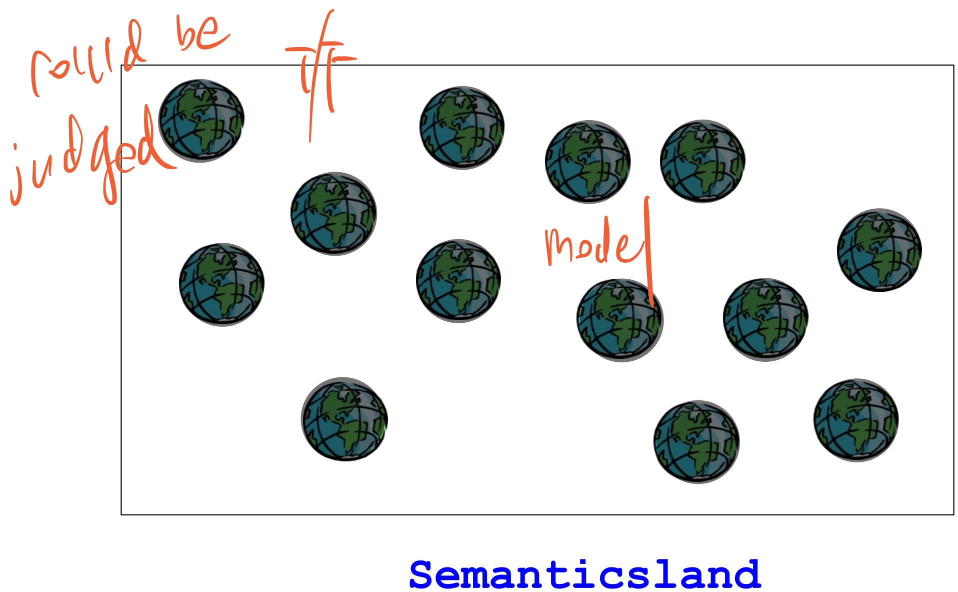
logic AI



Formal Language

- Components of a formal language in a logic
  - Syntax: What sentences are allowed?
  - Semantics:
    - Which sentences are true/false in each model (possible world)?





Formal Language

cuz no meaning

- Example: the language of arithmetic
  - Syntax
    - x+2 ≥ y is a sentence
    - x2+y > {} is not a sentence (not allowed)
  - Semantics
    - x+2 ≥ y is true in a world where x = 7, y = 1
    - x+2 ≥ y is false in a world where x = 0, y = 6

Propositional Logic

Propositional logic: Syntax

- Propositional logic is the "simplest" logic
  - The proposition symbols P1, P2, etc. are sentences
  - If S is a sentence, $\neg$S is a sentence (negation)
  - If S1 and S2 are sentences, S1 $\wedge$ S2 is a sentence (conjunction) 交
  - If S1 and S2 are sentences, S1 $\vee$ S2 is a sentence (disjunction) 并
  - If S1 and S2 are sentences, S1 $\Rightarrow$ S2 is a sentence (implication)
  - If S1 and S2 are sentences, S1 $\Leftrightarrow$ S2 is a sentence (biconditional)

$\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$ are called logic connectives or operators

Sometimes $\rightarrow$ and $\leftrightarrow$ are used proposition al logic

Examples of PL sentences

- P means “It is hot.”
- Q means “It is humid.”
- R means “It is raining.”
- $(P \wedge Q) \Rightarrow R$
  - “If it is hot and humid, then it is raining”
- $Q \Rightarrow P$
  - “If it is humid, then it is hot”

Propositional logic: Semantics

- Each model specifies true/false for each proposition symbol
  - E.g. P_1 false, P_2 true, P_3 false

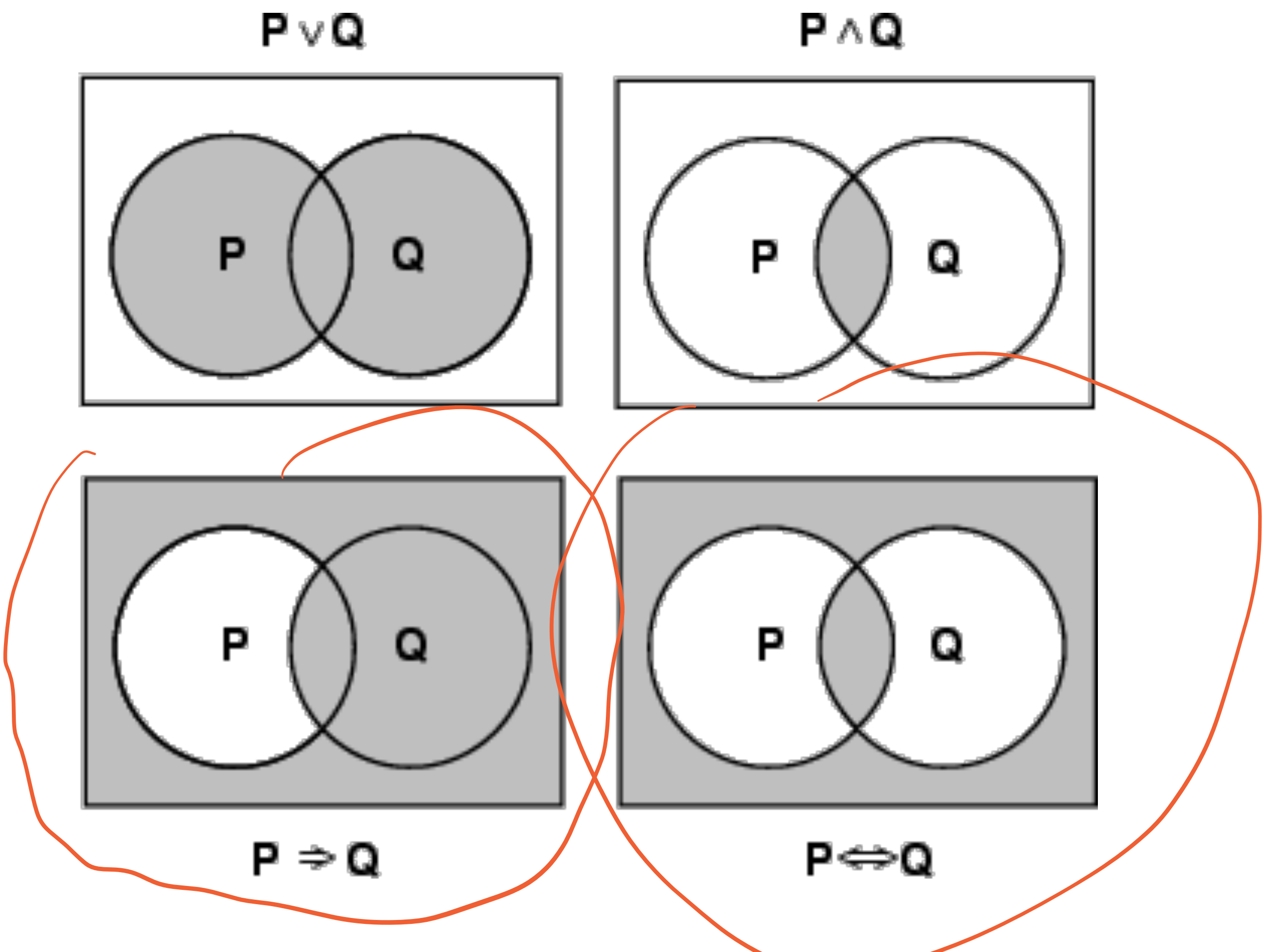
	P_1	P_2	P_3
E.g.	false	true	false

- Rules for evaluating truth with respect to a model m:
  - $\neg$S is true iff S is false
  - S1 $\wedge$ S2 is true iff S1 is true and S2 is true
  - S1 $\vee$ S2 is true iff S1 is true or S2 is true
  - S1 $\Rightarrow$ S2 is true iff S1 is false or S2 is true
  - S1 $\Leftrightarrow$ S2 is true iff S1$\Rightarrow$S2 is true and S2$\Rightarrow$S1 is true

Truth tables for connectives

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
$false$	$false$	$true$	$false$	$false$	$true$	$true$
$false$	$true$	$true$	$false$	$true$	$true$	$false$
$true$	$false$	$false$	$false$	$true$	$false$	$false$
$true$	$true$	$false$	$true$	$true$	$true$	$true$

Venn Diagrams



Material Implication

- S1 ⇒ S2 is true iff S1 is false or S2 is true
- Given the following propositions, is “S1 ⇒ S2” true?
  - S1 means “the moon is made of green cheese”
  - S2 means “the world is coming to an end”

no natural language

- Material implication does not capture the meaning of “if… then”.
- See “Paradoxes of material implication” in Wikipedia

φ

$(\neg B) \wedge (A)$ X ,

$(\neg \beta) \wedge (\neg c)$ X .

Logical equivalence

- Two sentences are logically equivalent iff true in same models

$$
\begin{aligned}
(\alpha \wedge \beta) &\equiv (\beta \wedge \alpha) && \text{commutativity of } \wedge \\
(\alpha \vee \beta) &\equiv (\beta \vee \alpha) && \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) &\equiv (\alpha \wedge (\beta \wedge \gamma)) && \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) &\equiv (\alpha \vee (\beta \vee \gamma)) && \text{associativity of } \vee \\
\neg(\neg \alpha) &\equiv \alpha && \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) &\equiv (\neg \beta \Rightarrow \neg \alpha) && \text{contraposition} \\
(\alpha \Rightarrow \beta) &\equiv (\neg \alpha \vee \beta) && \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) && \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) &\equiv (\neg \alpha \vee \neg \beta) && \text{de Morgan} \\
\neg(\alpha \vee \beta) &\equiv (\neg \alpha \wedge \neg \beta) && \text{de Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) &\equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) && \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) &\equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) && \text{distributivity of } \vee \text{ over } \wedge
\end{aligned}
$$

切记!

$$(A \vee B \vee C) \vee (A \wedge B \wedge \neg C)$$

$$= (A \vee B \vee C) \vee (A \wedge (B \wedge \neg C))$$

Validity and satisfiability

$A \wedge (A \to B)$
$= A \wedge (\neg A \vee B)$
$= (A \wedge \neg A) \vee (A \wedge B)$
$= A \wedge B.$

- A sentence is valid if it is true in all models
  - e.g., $A \vee \neg A$, $A \Rightarrow A$, $(A \wedge (A \Rightarrow B)) \Rightarrow B$

T

F

$A \wedge B$ true

A true

B true

- A sentence is satisfiable if it is true in some model
  - e.g., $A \vee B$, C
- A sentence is unsatisfiable if it is true in no models
  - e.g., $A \wedge \neg A$

- Obviously, S is valid iff. $\neg S$ is unsatisfiable

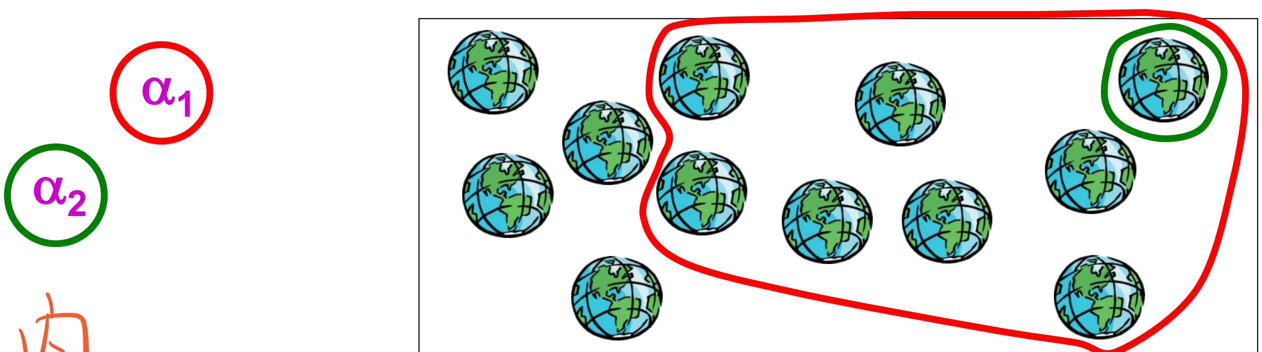
Inference: entailment

使必要

- Entailment: α |= β ("α entails β" or "β follows from α") means in every world where α is true, β is also true
  - i.e., the α-worlds are a subset of the β-worlds [models(α) ⊆ models(β)]
- In the example, α2 |= α1

world = model

Q why



set 内

同性： $A \cap B$ true, A true

"True $\models$ False" is indeed false: every model makes "true" true, but no model makes "false" true, so *every* model provides a counterexample.

However, since no model makes "false" true, "False $\models$ True" is actually *true*! It's an instance of *vacuous implication*: think of it as being true for the same reason the statement

$$\text{"If } 0 = 1\text{, then I'm the president"}$$

is true.

(a) α is valid if and only if $True \models \alpha$.

Forward direction: If $True \models \alpha$, then α is valid.

By definition, $True \models \alpha$ means that α is true in all worlds where $True$ is True; this is all worlds. Thus α is true in all worlds, which is exactly the defintion of validity. So α is in this case valid.

Backward direction: If α is valid, then $True \models \alpha$.

By definition, if α is valid then it is true in all worlds. In this case *anything* entails α, so clearly $True$ entails α.

(b) For any α, $False \models \alpha$.

Recall the definition of entailment: $p \models q$ means that in all worlds in which p is true, q is true as well. So, $False \models \alpha$ means that in all worlds in which $False$ is true, α is true. But there are no worlds in which $False$ is true! Clearly if there are no worlds in a set, then that set satisfies the condition that α be true in all worlds in that set.

(c) $\alpha \models \beta$ if and only if the sentence $(\alpha \Rightarrow \beta)$ is valid.

Forward direction: If $\alpha \models \beta$, then the sentence $(\alpha \Rightarrow \beta)$ is valid.

By definition, $\alpha \models \beta$ means that β is true in all worlds in which α is true. Thus, in all worlds in which α is true $\alpha \Rightarrow \beta$ holds because both α and β will be true. We must also consider worlds in which α is false; in these worlds, $\alpha \Rightarrow \beta$ also holds by definition of the falsehood of α.

Backward direction: If the sentence $(\alpha \Rightarrow \beta)$ is valid, then $\alpha \models \beta$.

If the sentence $(\alpha \Rightarrow \beta)$ is valid, then it is true in all worlds. Thus, for every world, it must be the case that either both α and β are true, *or* α is false. This is enough to tell us that in every world in which α is true, β is also true, which is the definition of entailment.

Inference: proof

- A proof (α |- β) is a demonstration of entailment from α to β
  - Method 1: model checking 枚举
    - Truth table enumeration to check if models(α) $\subseteq$ models(β)
    - Time complexity always exponential in n ☹

P1	**P2**	**…**	**Pn**	**α**	**β**
F	F	…	F	F	T
F	F	…	T	T	T
		……			
T	T	…	F	T	T
T	T	…	T	F	F

CNF

Examples and non-examples [edit]

All of the following formulas in the variables A, B, C, D, E, and F are [illegible]onjunctive normal form:

- $(A \lor \neg B \lor \neg C) \land (\neg D \lor E \lor F)$
- $(A \lor B) \land (C)$
- $(A \lor B)$
- (A)

For clarity, the disjunctive clauses are written inside parentheses above. In disjunctive normal form with parenthesized conjunctive clauses, the last case is the same, but the next to last is $(A) \lor (B)$. The constants *true* and *false* are denoted by the empty conjunct and one clause consisting of the empty disjunct, but are normally written explicitly.[1]

The following formulas are **not** in conjunctive normal form:

- $\neg(B \lor C)$, since an OR is nested within a NOT
- $(A \land B) \lor C$
- $A \land (B \lor (D \land E))$, since an AND is nested within an OR

Every formula can be equivalently written as a formula in conjunctive normal form. The three non-examples in CNF are:

- $(\neg B) \land (\neg C)$
- $(A \lor C) \land (B \lor C)$
- $(A) \land (B \lor D) \land (B \lor E)$.

都可

Conversion into CNF [edit]

[2]Every propositional formula can be converted into an equivalent formula that is in CNF. This transformation is based on rules about logical equivalences: double negation elimination, De Morgan's laws, and the distributive law.

Since all propositional formulas can be converted into an equivalent formula in conjunctive normal form, proofs are often based on the assumption that all formulae are CNF. However, in some cases this conversion to CNF can lead to an exponential explosion of the formula. For example, translating the following non-CNF formula into CNF produces a formula with 2^n clauses:

$$(X_1 \land Y_1) \lor (X_2 \land Y_2) \lor \cdots \lor (X_n \land Y_n).$$

In particular, the generated formula is:

$$(X_1 \lor X_2 \lor \cdots \lor X_n) \land (Y_1 \lor X_2 \lor \cdots \lor X_n) \land (X_1 \lor Y_2 \lor \cdots \lor X_n) \land (Y_1 \lor Y_2 \lor \cdots \lor X_n) \land \cdots \land (Y_1 \lor Y_2 \lor \cdots \lor Y_n).$$

This formula contains 2^n clauses; each clause contains either X_i or Y_i for each i.

There exist transformations into CNF that avoid an exponential increase in size by preserving satisfiability rather than equivalence.[3][4] These transformations are guaranteed to only linearly increase the size of the formula, but introduce new variables. For example, the above formula can be transformed into CNF by adding variables $Z_1, \ldots, Z_n$ as follows:

$$(Z_1 \lor \cdots \lor Z_n) \land (\neg Z_1 \lor X_1) \land (\neg Z_1 \lor Y_1) \land \cdots \land (\neg Z_n \lor X_n) \land (\neg Z_n \lor Y_n).$$

An interpretation satisfies this formula only if at least one of the new variables is true. If this variable is Z_i, then both X_i and Y_i are true as well. This means that every model that satisfies this formula also satisfies the original one. On the other hand, only some of the models of the original formula satisfy this one: since the Z_i are not mentioned in the original formula, their values are irrelevant to satisfaction of it, which is not the case in the last formula. This means that the original formula and the result of the translation are equisatisfiable but not equivalent.

An alternative translation, the Tseitin transformation, includes also the clauses $Z_i \lor \neg X_i \lor \neg Y_i$. With these clauses, the formula implies $Z_i \equiv X_i \land Y_i$; this formula is often regarded to "define" Z_i to be a name for $X_i \land Y_i$.

推断
Inference: proof

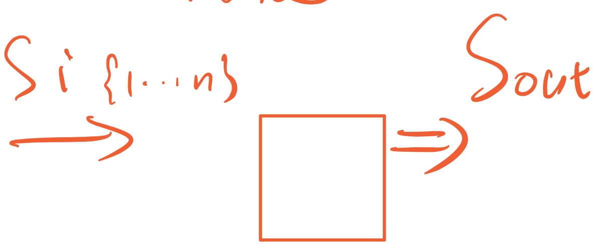
- A proof (α |- β) is a demonstration of entailment from α to β
  - Method 2: application of inference rules
    - Search for a finite sequence of sentences each of which is an axiom or follows from the preceding sentences by a rule of inference

自明之理

先前的、

    - Axiom: a sentence known to be true
    - Rule of inference: a function that takes one or more sentences (premises) and returns a sentence (conclusion)

假设



Inference: soundness & completeness

- Sound inference 必要.
  - everything that can be proved is in fact entailed
- Complete inference
  - everything that is entailed can be proved

- Method 1 (enumeration) is obviously sound and complete
- For method 2 (applying inference rules), it is much less obvious
  - Example: arithmetic is found to be not complete! (Gödel's theorem, 1931)

Quiz

- What's the connection between complete inference algorithms and complete search algorithms?
- Answer 1: they both have the words "complete…algorithm"
- Answer 2: Formulate inference α |- β as a search problem
  - Initial state: KB contains α
  - Actions: apply any inference rule that matches KB, add conclusion
  - Goal test: KB contains β

Q

KB?

Hence any complete search algorithm can be used to produce a complete inference algorithm

guarantee to find a solution if exist

Resolution: an inference rule in PL

- **Conjunctive Normal Form** (CNF) 子句
  - conjunction of <u>disjunctions of literals</u> (clauses)
  - Ex
    - $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$
    - $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

≥ 2 entry

Conversion to CNF

$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

step

1. Eliminate $\Leftrightarrow$, replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.

   $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$

2. Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$.

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$

3. Move $\neg$ inwards using de Morgan's rules and double-negation:

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$

4. Apply distributivity law ($\wedge$ over $\vee$) and flatten:

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

Resolution: an inference rule in PL

该有[illegible]也成立 C

- Resolution inference rule (for CNF):

  Suppose l_i is $\neg m_j$

entail

$$\frac{l_1 \vee \ldots \vee l_k, \; (\vee C) \qquad m_1 \vee \ldots \vee m_n \; (\vee C)}{l_1 \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k \vee m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n}$$

  Examples:

$$\frac{P_{1,3} \vee P_{2,2}, \qquad P_{2,3} \vee \neg P_{2,2}}{P_{1,3} \vee P_{2,3}} \qquad\qquad \frac{P_1, \neg P_1}{\{\}}$$

- Resolution is sound and complete for propositional logic

The **resolution rule** in propositional logic is a single valid inference rule that produces a new clause implied by two clauses containing complementary literals. A literal is a propositional variable or the negation of a propositional variable. Two literals are said to be complements if one is the negation of the other (in the following, $\neg c$ is taken to be the complement to c). The resulting clause contains all the literals that do not have complements. Formally:

$$\frac{a_1 \vee a_2 \vee \cdots \vee c, \quad b_1 \vee b_2 \vee \cdots \vee \neg c}{a_1 \vee a_2 \vee \cdots \vee b_1 \vee b_2 \vee \cdots}$$

where

all a_i, b_i, and c are literals,

the dividing line stands for "entails".

$$\frac{p \to q, \ p}{q} \qquad \frac{\neg p \vee q, \ p}{q}$$

The above may also be written as:

$$\frac{(\neg a_1 \wedge \neg a_2 \wedge \cdots) \to c, \quad c \to (b_1 \vee b_2 \vee \cdots)}{(\neg a_1 \wedge \neg a_2 \wedge \cdots) \to (b_1 \vee b_2 \vee \cdots)}$$

Or schematically as:

$$\frac{\Gamma_1 \cup \{\ell\} \quad \Gamma_2 \cup \{\bar{\ell}\}}{\Gamma_1 \cup \Gamma_2} |\ell|$$

key

Example OF Propositional Resolution

Consider the following Knowledge Base:

1. The humidity is high or the sky is cloudy. $P \vee Q$
2. If the sky is cloudy, then it will rain. $Q \to R$
3. If the humidity is high, then it is hot. $P \to S$
4. It is not hot. $\neg S$

KB true = 全取 ∧ (CNF)

Goal: It will rain.

Use propositional logic and apply resolution method to prove that the goal is derivable from the given knowledge base.

Solution: Let's construct propositions of the given sentences one by one:

1. Let, P: Humidity is high.

   Q: Sky is cloudy.

It will be represented as **P V Q.**

2) Q: Sky is cloudy. ...**from(1)**

Let, R: It will rain.

It will be represented as b**Q→R.**

3) P: Humidity is high. ...**from(1)**

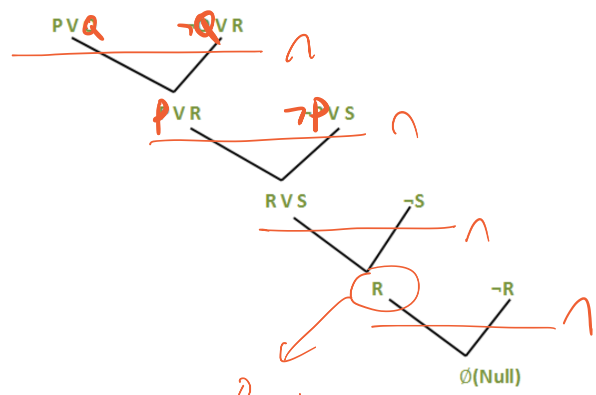
KB | α

Let, S: It is hot.

It will be represented as **P** ? **S.**

4) ¬S: It is not hot.

Applying resolution method:



In (2), Q ? R will be converted as (¬Q V R)

In (3), P ? S will be converted as (¬P V S)

Negation of Goal (¬R): It will ~~not~~ rain. it will rain

Rain . ok here .

Finally, apply the rule as shown below:

After applying Proof by Refutation (Contradiction) on the goal, the problem is solved, and it has terminated with a **Null clause (Ø).** Hence, the goal is achieved. ~~Thus, It is not raining.~~

证明了 $KB \models \alpha$

target: prove $KB \models \alpha$

$\alpha_1 \wedge \alpha_2 \wedge \cdots \wedge \alpha_k$ is true.

KB: knowledge base.

of sentences $\alpha_1 \cdots \alpha_k$.

Collection of

we know are true.

Resolution algorithm

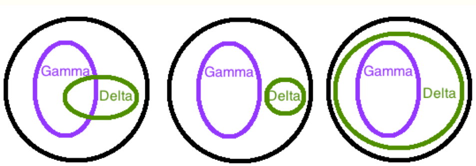
KB true, α true.

- The best way to prove *KB* |= α ?
  - Proof by contradiction, i.e., show *KB*∧¬α is unsatisfiable

1. Convert *KB*∧¬α to CNF
2. Repeatedly apply the resolution rule to add new clauses, until one of the two things happens
   a) Two clauses resolve to yield the empty clause, in which case *KB* entails α
   b) There is no new clause that can be added, in which case *KB* does not entail α

Knowledge Base

A knowledge base is a collection of sentences $\alpha_1, \alpha_2, \ldots, \alpha_k$ that we know are true, i.e. $\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_k$. The sentences in the knowledge base allows us to compactly specify the allowable states of the world. By adding new sentences, the number of possible worlds consistent with our knowledge base could decrease (if we gain new information), go to zero (if the new sentence is inconsistent with the existing knowledge base), or remain the same (if the new sentence is entailed by existing sentences).



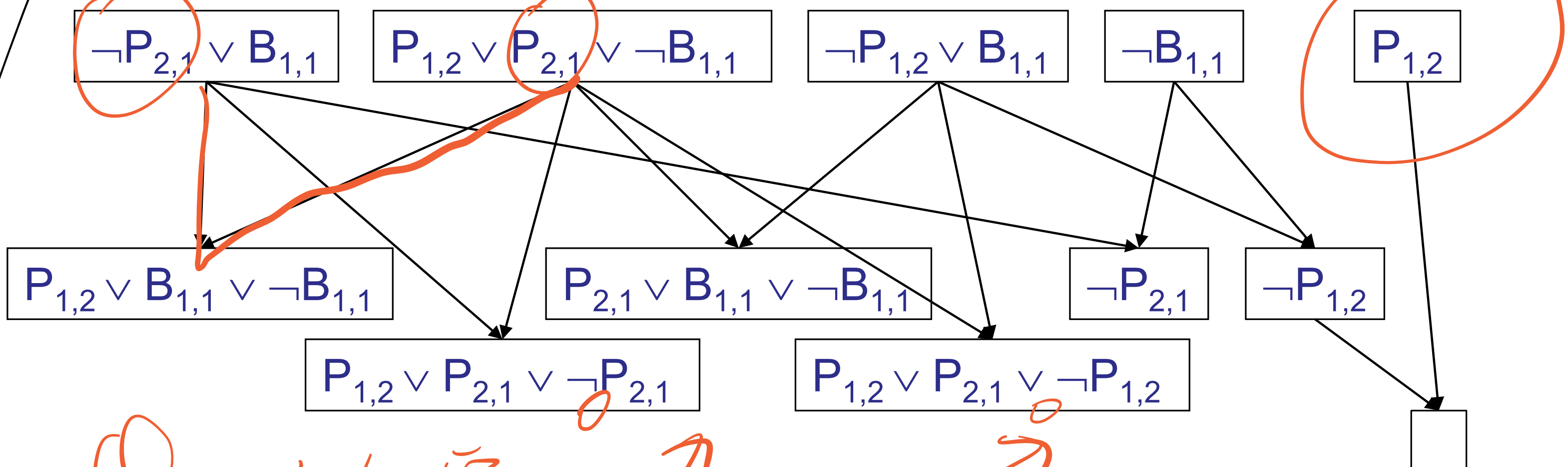
Resolution example

negative target

$KB = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1}$

$\alpha = \neg P_{1,2}$

CNF



Q：没体现吧

$(B \Rightarrow P_{1,2} \vee P_{2,1}) \wedge (P_{1,2} \vee P_{2,1} \Rightarrow B)$

$= (\neg B \cup P_{1,2} \cup P_{2,1}) \cap (\neg(P_{1,2} \vee P_{2,1}) \cup B)$

De

$= (\quad) \cap ((\neg P_{1,2} \wedge \neg P_{2,1}) \cup B)$

括号!

$= (\quad) \cap (\neg P_{1,2} \cup B) \wedge (\neg P_{2,1} \cup B)$