CS 480

Introduction to Artificial Intelligence

March 1, 2022

Announcements / Reminders

- **Written Assignment #02:**
  - **due: March 3rd, 11:00 PM CST**

- **Programming Assignment #01:**
  - **due: March 6th, 11:00 PM CST**

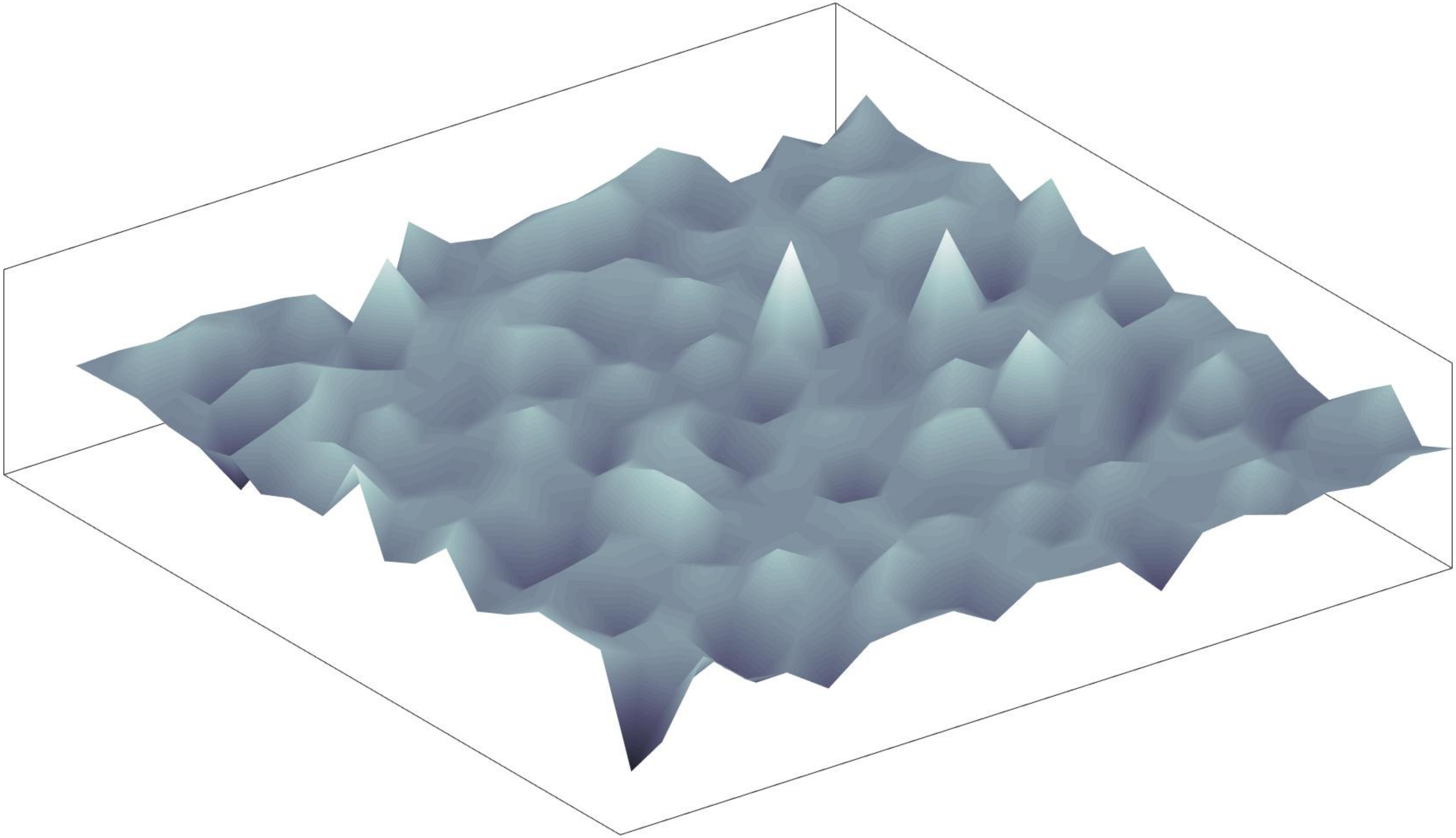
- **Grading TA assignment:**

  https://docs.google.com/spreadsheets/d/1avK4P4MDjKZQceG82mSZd0wkYEDH07_DpQqYJHDQctw/edit?usp=sharing

Plan for Today

- **Propositional logic and inference**

Hill Climbing Assignment: Comments



Implication | Equivalence | Entailment

IMPLICATION

A sentence is satisfiable if it is true for AT LEAST ONE interpretation.

In plain English:
true implies true
true DOES NOT imply false
false implies true
false implies false

Notation:

$KB \Rightarrow Q$

KB	Q	$KB \Rightarrow Q$
true	true	true
true	false	false
false	true	true
false	false	true

EQUIVALENCE

A sentence is (logically) valid if it is true for ALL interpretations. Also called a tautology.

In plain English:
true equivalent to true
true NOT equivalent to false
false NOT equivalent to true
false equivalent to false

Notation:

$KB \Leftrightarrow Q$

KB	Q	$KB \Leftrightarrow Q$
true	true	true
true	false	false
false	true	false
false	false	true

ENTAILMENT

A sentence is unsatisfiable if it is NOT true for ANY interpretation. Also called a contradiction.

In plain English:
true follows from true
false DOES NOT follow from true
true DOES NOT follow from false
false DOES NOT follow from false

Notation:

$KB \vDash Q$

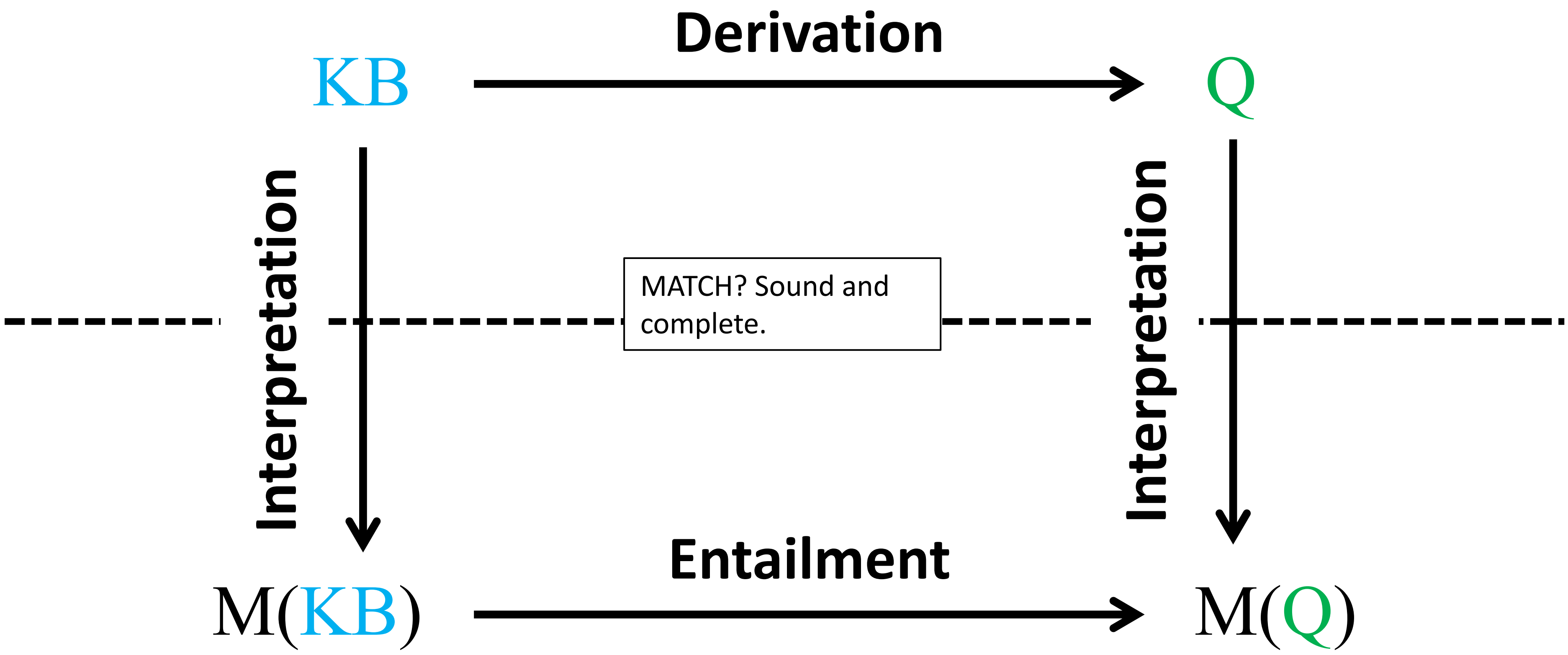
KB	Q	$KB \vDash Q$
true	true	true
true	false	false
false	true	false
false	false	false

Propositional Logic and KB-Agents

Propositional Logic: Syntax	Propositional Logic: Semantics	Propositional Logic: Inference and Proof Systems	KB-Agents: Inference algorithms

Proving Entailment: Two Levels

Syntax level



Semantic level

Inference

Bottom line:

An inference system has to be sound and complete.

Resolution rule is. Couple it with a complete search algorithm and an inference system is in place.

Inference Rules: Resolution

Rules of Inference:

Modus Ponens	Modus Tollens	Hypothetical Syllogism (Transitivity)	Conjunction
$P \Rightarrow Q$ P	$P \Rightarrow Q$ $\neg Q$	$P \Rightarrow Q$ $Q \Rightarrow R$	P Q
$\therefore Q$	$\therefore \neg P$	$\therefore P \Rightarrow R$	$\therefore P \wedge Q$

Addition	Simplification	Disjunctive Syllogism	Resolution
P	$P \wedge Q$	$P \vee Q$ $\neg P$	$P \vee Q$ $\neg P \vee R$
$\therefore P \vee Q$	$\therefore P$	$\therefore Q$	$\therefore Q \vee R$

Tautological forms:

Modus Ponens: $((P \Rightarrow Q) \wedge P) \Rightarrow Q$ | **Modus Tollens:** $((P \Rightarrow Q) \wedge \neg Q) \Rightarrow \neg P$

Hypothetical Syllogism: $((P \Rightarrow Q) \wedge (Q \Rightarrow R)) \Rightarrow (P \Rightarrow R)$

Disjunctive Syllogism: $((P \vee Q) \wedge \neg P) \Rightarrow \neg Q$

Addition: $P \Rightarrow P \vee Q$ | **Simplification:** $(P \wedge Q) \Rightarrow P$

Conjunction: $(P) \wedge (Q) \Rightarrow (P \wedge Q)$ | **Resolution:** $((P \vee Q) \wedge (\neg P \vee R)) \Rightarrow (Q \vee R)$

Proof by Resolution

Recall that we can show that KB entails sentence Q (or Q follows from KB):

$$KB \models Q$$

by proving that:

$$(KB \land \neg Q) \Leftrightarrow \bot$$

(show that $KB \land \neg Q$ is a contradiction / empty clause)

Resolution: Two Forms of Notation

Resolution

$P \vee Q$

$\neg P \vee R$

$\therefore\ Q \vee R$

Resolution (textbook)

$$\frac{(P \vee Q),\ (\neg P \vee R)}{(Q \vee R)}$$

Resolution: Two Forms of Notation

Resolution

$P \vee Q$

$\neg P \vee R$

$\therefore\ Q \vee R$

Resolution (textbook)

$$\frac{(P \vee Q),\ (\neg P \vee R)}{(Q \vee R)}$$

derived clause (resolvent)

The Empty Clause: $(p \wedge \neg p) \Leftrightarrow \bot$

Symbol	Name	Alternative symbols*	Should be read
$\neg$	Negation	$\sim$, !	not
$\wedge$	(Logical) conjunction	$\bullet$, &	and
$\vee$	(Logical) disjunction	$+$, $\|\|$	or
$\Rightarrow$	(Material) implication	$\rightarrow$, $\supset$	implies
$\Leftrightarrow$	(Material) equivalence	$\leftrightarrow$, $\equiv$, iff	if and only if
$\top$	Tautology	T, 1, $\blacksquare$	truth
$\bot$	Contradiction	F, 0, $\square$	falsum empty clause
$\therefore$	Therefore		therefore

* you can encounter it elsewhere in literature

Conjunctive Normal Form (CNF)

A sentence is in conjunctive normal form (CNF) if and only if consists of conjunction:

$$K_1 \wedge K_2 \wedge \ldots \wedge K_m$$

of clauses. A clause Ki consists of a disjunction

$$(l_{i1} \vee l_{i2} \vee \ldots \vee l_{ini})$$

of literals. Finally, a literal is propositional variable (positive literal) or a negated propositional variable (negative literal).

Conjunctive Normal Form (CNF)

Example:

Convert $m \Leftrightarrow (n \vee o)$ **into CNF**:

by Equivalence law $(p \Rightarrow q) \wedge (q \Rightarrow p) \;\Leftrightarrow\; (p \Leftrightarrow q)$

$$(m \Rightarrow (n \vee o)) \wedge ((n \vee o) \Rightarrow m)$$

by Implication law $\neg\, p \vee q \Leftrightarrow p \Rightarrow q$

$$(\neg\, m \vee (n \vee o)) \wedge (\neg\, (n \vee o) \vee m)$$

we can remove parentheses

$$(\neg\, m \vee n \vee o) \wedge (\neg\, (n \vee o) \vee m)$$

by De Morgan's law $\neg\, (p \wedge q) \Leftrightarrow \neg\, q \vee \neg\, p$

$$(\neg\, m \vee n \vee o) \wedge ((\neg\, n \wedge \neg\, o) \vee m)$$

by Distributive law $p \vee (q \wedge r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$

$$(\neg\, m \vee n \vee o) \wedge (\neg\, n \vee m) \wedge (\neg\, o \vee m)$$

Conjunctive Normal Form (CNF)

Example:

Sentence $m \Leftrightarrow (n \vee o)$ **converted into CNF:**

literals

$$(\neg m \vee n \vee o) \wedge (\neg n \vee m) \wedge (\neg o \vee m)$$

clauses

CNF Grammar

$$
\begin{aligned}
CNFSentence &\rightarrow Clause_1 \wedge \cdots \wedge Clause_n \\
Clause &\rightarrow Literal_1 \vee \cdots \vee Literal_m \\
Fact &\rightarrow Symbol \\
Literal &\rightarrow Symbol \mid \neg Symbol \\
Symbol &\rightarrow P \mid Q \mid R \mid \ldots \\
HornClauseForm &\rightarrow DefiniteClauseForm \mid GoalClauseForm \\
DefiniteClauseForm &\rightarrow Fact \mid (Symbol_1 \wedge \cdots \wedge Symbol_l) \Rightarrow Symbol \\
GoalClauseForm &\rightarrow (Symbol_1 \wedge \cdots \wedge Symbol_l) \Rightarrow False
\end{aligned}
$$

*** I will:**

- **be using** true **and** false **instead of** True **and** False
- **use lowercase** p, q **for atomic and uppercase** P, Q **for complex**

General Resolution Rule

General resolution rule allows clauses with arbitrary number of literals

$$\frac{(a_1 \vee \ldots \vee a_m \vee b), (\neg\, b \vee c_1 \vee \ldots \vee c_n)}{(a_1 \vee \ldots \vee a_m \vee c_1 \vee \ldots \vee c_n)}$$

where: a_i, b, $\neg\, b$, c_j **are literals.**

Unit Resolution

General resolution rule allows clauses with arbitrary number of literals

initial clauses

$$\frac{(a_1 \vee \ldots \vee a_m \vee \mathbf{b}),\ (\neg\, \mathbf{b} \vee c_1 \vee \ldots \vee c_n)}{(a_1 \vee \ldots \vee a_m \vee c_1 \vee \ldots \vee c_n)}$$

new clause

Literals $\mathbf{b}$ and $\neg\, \mathbf{b}$ are complimentary. The resolution rule deletes a pair of complimentary literals from two clauses and combines the rest.

Unit Resolution

General resolution rule allows clauses with arbitrary number of literals

initial clauses

$$\frac{(a_1 \vee \ldots \vee a_m \vee \mathbf{b}),\ (\neg\, \mathbf{b} \vee c_1 \vee \ldots \vee c_n)}{(a_1 \vee \ldots \vee a_m \vee c_1 \vee \ldots \vee c_n)}$$

derived clause (resolvent)

Literals $\mathbf{b}$ and $\neg\,\mathbf{b}$ are complimentary. The clause ($\mathbf{b} \wedge \neg\,\mathbf{b}$) is a contradiction (an <u>empty clause</u>).

Unit Resolution

General resolution rule allows clauses with arbitrary number of literals

$$\frac{(a_1 \vee \ldots \vee a_m \vee \mathbf{b}),\ (\neg\ \mathbf{b} \vee c_1 \vee \ldots \vee c_n)}{(a_1 \vee \ldots \vee a_m \vee c_1 \vee \ldots \vee c_n)}$$

Literals $\mathbf{b}$ and $\neg\ \mathbf{b}$ are complimentary. The resolution rule deletes a pair of complimentary literals from two clauses and combines the rest.

Factorization

Ocassionally, unit resolution will produce a new clause with the the following clause ($\mathbf{d} \vee \mathbf{d}$):

$$\frac{(a_1 \vee \ldots \vee a_m \vee \mathbf{d} \vee b), (\neg\, b \vee c_1 \vee \ldots \vee c_n \vee \mathbf{d})}{(a_1 \vee \ldots \vee a_m \vee c_1 \vee \ldots \vee c_n \vee \mathbf{d} \vee \mathbf{d})}$$

Disjunction of multiple copies of literals ($\mathbf{d} \vee \mathbf{d}$) can be replaced by a single literal $\mathbf{d}$. This is called factorization.

Resolution and Factorization

In this example resolution along with factorization will generate a new clause:

$$\frac{(a_1 \vee \ldots \vee a_m \vee \mathbf{d} \vee \mathbf{b}),\ (\neg\, \mathbf{b} \vee c_1 \vee \ldots \vee c_n \vee \mathbf{d})}{(a_1 \vee \ldots \vee a_m \vee c_1 \vee \ldots \vee c_n \vee \mathbf{d})}$$

Clause is ($\mathbf{d} \vee \mathbf{d}$) is replaced by a single literal $\mathbf{d}$. This is called factorization. Contradiction ($\mathbf{b} \wedge \neg\, \mathbf{b}$) becomes an "empty clause" and is removed.

Proof by Resolution: Example

Consider the following problem:

Three girls practice high jump for their physical education exam. The bar is set to 1.20 meters. "I bet", says the first girl to the second, "that I will make it over it, and only if, you don't".

If the second girl said the same to the third, who in turn said the same to the first, would it be possible for all three to win their bets?

Proof by Resolution: Example

Formalization step (English to Propositional Logic):

Propositional variables:

a: **the first girl's jump succeeds**

b: **the second girl's jump succeeds**

c: **the third girl's jump succeeds**

Sentences (bets):

First girl's bet: $(a \Leftrightarrow \neg b)$

Second girl's bet: $(b \Leftrightarrow \neg c)$

Third girl's bet: $(c \Leftrightarrow \neg a)$

Proof by Resolution: Example

Claim step (what are we trying to prove):

Claim: the three CANNOT all win their bets

$$C \equiv \neg((a \Leftrightarrow \neg\, b) \wedge (b \Leftrightarrow \neg\, c) \wedge (c \Leftrightarrow \neg\, a))$$

First girl's bet: $(a \Leftrightarrow \neg\, b)$

Second girl's bet: $(b \Leftrightarrow \neg\, c)$

Third girl's bet: $(c \Leftrightarrow \neg\, a)$

We want to prove Claim by Contradiction:

Negated Claim: all three CAN win their bets

$$\neg\, C \equiv (a \Leftrightarrow \neg\, b) \wedge (b \Leftrightarrow \neg\, c) \wedge (c \Leftrightarrow \neg\, a)$$

Proof by Resolution: Example

Convert negated claim to CNF step:

Original negated claim:

$$\neg C \equiv (a \Leftrightarrow \neg b) \land (b \Leftrightarrow \neg c) \land (c \Leftrightarrow \neg a)$$

So:

$(a \Leftrightarrow \neg b) \equiv (a \Rightarrow \neg b) \land (\neg b \Rightarrow a)$ **by Equivalence Law**

$(a \Leftrightarrow \neg b) \equiv (\neg a \lor \neg b) \land (b \lor a)$ **by Implication Law**

And similarly for $(b \Leftrightarrow \neg c)$**,** $(c \Leftrightarrow \neg a)$**.**

We obtain:

$$\neg C \equiv (\neg a \lor \neg b) \land (b \lor a) \land (\neg b \lor \neg c) \land (b \lor c) \land (\neg c \lor \neg a) \land (c \lor a)$$

Proof by Resolution: Example

Resolution steps:

Using resolution rule $\neg C$ in CNF form:

$$(\neg a \vee \neg b)_1 \wedge (b \vee a)_2 \wedge (\neg b \vee \neg c)_3 \wedge (b \vee c)_4 \wedge (\neg c \vee \neg a)_5 \wedge (c \vee a)_6$$

Resolution applied to clauses 1 and 6

$$\frac{(\neg a \vee \neg b), (c \vee a)}{(\neg b \vee c)}$$

Produces new clause (7). Add it to the list

Known clauses:

1. $(\neg a \vee \neg b)$
2. $(b \vee a)$
3. $(\neg b \vee \neg c)$
4. $(b \vee c)$
5. $(\neg c \vee \neg a)$
6. $(c \vee a)$

Added clauses:

7. $(\neg b \vee c)$

Proof by Resolution: Example

Resolution steps:

Using resolution rule $\neg C$ in CNF form:

$$(\neg a \vee \neg b)_1 \wedge (b \vee a)_2 \wedge (\neg b \vee \neg c)_3 \wedge (b \vee c)_4 \wedge (\neg c \vee \neg a)_5 \wedge (c \vee a)_6$$

Resolution applied to clauses 4 and 7

$$\frac{(b \vee c),\ (\neg b \vee c)}{(c)}$$

Produces new clause (8). Add it to the list

Known clauses:

1. $(\neg a \vee \neg b)$
2. $(b \vee a)$
3. $(\neg b \vee \neg c)$
4. $(b \vee c)$
5. $(\neg c \vee \neg a)$
6. $(c \vee a)$

Added clauses:

7. $(\neg b \vee c)$
8. (c)

Proof by Resolution: Example

Resolution steps:

Using resolution rule $\neg C$ in CNF form:

$$(\neg a \lor \neg b)_1 \land (b \lor a)_2 \land (\neg b \lor \neg c)_3 \land (b \lor c)_4 \land (\neg c \lor \neg a)_5 \land (c \lor a)_6$$

Resolution applied to clauses 2 and 5

$$\frac{(b \lor a),\ \ (\neg c \lor \neg a)}{(b \lor \neg c)}$$

Produces new clause (9). Add it to the list

Known clauses:

1. $(\neg a \lor \neg b)$
2. $(b \lor a)$
3. $(\neg b \lor \neg c)$
4. $(b \lor c)$
5. $(\neg c \lor \neg a)$
6. $(c \lor a)$

Added clauses:

7. $(\neg b \lor c)$
8. (c)
9. $(b \lor \neg c)$

Proof by Resolution: Example

Resolution steps:

Using resolution rule $\neg C$ in CNF form:

$$(\neg a \lor \neg b)_1 \land (b \lor a)_2 \land (\neg b \lor \neg c)_3 \land (b \lor c)_4 \land (\neg c \lor \neg a)_5 \land (c \lor a)_6$$

Resolution applied to clauses 3 and 9

$$\frac{(\neg b \lor \neg c),\ (b \lor \neg c)}{(\neg c)}$$

Produces new clause (10). Add it to the list

Known clauses:

1. $(\neg a \lor \neg b)$
2. $(b \lor a)$
3. $(\neg b \lor \neg c)$
4. $(b \lor c)$
5. $(\neg c \lor \neg a)$
6. $(c \lor a)$

Added clauses:

7. $(\neg b \lor c)$
8. (c)
9. $(b \lor \neg c)$
10. $(\neg c)$

Proof by Resolution: Example

Resolution steps:

Using resolution rule:

$$(\neg a \lor \neg b)_1 \land (b \lor a)_2 \land (\neg b \lor \neg c)_3 \land (b \lor c)_4 \land (\neg c \lor \neg a)_5 \land (c \lor a)_6$$

Resolution applied to clauses 8 and 10

$$\frac{(\neg c),\ (c)}{\perp \text{ (empty clause)}}$$

No new clause to add. Negated claim $\neg C$ was proved false, so original claim C must be true.

Known clauses:

1. $(\neg a \lor \neg b)$
2. $(b \lor a)$
3. $(\neg b \lor \neg c)$
4. $(b \lor c)$
5. $(\neg c \lor \neg a)$
6. $(c \lor a)$

Added clauses:

7. $(\neg b \lor c)$
8. (c)
9. $(b \lor \neg c)$
10. $(\neg c)$

Proof by Resolution

The process of proving by resolution is as follows:

A. **Formalize the problem: "English to Propositional Logic"**
B. **derive** $KB \wedge \neg Q$
C. **convert** $KB \wedge \neg Q$ **into CNF ("standardized") form**
D. **Apply resolution rule to resulting clauses. New clauses will be generated (add them to the set if not already present)**
E. **Repeat (C) until:**
   a. **no new clause can be added (**KB **does NOT entail** Q**)**
   b. **last two clauses resolve to yield the empty clause (**KB **entails** Q**)**

Logical Entailment

So far, we have been asking the question:

"Does KB entail Q (does Q follow from KB)?"

$$KB \models Q$$

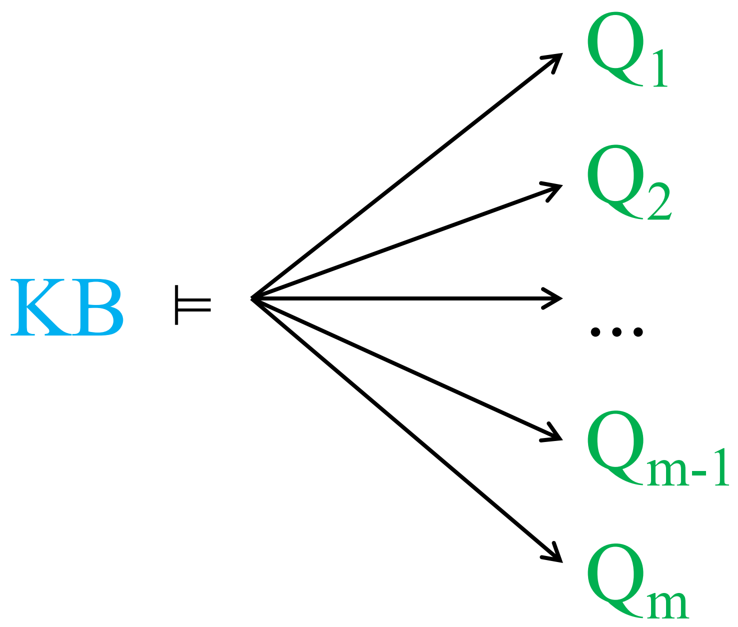
But we could ask the following question:

"Which Qs follow from KB?"

Logical Entailment

But we could ask the following question:

"Which Qs follow from KB?"



Logical Entailment

But we could ask the following question:

"Which Qs follow from KB?"

$KB \models$

Q_1 follows from KB

Q_2 does NOT

...

Q_{m-1} follows from KB

Q_m follows from KB

KB Agents

Sentences are physical configurations of the agent, and reasoning is a process of constructing new physical configurations from old ones.

Logical reasoning should ensure that the new configurations represent aspects of the world that actually follow from the aspects that the old configurations represent.

Knowledge-based Agents

function KB-AGENT(*percept*) **returns** an *action*
  persistent: *KB*, a knowledge base
    t, a counter, initially 0, indicating time

KBBEFORE

TELL(*KB*, MAKE-PERCEPT-SENTENCE(*percept*, t))
action ← ASK(*KB*, MAKE-ACTION-QUERY(t))
TELL(*KB*, MAKE-ACTION-SENTENCE(*action*, t))
$t \leftarrow t + 1$
return *action*

CURRENTKB

new percept

CURRENTKB ⇔ KBBEFORE ∧ percept

Logical Entailment with KB Agents

But we could ask the following question:

"Which ACTIONs follow from CURRENTKB?"

CURRENTKB $\models$

$ACTION_1$ follows from

$ACTION_2$ does NOT follow

...

$ACTION_{m-1}$ follows from

$ACTION_m$ follows from

Logical Entailment with KB Agents

But we could ask the following question:

"Which ACTIONs follow from CURRENTKB?"

$(\text{KBBEFORE} \wedge \text{percept}) \vDash$

- ACTION_1 follows from
- ACTION_2 does NOT follow
- ...
- ACTION_{m-1} follows from
- ACTION_m follows from