CS 480

Introduction to Artificial Intelligence

February 8, 2024

Announcements / Reminders

- **Please follow the Week 05 To Do List instructions (if you haven't already):**

- **Quiz #03: due on Sunday (02/11/24) at 11:59 PM CST**
  - **New quiz will be posted on Monday!**

- **New written assignment will be posted this week!**

- **Programming Assignment #01 due on Sunday (02/18/24) at 11:59 PM CST**

Plan for Today

- **Logical Agents and Reasoning**

Plan for Today

- Entailment
- Proof by Resolution

Inference: The idea

The idea:

Given everything (expressed as sentences) that we know, can we infer if some query (another sentence) is true or not (satisfied or not)?

Logical Entailment

A set of sentences (called premises) logically entails a sentence (called a conclusion) if and only if every truth assignment that satisfies the premises also satisfies the conclusion.

$$\text{PREMISES} \models \text{CONCLUSION}$$

Logical Entailment

Definition: A sentence KB entails a sentence Q (or Q follows from KB) if every model of KB is also a model of Q. We write:

$$KB \models Q$$

In other words:

- **For every interpretation in which KB is true, Q is also true**
- **"Whenever KB is true, Q is also true"**

Entailment: Deriving Conclusions

You can prove if:

$$KB \models Q$$

is true **in a number of ways:**

- **Model checking (enumeration)**
- **Truth table proof (enumeration)**
- **Proof by resolution**

You can also prove related sentences:

- **prove that** $KB \land \neg Q$ **is unsatisfiable (by contradiction)**
- **prove that** $KB \Rightarrow Q$ **is a tautology**

Entailment Proofs

$KB \equiv (p \Rightarrow q) \wedge (q \Rightarrow \neg r) \wedge (\neg p \Rightarrow \neg r)$ | $Q \equiv \neg r$

Prove that $(KB \Rightarrow Q) \Leftrightarrow T$
(Show that $KB \Rightarrow Q$ **is a tautology)**

Prove that $(KB \wedge \neg Q) \Leftrightarrow \perp$
(Show that $KB \wedge \neg Q$ **is a contradiction)**

Proof by model checking
Show that all models that are true **for** Q **are also** true **for** KB

Model	p	q	r	$p \Rightarrow q$	$q \Rightarrow \neg r$	$\neg p \Rightarrow \neg r$	KB	Q	$KB \Rightarrow Q$	$KB \wedge \neg Q$
M1	true	true	true	true	false	true	false	false	true	false
M2	true	true	false	true	true	true	true	true	true	false
M3	true	false	true	false	true	true	false	false	true	false
M4	true	false	false	false	true	true	false	true	true	false
M5	false	true	true	true	false	false	false	false	true	false
M6	false	true	false	true	true	true	true	true	true	false
M7	false	false	true	true	true	false	false	false	true	false
M8	false	false	false	true	true	true	true	true	true	false

Entailment Proofs

$KB \equiv (p \Rightarrow q) \wedge (q \Rightarrow \neg r) \wedge (\neg p \Rightarrow \neg r) \quad | \quad Q \equiv \neg r$

Prove that $(KB \Rightarrow Q) \Leftrightarrow$ T
(Show that $KB \Rightarrow Q$ **is a tautology)**

$KB \Rightarrow Q$ **is** true **for all models, so** KB **entails** Q

Prove that $(KB \wedge \neg Q) \Leftrightarrow \perp$
(Show that $KB \wedge \neg Q$ **is a contradiction)**

Proof by model checking
Show that all models that are true **for** Q **are also** true **for** KB

Model	p	q	r	$p \Rightarrow q$	$q \Rightarrow \neg r$	$\neg p \Rightarrow \neg r$	KB	Q	$KB \Rightarrow Q$	$KB \wedge \neg Q$
M1	true	true	true	true	false	true	false	false	true	false
M2	true	true	false	true	true	true	true	true	true	false
M3	true	false	true	false	true	true	false	false	true	false
M4	true	false	false	false	true	true	false	true	true	false
M5	false	true	true	true	false	false	false	false	true	false
M6	false	true	false	true	true	true	true	true	true	false
M7	false	false	true	true	true	false	false	false	true	false
M8	false	false	false	true	true	true	true	true	true	false

Entailment Proofs

$KB \equiv (p \Rightarrow q) \wedge (q \Rightarrow \neg r) \wedge (\neg p \Rightarrow \neg r)$ | $Q \equiv \neg r$

Prove that $(KB \Rightarrow Q) \Leftrightarrow T$
(Show that $KB \Rightarrow Q$ **is a tautology)**

$KB \Rightarrow Q$ **is** true **for all models, so** KB **entails** Q

Prove that $(KB \wedge \neg Q) \Leftrightarrow \perp$
(Show that $KB \wedge \neg Q$ **is a contradiction)**

$KB \wedge \neg Q$ **is** false **for all models, so** KB **entails** Q

Proof by model checking
Show that all models that are true **for** Q **are also** true **for** KB

Model	p	q	r	$p \Rightarrow q$	$q \Rightarrow \neg r$	$\neg p \Rightarrow \neg r$	KB	Q	$KB \Rightarrow Q$	$KB \wedge \neg Q$
M1	true	true	true	true	false	true	false	false	true	false
M2	true	true	false	true	true	true	true	true	true	false
M3	true	false	true	false	true	true	false	false	true	false
M4	true	false	false	false	true	true	false	true	true	false
M5	false	true	true	true	false	false	false	false	true	false
M6	false	true	false	true	true	true	true	true	true	false
M7	false	false	true	true	true	false	false	false	true	false
M8	false	false	false	true	true	true	true	true	true	false

Entailment Proofs

$KB \equiv (p \Rightarrow q) \wedge (q \Rightarrow \neg r) \wedge (\neg p \Rightarrow \neg r)$ | $Q \equiv \neg r$

Prove that $(KB \Rightarrow Q) \Leftrightarrow T$
(Show that $KB \Rightarrow Q$ is a tautology)

$KB \Rightarrow Q$ **is** true **for all models, so** KB **entails** Q

Prove that $(KB \wedge \neg Q) \Leftrightarrow \perp$
(Show that $KB \wedge \neg Q$ is a contradiction)

$KB \wedge \neg Q$ **is** false **for all models, so** KB **entails** Q

Proof by model checking
Show that all models that are true for Q are also true for KB

$M(KB) \subseteq M(Q)$ **so** KB **entails** Q

Model	p	q	r	$p \Rightarrow q$	$q \Rightarrow \neg r$	$\neg p \Rightarrow \neg r$	KB	Q	$KB \Rightarrow Q$	$KB \wedge \neg Q$
M1	true	true	true	true	false	true	false	false	true	false
M2	true	true	false	true	true	true	true	true	true	false
M3	true	false	true	false	true	true	false	false	true	false
M4	true	false	false	false	true	true	false	true	true	false
M5	false	true	true	true	false	false	false	false	true	false
M6	false	true	false	true	true	true	true	true	true	false
M7	false	false	true	true	true	false	false	false	true	false
M8	false	false	false	true	true	true	true	true	true	false

Model Checking: Q is Satisfiable

$$KB \equiv P1 \wedge P2 \wedge P3 \mid Q \equiv \ldots..$$

If $M(KB) \subseteq M(Q)$ Q **follows** KB, **otherwise it does NOT.**

Model	p	q	r	P1	P2	P3	KB	Q
M1	true	true	true	...	...	...	...	**false**
M2	true	true	false	...	...	...	...	**true**
M3	true	false	true	...	...	...	...	**false**
M4	true	false	false	...	...	...	...	**false**
M5	false	true	true	...	...	...	...	**false**
M6	false	true	false	...	...	...	...	**false**
M7	false	false	true	...	...	...	...	**false**
M8	false	false	false	...	...	...	...	**false**

Model Checking: Q is a Contradiction

$$KB \equiv P1 \wedge P2 \wedge P3 \mid Q \equiv .....$$

Regardless of $M(KB) \subseteq M(Q)$ Q **will NOT follow** KB.

Model	p	q	r	P1	P2	P3	KB	Q
M1	true	true	true	...	...	...	...	false
M2	true	true	false	...	...	...	...	false
M3	true	false	true	...	...	...	...	false
M4	true	false	false	...	...	...	...	false
M5	false	true	true	...	...	...	...	false
M6	false	true	false	...	...	...	...	false
M7	false	false	true	...	...	...	...	false
M8	false	false	false	...	...	...	...	false

Model Checking: Q is a Tautology

$$KB \equiv P1 \wedge P2 \wedge P3 \mid Q \equiv .....$$

Regardless of $M(KB) \subseteq M(Q)$ Q **WILL follow** KB.

Model	p	q	r	P1	P2	P3	KB	Q
M1	true	true	true	...	...	...	...	**true**
M2	true	true	false	...	...	...	...	**true**
M3	true	false	true	...	...	...	...	**true**
M4	true	false	false	...	...	...	...	**true**
M5	false	true	true	...	...	...	...	**true**
M6	false	true	false	...	...	...	...	**true**
M7	false	false	true	...	...	...	...	**true**
M8	false	false	false	...	...	...	...	**true**

Propositional Logic Calculus

Syntactic proof systems are called calculi.

To ensure that a calculus DOES NOT generate errors, two properties need to be satisfied:

- **A calculus is SOUND if every derived proposition follows semantically**
- **A calculus is COMPLETE if all semantic consequences can be derived**

Propositional Logic Calculus

Soundness:

The calculus does NOT produce any FALSE consequences

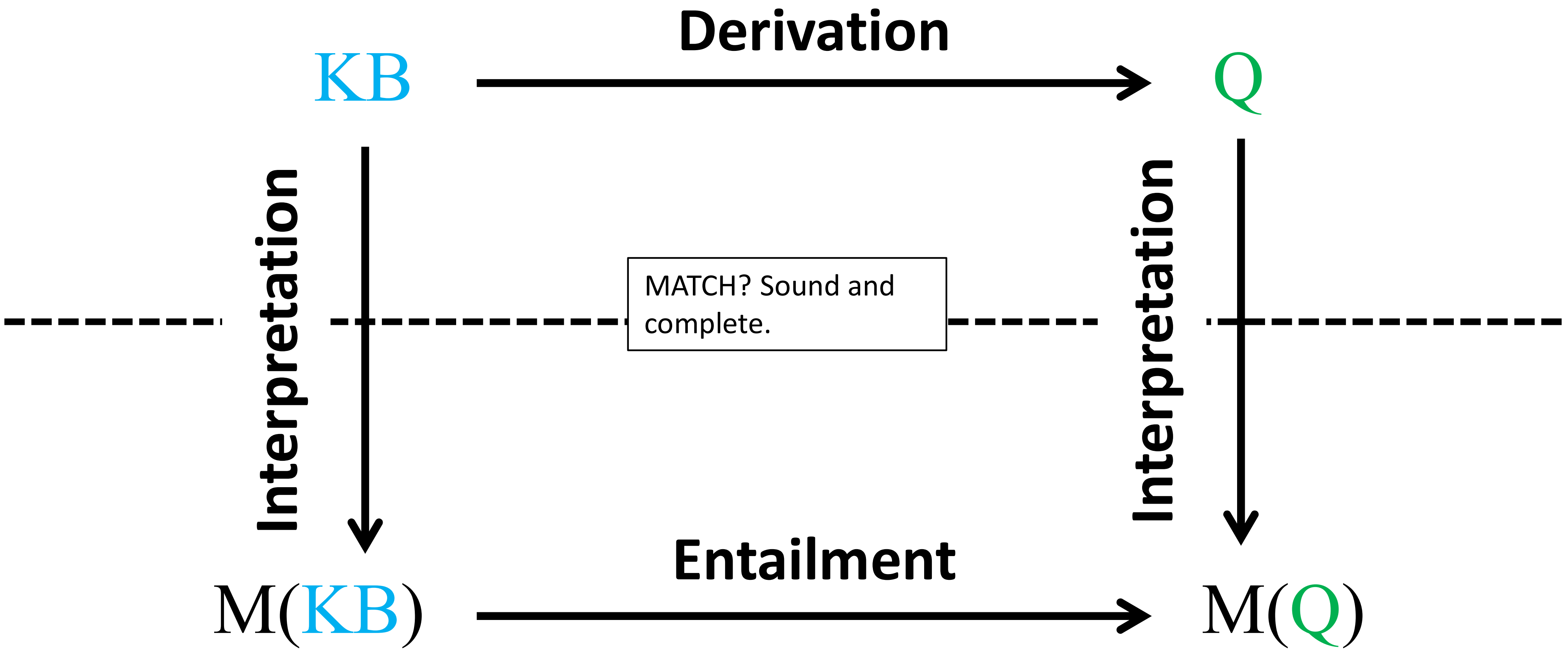
Completness:

A complete calculus ALWAYS finds a proof if the sentence to be proved follows from the knowledge base

If a calculus is sound and complete, then syntactic derivation and semantic entailment are two equivalent relations.

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):

$$KB \vDash Q$$

by proving that:

$$(KB \wedge \neg Q) \Leftrightarrow \perp$$

(show that $KB \wedge \neg Q$ is a contradiction / <u>empty clause</u>)

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

$P \lor Q$

$\neg P \lor R$

$\therefore Q \lor R$

Resolution (textbook)

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

derived clause (resolvent)

Proof by Resolution: ANY CLAIM

The process of proving by resolution is as follows:

A. Formalize the problem: “English to Propositional Logic”
   - state, using propositional logic, your claim C that you want to prove true
   - we start with a claim that C is true

B. Derive $\neg C$ (negate C)

C. Convert $\neg C$ into CNF (“standardized”) form:
   - $\neg C$ is transformed into a “conjunction of disjunctions”
   - $\neg C$ is transformed into clauses (could be one clause)

D. Apply unit resolution rule to resulting clauses. New clauses will be generated (add them to the set if not already present)

E. Repeat (step D) until:
   a. no new clause can be added (STOP: C is false)
   b. last two clauses resolve to yield the empty clause (STOP: C is true)

Proof by Resolution: ENTAILMENT

The process of proving by resolution is as follows:

A. Formalize the problem: "English to Propositional Logic"
   - state, using propositional logic, the entailment argument $KB \models Q$
   - we start with a claim that $KB \models Q$ is true (KB entails Q)

B. Derive $\neg (KB \models Q)$ (negated $KB \models Q$ is $KB \wedge \neg Q$)

C. Convert $KB \wedge \neg Q$ into CNF ("standardized") form:
   - $KB \wedge \neg Q$ is transformed into a "conjunction of disjunctions"
   - $KB \wedge \neg Q$ is transformed into clauses (could be one clause)

D. Apply unit resolution rule to resulting clauses. New clauses will be generated (add them to the set if not already present)

E. Repeat (step D) 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)

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

Symbol	Name	Alternative symbols*	Should be read
$\neg$	Negation	$\sim$, !	not
$\land$	(Logical) conjunction	$\bullet$, &	and
$\lor$	(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:

$$K_1 \wedge K_2 \wedge \dots \wedge K_m$$

of clauses. A clause Ki consists of a disjunction

$$(l_{i1} \vee l_{i2} \vee \dots \vee l_{in_i})$$

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

Conjunctive Normal Form (CNF)

Example:

$$(a \vee b \vee \neg c) \wedge (a \vee b \vee \neg c) \wedge (\neg b \vee \neg c)$$

where: a, b, c **are literals.**

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 complementary. The resolution rule deletes a pair of complementary 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 b and $\neg\, b$ are complementary. The clause ($b \wedge \neg\, b$) is a contradiction (an empty clause).

Unit Resolution

General resolution rule allows clauses with arbitrary number of literals

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

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

Factorization

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

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

Disjunction of multiple copies of literals ($\mathbf{d} \lor \mathbf{d}$) 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.

Logical Entailment

So far, we have been asking the question:

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

$$KB \models Q$$

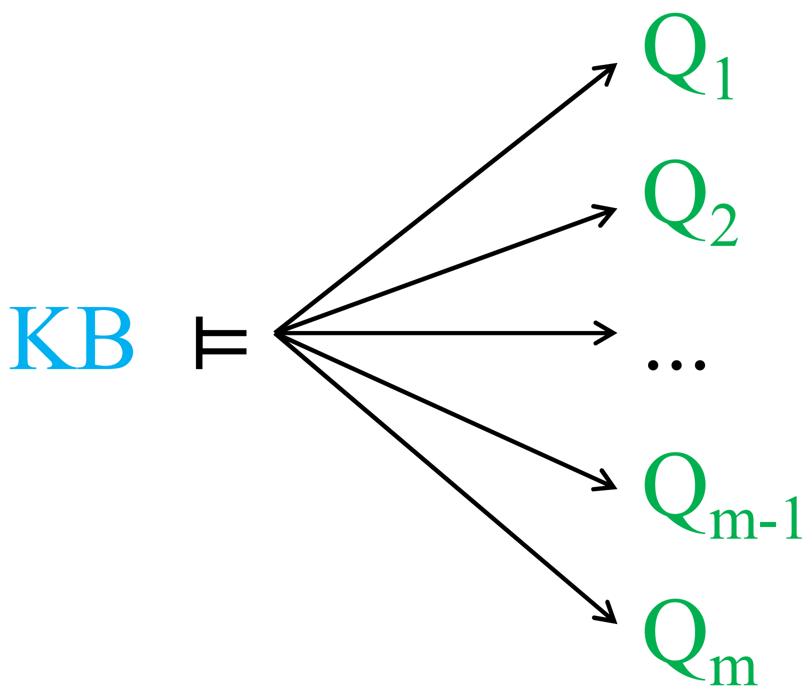
But we could ask the following question:

"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?"

$(KBBEFORE \wedge percept) \models$

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

Logical Entailment with KB Agents

Let's try a simpler example with just ONE ACTION to consider. The question is:

"Does ACTION follow from CURRENTKB?"

Test / prove:

$$\left(\text{KBBEFORE} \wedge \text{percept}\right) \vDash \text{ACTION}$$ follows from

to decide whether to apply ACTION or not.

Proof by Resolution

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

$$KB \models Q$$

by proving that:

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

(show that $KB \wedge \neg Q$ **is a contradiction / <u>empty clause</u>)**

KB Agent: Should I Stay or Should I Go

Sensor

PERCEPT:

Knowledge Base and Inference System

Actuator

ACTION:

KB Agent: Street Crosser

Environment: Street

KB Agent: Should I Stay or Should I Go

Sensor

TRAFFIC LEFT / RIGHT

Knowledge Base and Inference System

Actuator

WALK OR STAY

KB Agent: Street Crosser

Environment: Street

Proof by Resolution: ENTAILMENT

The process of proving by resolution is as follows:

A. Formalize the problem: "English to Propositional Logic"
   - state, using propositional logic, the entailment argument $KB \vDash Q$
   - we start with a claim that $KB \vDash Q$ is true (KB entails Q)

B. Derive $\neg(KB \vDash Q)$ (negated $KB \vDash Q$ is $KB \wedge \neg Q$)

C. Convert $KB \wedge \neg Q$ into CNF ("standardized") form:
   - $KB \wedge \neg Q$ is transformed into a "conjunction of disjunctions"
   - $KB \wedge \neg Q$ is transformed into clauses (could be one clause)

D. Apply unit resolution rule to resulting clauses. New clauses will be generated (add them to the set if not already present)

E. Repeat (step D) 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)

Agent: "Built-In" Knowledge Base KB

English:

A: "Walk if and only if there is NO traffic coming from the left AND NO traffic coming from the right."

or

B: "DON'T walk if and only if there is traffic coming from the left OR traffic coming from the right."

Agent: "Built-In" Knowledge Base KB

English and Propositional Logic:

A: "Walk if and only if there is NO traffic coming from the left AND NO traffic coming from the right."

$$walk \Leftrightarrow (\neg trafficLeft \wedge \neg trafficRight)$$

or

B: "DON'T walk if and only if there is traffic coming from the left OR traffic coming from the right."

$$\neg walk \Leftrightarrow (trafficLeft \vee trafficRight)$$

Agent: "Built-In" Knowledge Base KB

"Walk if and only if there is NO traffic coming from the left AND NO traffic coming from the right."

$$\mathrm{walk} \Leftrightarrow (\neg \mathrm{trafficLeft} \wedge \neg \mathrm{trafficRight})$$

Agent: "Built-In" Knowledge Base KB

"Walk if and only if there is NO traffic coming from the left AND NO traffic coming from the right."

$$w \Leftrightarrow (\neg tL \wedge \neg tR)$$

Agent: Possible PERCEPT

"NO traffic coming from the left AND NO traffic coming from the right."

$(\neg trafficLeft \wedge \neg trafficRight)$

"traffic coming from the left AND traffic coming from the right."

$(trafficLeft \wedge trafficRight)$

"NO traffic coming from the left AND traffic coming from the right."

$(\neg trafficLeft \wedge trafficRight)$

"traffic coming from the left AND NO traffic coming from the right."

$(trafficLeft \wedge \neg trafficRight)$

Agent: Possible PERCEPT

"NO traffic coming from the left AND NO traffic coming from the right."

$(\neg tL \wedge \neg tR)$

"traffic coming from the left AND traffic coming from the right."

$(tL \wedge tR)$

"NO traffic coming from the left AND traffic coming from the right."

$(\neg tL \wedge tR)$

"traffic coming from the left AND NO traffic coming from the right."

$(tL \wedge \neg tR)$

Agent: Possible PERCEPT

"NO traffic coming from the left AND NO traffic coming from the right."

$(\neg tL \wedge \neg tR)$

"traffic coming from the left AND traffic coming from the right."

$(tL \wedge tR)$

"NO traffic coming from the left AND traffic coming from the right."

$(\neg tL \wedge tR)$

"traffic coming from the left AND NO traffic coming from the right."

$(tL \wedge \neg tR)$

Agent: Possible Query Q

“Walk” (GO)

w

“DON’T Walk” (STAY)

$\neg w$

Should I walk If tR and tL?

Sensor

TRAFFIC LEFT / RIGHT

$PERCEPT \equiv (tR) \wedge (tL)$

Knowledge Base and Inference System

Should I walk?

Actuator

WALK OR STAY??

Traffic coming from the left: YES!

Traffic coming from the right: YES!

KB Agent: Street Crosser

Environment: Street

Proof by Resolution: ENTAILMENT

The process of proving by resolution is as follows:

A. Formalize the problem: "English to Propositional Logic"
   - state, using propositional logic, the entailment argument $KB \vDash Q$
   - we start with a claim that $KB \vDash Q$ is true (KB entails Q)

B. Derive $\neg (KB \vDash Q)$ (negated $KB \vDash Q$ is $KB \wedge \neg Q$)

C. Convert $KB \wedge \neg Q$ into CNF ("standardized") form:
   - $KB \wedge \neg Q$ is transformed into a "conjunction of disjunctions"
   - $KB \wedge \neg Q$ is transformed into clauses (could be one clause)

D. Apply unit resolution rule to resulting clauses. New clauses will be generated (add them to the set if not already present)

E. Repeat (step D) 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)

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

CURRENTKB

new PERCEPT

CURRENTKB $\Leftrightarrow$ KBBEFORE $\wedge$ PERCEPT

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*

KB_N

new PERCEPT

$$KB_N \Leftrightarrow KBBEFORE \wedge PERCEPT$$

Should I walk: Proof by Resolution

A. Claim that $KB_N \vDash Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \vDash Q$**) is true:**

$$[(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL)] \vDash (w)$$

Should I walk: Proof by Resolution

A. Claim that $KB_N \vDash Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \vDash Q$**) is true:**

$$[(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL)] \vDash (w)$$

B. Negated claim: $KB_N \wedge \neg Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \wedge \neg Q$**) :**

$$[(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL)] \wedge (\neg w)$$

Should I walk: Proof by Resolution

A. Claim that $KB_N \vDash Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \vDash Q$**) is true:**

$$[(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL)] \vDash (w)$$

B. Negated claim: $KB_N \wedge \neg Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \wedge \neg Q$**) :**

$$(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL) \wedge (\neg w)$$

C. Convert negated claim $KB_N \wedge \neg Q$ **to CNF:**

Should I walk: Proof by Resolution

A. Claim that $KB_N \vDash Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \vDash Q$**) is true:**

$$[(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL)] \vDash (w)$$

B. Negated claim: $KB_N \wedge \neg Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \wedge \neg Q$**) :**

KBBEFORE

Already in CNF

$$(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL) \wedge (\neg w)$$

NOT in CNF

C. Convert negated claim $KB_N \wedge \neg Q$ **to CNF:**

Convert Negated Claim to CNF

$$KB_N \wedge \neg Q \equiv (w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL) \wedge (\neg w)$$

$$(w \Rightarrow (\neg tL \wedge \neg tR)) \wedge ((\neg tL \wedge \neg tR) \Rightarrow w) \wedge (tR) \wedge (tL) \wedge (\neg w)$$

by Biconditional Elimination

$$(\neg w \vee (\neg tL \wedge \neg tR)) \wedge (\neg(\neg tL \wedge \neg tR) \vee w) \wedge (tR) \wedge (tL) \wedge (\neg w)$$

by Implication Law

$$((\neg w \vee \neg tL) \wedge (\neg w \vee \neg tR)) \wedge (\neg(\neg tL \wedge \neg tR) \vee w) \wedge (tR) \wedge (tL) \wedge (\neg w)$$

by Distributivity Rule

$$((\neg w \vee \neg tL) \wedge (\neg w \vee \neg tR)) \wedge ((\neg\neg tL \vee \neg\neg tR) \vee w) \wedge (tR) \wedge (tL) \wedge (\neg w)$$

by De Morgan's Rule

$$((\neg w \vee \neg tL) \wedge (\neg w \vee \neg tR)) \wedge ((tL \vee tR) \vee w) \wedge (tR) \wedge (tL) \wedge (\neg w)$$

by Double Negation Law

$$(\neg w \vee \neg tL) \wedge (\neg w \vee \neg tR) \wedge (tL \vee tR \vee w) \wedge (tR) \wedge (tL) \wedge (\neg w)$$

remove extraneous parentheses

Should I walk: Proof by Resolution

A. Claim that $KB_N \vDash Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \vDash Q$**) is true:**

$$[(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL)] \vDash (w)$$

B. Negated claim: $KB_N \wedge \neg Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \wedge \neg Q$**) :**

$$(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL) \wedge (\neg w)$$

C. Convert negated claim $KB_N \wedge \neg Q$ **to CNF:**

$$(w \vee tL \vee tR) \wedge (\neg tL \vee \neg w) \wedge (\neg tR \vee \neg w) \wedge (tR) \wedge (tL) \wedge (\neg w)$$

Should I walk: Proof by Resolution

A. Claim that $KB_N \vDash Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \vDash Q$**) is true:**

$$KB_N \vDash Q \equiv [(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL)] \vDash (w)$$

B. Negated claim: $KB_N \wedge \neg Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \wedge \neg Q$**) :**

$$KB_N \wedge \neg Q \equiv [(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL)] \wedge (\neg w)$$

C. Convert negated claim $KB_N \wedge \neg Q$ **to CNF (a set of six clauses):**

$$KB_N \wedge \neg Q \equiv (w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (tR)_4 \wedge (tL)_5 \wedge (\neg w)_6$$

Should I walk: Proof by Resolution

A. Claim that $KB_N \models Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \models Q$**) is true:**

$$KB_N \models Q \equiv [(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL)] \models (w)$$

B. Negated claim: $KB_N \wedge \neg Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \wedge \neg Q$**) :**

$$KB_N \wedge \neg Q \equiv [(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL)] \wedge (\neg w)$$

C. Convert negated claim $KB_N \wedge \neg Q$ **to CNF (a set of six clauses):**

$$KB_N \wedge \neg Q \equiv (w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (tR)_4 \wedge (tL)_5 \wedge (\neg w)_6$$

D and E. Apply unit resolution steps until no new clause can be added or empty clause

Should I walk: Proof by Resolution

Prove:

$$KB_N \wedge \neg Q \equiv$$

$$(w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (tR)_4 \wedge (tL)_5 \wedge (\neg w)_6$$

Should I walk: Proof by Resolution

Prove:

$$KB_N \wedge \neg Q \equiv$$

$$(w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (tR)_4 \wedge (tL)_5 \wedge (\neg w)_6$$

Known clauses:

1. $(w \vee tL \vee tR)$
2. $(\neg tL \vee \neg w)$
3. $(\neg tR \vee \neg w)$
4. (tR)
5. (tL)
6. $(\neg w)$

Added clauses:

Proof by Resolution: Example

Prove:

$$KB_N \wedge \neg Q \equiv$$

$$(w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (tR)_4 \wedge (tL)_5 \wedge (\neg w)_6$$

Resolution applied to clauses 1 and 6

$$\frac{(w \vee tL \vee tR),\ (\neg w)}{(tL \vee tR)}$$

Produces a new clause ($tL \vee tR$). We can add it to the list as clause (7).

Known clauses:

1. $(w \vee tL \vee tR)$
2. $(\neg tL \vee \neg w)$
3. $(\neg tR \vee \neg w)$
4. (tR)
5. (tL)
6. $(\neg w)$

Added clauses:

7. $(tL \vee tR)$

Proof by Resolution: Example

Prove:

$$KB_N \wedge \neg Q \equiv$$

$$(w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (tR)_4 \wedge (tL)_5 \wedge (\neg w)_6$$

Resolution applied to clauses 2 and 5

$$\frac{(\neg tL \vee \neg w),\ (tL)}{(\neg w)}$$

Produces a clause ($\neg w$), but we already have it (6). Don't add it to the list.

Known clauses:

1. $(w \vee tL \vee tR)$
2. $(\neg tL \vee \neg w)$
3. $(\neg tR \vee \neg w)$
4. (tR)
5. (tL)
6. $(\neg w)$

Added clauses:

7. $(tL \vee tR)$

Proof by Resolution: Example

Prove:

$$KB_N \land \neg Q \equiv$$

$$(w \lor tL \lor tR)_1 \land (\neg tL \lor \neg w)_2 \land (\neg tR \lor \neg w)_3 \land (tR)_4 \land (tL)_5 \land (\neg w)_6$$

Resolution applied to clauses 3 and 4

$$\frac{(\neg tR \lor \neg w),\ (tR)}{(\neg w)}$$

Produces a clause ($\neg w$), but we already have it (6). Don't add it to the list.

Known clauses:

1. $(w \lor tL \lor tR)$
2. $(\neg tL \lor \neg w)$
3. $(\neg tR \lor \neg w)$
4. (tR)
5. (tL)
6. $(\neg w)$

Added clauses:

7. $(tL \lor tR)$

Proof by Resolution: Example

Prove:

$$KB_N \land \neg Q \equiv$$

$$(w \lor tL \lor tR)_1 \land (\neg tL \lor \neg w)_2 \land (\neg tR \lor \neg w)_3 \land (tR)_4 \land (tL)_5 \land (\neg w)_6$$

Resolution applied to clauses 2 and 7

$$\frac{(\neg tL \lor \neg w),\ (tL \lor tR)}{(\neg w \lor tR)}$$

Produces a new clause $(\neg w \lor tR)$. We can add it to the list as clause (8).

Known clauses:

1. $(w \lor tL \lor tR)$
2. $(\neg tL \lor \neg w)$
3. $(\neg tR \lor \neg w)$
4. (tR)
5. (tL)
6. $(\neg w)$

Added clauses:

7. $(tL \lor tR)$
8. $(\neg w \lor tR)$

Proof by Resolution: Example

Prove:

$$KB_N \wedge \neg Q \equiv$$

$$(w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (tR)_4 \wedge (tL)_5 \wedge (\neg w)_6$$

Resolution applied to clauses 1 and 8

$$\frac{(w \vee tL \vee tR), (\neg w \vee tR)}{(tL \vee tR)}$$

Produces a clause ($tL \vee tR$), but we already have it (7). Don't add it to the list.

Known clauses:

1. $(w \vee tL \vee tR)$
2. $(\neg tL \vee \neg w)$
3. $(\neg tR \vee \neg w)$
4. (tR)
5. (tL)
6. $(\neg w)$

Added clauses:

7. $(tL \vee tR)$
8. $(\neg w \vee tR)$

Proof by Resolution: Example

Prove:

$$KB_N \wedge \neg Q \equiv$$

$$(w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (tR)_4 \wedge (tL)_5 \wedge (\neg w)_6$$

Resolution applied to clauses 3 and 8

$$\frac{(\neg tR \vee \neg w),\ (\neg w \vee tR)}{(\neg w)}$$

Produces a clause ($\neg w$), but we already have it (6). Don't add it to the list.

Known clauses:

1. $(w \vee tL \vee tR)$
2. $(\neg tL \vee \neg w)$
3. $(\neg tR \vee \neg w)$
4. (tR)
5. (tL)
6. $(\neg w)$

Added clauses:

7. $(tL \vee tR)$
8. $(\neg w \vee tR)$

Proof by Resolution: Example

Prove:

$$\mathrm{KB_N} \wedge \neg \mathrm{Q} \equiv$$

$$(\mathrm{w} \vee \mathrm{tL} \vee \mathrm{tR})_1 \wedge (\neg \mathrm{tL} \vee \neg \mathrm{w})_2 \wedge (\neg \mathrm{tR} \vee \neg \mathrm{w})_3 \wedge (\mathrm{tR})_4 \wedge (\mathrm{tL})_5 \wedge (\neg \mathrm{w})_6$$

Resolution applied to clauses 3 and 7

$$\frac{(\neg \mathrm{tR} \vee \neg \mathrm{w}),\ (\mathrm{tL} \vee \mathrm{tR})}{(\neg \mathrm{w} \vee \mathrm{tL})}$$

Produces a new clause ($\neg \mathrm{w} \vee \mathrm{tL}$). We can add it to the list as clause (9).

Known clauses:

1. $(\mathrm{w} \vee \mathrm{tL} \vee \mathrm{tR})$
2. $(\neg \mathrm{tL} \vee \neg \mathrm{w})$
3. $(\neg \mathrm{tR} \vee \neg \mathrm{w})$
4. (tR)
5. (tL)
6. $(\neg \mathrm{w})$

Added clauses:

7. $(\mathrm{tL} \vee \mathrm{tR})$
8. $(\neg \mathrm{w} \vee \mathrm{tR})$
9. $(\neg \mathrm{w} \vee \mathrm{tL})$

Proof by Resolution: Example

Prove:

$$KB_N \wedge \neg Q \equiv$$

$$(w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (tR)_4 \wedge (tL)_5 \wedge (\neg w)_6$$

Resolution applied to clauses 2 and 9

$$\frac{(\neg tL \vee \neg w),\ (\neg w \vee tL)}{(\neg w)}$$

Produces a clause ($\neg w$), but we already have it (6). Don't add it to the list.

Known clauses:

1. $(w \vee tL \vee tR)$
2. $(\neg tL \vee \neg w)$
3. $(\neg tR \vee \neg w)$
4. (tR)
5. (tL)
6. $(\neg w)$

Added clauses:

7. $(tL \vee tR)$
8. $(\neg w \vee tR)$
9. $(\neg w \vee tL)$

Proof by Resolution: Example

Prove:

$$KB_N \wedge \neg Q \equiv$$

$$(w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (tR)_4 \wedge (tL)_5 \wedge (\neg w)_6$$

At this point, we tried to resolve all promising clause pairs, but we have not reached an empty clause → KB_N does NOT entail Q.

Given PERCEPTS: $(tR) \wedge (tL)$

we should NOT apply action walk (w) and stay.

Known clauses:

1. $(w \vee tL \vee tR)$
2. $(\neg tL \vee \neg w)$
3. $(\neg tR \vee \neg w)$
4. (tR)
5. (tL)
6. $(\neg w)$

Added clauses:

7. $(tL \vee tR)$
8. $(\neg w \vee tR)$
9. $(\neg w \vee tL)$

Should I walk If $\neg$tR and $\neg$tL?

Sensor

TRAFFIC LEFT / RIGHT

$PERCEPT \equiv (\neg tR) \wedge (\neg tL)$

Knowledge Base and Inference System

Should I walk?

Actuator

WALK OR STAY??

Traffic coming from the left: NO!

Traffic coming from the right: NO!

KB Agent: Street Crosser

Environment: Street

Should I walk: Proof by Resolution

A. Claim that $KB_N \vDash Q$ (which really is $(KBBEFORE \wedge PERCEPT) \vDash Q$) is true:

$$[(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (\neg tR) \wedge (\neg tL)] \vDash (w)$$

Should I walk: Proof by Resolution

A. Claim that $KB_N \vDash Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \vDash Q$**) is true:**

$$[(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (\neg tR) \wedge (\neg tL)] \vDash (w)$$

B. Negated claim: $KB_N \wedge \neg Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \wedge \neg Q$**) :**

$$[(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (\neg tR) \wedge (\neg tL)] \wedge (\neg w)$$

Should I walk: Proof by Resolution

A. Claim that $KB_N \models Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \models Q$**) is true:**

$$[(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (\neg tR) \wedge (\neg tL)] \models (w)$$

B. Negated claim: $KB_N \wedge \neg Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \wedge \neg Q$**) :**

$$(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (\neg tR) \wedge (\neg tL) \wedge (\neg w)$$

C. Convert negated claim $KB_N \wedge \neg Q$ **to CNF:**

Should I walk: Proof by Resolution

A. Claim that $KB_N \models Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \models Q$**) is true:**

$$[(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (tR) \wedge (tL)] \models (w)$$

B. Negated claim: $KB_N \wedge \neg Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \wedge \neg Q$**) :**

$$\underbrace{(w \Leftrightarrow (\neg tL \wedge \neg tR))}_{\text{NOT in CNF}}^{\text{KBBEFORE}} \wedge \overbrace{(\neg tR) \wedge (\neg tL) \wedge (\neg w)}^{\text{Already in CNF}}$$

C. Convert negated claim $KB_N \wedge \neg Q$ **to CNF:**

Convert Negated Claim to CNF

$$KB_N \wedge \neg Q \equiv (w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (\neg tR) \wedge (\neg tL) \wedge (\neg w)$$

$$(w \Rightarrow (\neg tL \wedge \neg tR)) \wedge ((\neg tL \wedge \neg tR) \Rightarrow w) \wedge (\neg tR) \wedge (\neg tL) \wedge (\neg w)$$

by Biconditional Elimination

$$(\neg w \vee (\neg tL \wedge \neg tR)) \wedge (\neg(\neg tL \wedge \neg tR) \vee w) \wedge (\neg tR) \wedge (\neg tL) \wedge (\neg w)$$

by Implication Law

$$((\neg w \vee \neg tL) \wedge (\neg w \vee \neg tR)) \wedge (\neg(\neg tL \wedge \neg tR) \vee w) \wedge (\neg tR) \wedge (\neg tL) \wedge (\neg w)$$

by Distributivity Rule

$$((\neg w \vee \neg tL) \wedge (\neg w \vee \neg tR)) \wedge ((\neg\neg tL \vee \neg\neg tR) \vee w) \wedge (\neg tR) \wedge (\neg tL) \wedge (\neg w)$$

by De Morgan's Rule

$$((\neg w \vee \neg tL) \wedge (\neg w \vee \neg tR)) \wedge ((tL \vee tR) \vee w) \wedge (\neg tR) \wedge (\neg tL) \wedge (\neg w)$$

by Double Negation Law

$$(\neg w \vee \neg tL) \wedge (\neg w \vee \neg tR) \wedge (tL \vee tR \vee w) \wedge (\neg tR) \wedge (\neg tL) \wedge (\neg w)$$

remove extraneous parentheses

Should I walk: Proof by Resolution

A. Claim that $KB_N \vDash Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \vDash Q$**) is true:**

$$[(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (\neg tR) \wedge (\neg tL)] \vDash (w)$$

B. Negated claim: $KB_N \wedge \neg Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \wedge \neg Q$**) :**

$$(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (\neg tR) \wedge (\neg tL) \wedge (\neg w)$$

C. Convert negated claim $KB_N \wedge \neg Q$ **to CNF:**

$$(w \vee tL \vee tR) \wedge (\neg tL \vee \neg w) \wedge (\neg tR \vee \neg w) \wedge (\neg tR) \wedge (\neg tL) \wedge (\neg w)$$

Should I walk: Proof by Resolution

A. Claim that $KB_N \models Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \models Q$**) is true:**

$$KB_N \models Q \equiv [(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (\neg tR) \wedge (\neg tL)] \models (w)$$

B. Negated claim: $KB_N \wedge \neg Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \wedge \neg Q$**) :**

$$KB_N \wedge \neg Q \equiv [(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (\neg tR) \wedge (\neg tL)] \wedge (\neg w)$$

C. Convert negated claim $KB_N \wedge \neg Q$ **to CNF (a set of six clauses):**

$$KB_N \wedge \neg Q \equiv (w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (\neg tR)_4 \wedge (\neg tL)_5 \wedge (\neg w)_6$$

Should I walk: Proof by Resolution

A. Claim that $KB_N \vDash Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \vDash Q$**) is true:**

$$KB_N \vDash Q \equiv [(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (\neg tR) \wedge (\neg tL)] \vDash (w)$$

B. Negated claim: $KB_N \wedge \neg Q$ **(which really is** $(KBBEFORE \wedge PERCEPT) \wedge \neg Q$**) :**

$$KB_N \wedge \neg Q \equiv [(w \Leftrightarrow (\neg tL \wedge \neg tR)) \wedge (\neg tR) \wedge (\neg tL)] \wedge (\neg w)$$

C. Convert negated claim $KB_N \wedge \neg Q$ **to CNF (a set of six clauses):**

$$KB_N \wedge \neg Q \equiv (w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (\neg tR)_4 \wedge (\neg tL)_5 \wedge (\neg w)_6$$

D and E. Apply unit resolution steps until no new clause can be added or empty clause

Should I walk: Proof by Resolution

Prove:

$$KB_N \wedge \neg Q \equiv$$

$$(w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (\neg tR)_4 \wedge (\neg tL)_5 \wedge (\neg w)_6$$

Should I walk: Proof by Resolution

Prove:

$$KB_N \land \neg Q \equiv$$

$$(w \lor tL \lor tR)_1 \land (\neg tL \lor \neg w)_2 \land (\neg tR \lor \neg w)_3 \land (\neg tR)_4 \land (\neg tL)_5 \land (\neg w)_6$$

Known clauses:

1. $(w \lor tL \lor tR)$
2. $(\neg tL \lor \neg w)$
3. $(\neg tR \lor \neg w)$
4. $(\neg tR)$
5. $(\neg tL)$
6. $(\neg w)$

Added clauses:

Should I walk: Proof by Resolution

Prove:

$$KB_N \wedge \neg Q \equiv$$

$$(w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (\neg tR)_4 \wedge (\neg tL)_5 \wedge (\neg w)_6$$

Resolution applied to clauses 1 and 6

$$\frac{(w \vee tL \vee tR),\ (\neg w)}{(tL \vee tR)}$$

Produces a new clause ($tL \vee tR$). We can add it to the list as clause (7).

Known clauses:

1. $(w \vee tL \vee tR)$
2. $(\neg tL \vee \neg w)$
3. $(\neg tR \vee \neg w)$
4. $(\neg tR)$
5. $(\neg tL)$
6. $(\neg w)$

Added clauses:

7. $(tL \vee tR)$

Should I walk: Proof by Resolution

Prove:

$$KB_N \land \neg Q \equiv$$

$$(w \lor tL \lor tR)_1 \land (\neg tL \lor \neg w)_2 \land (\neg tR \lor \neg w)_3 \land (\neg tR)_4 \land (\neg tL)_5 \land (\neg w)_6$$

Resolution applied to clauses 1 and 6

$$\frac{(w \lor tL \lor tR),\ (\neg w)}{(tL \lor tR)}$$

Produces a new clause ($tL \lor tR$). We can add it to the list as clause (7).

Known clauses:

1. $(w \lor tL \lor tR)$
2. $(\neg tL \lor \neg w)$
3. $(\neg tR \lor \neg w)$
4. $(\neg tR)$
5. $(\neg tL)$
6. $(\neg w)$

Added clauses:

7. $(tL \lor tR)$

Should I walk: Proof by Resolution

Prove:

$$KB_N \wedge \neg Q \equiv$$

$$(w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (\neg tR)_4 \wedge (\neg tL)_5 \wedge (\neg w)_6$$

Resolution applied to clauses 4 and 7

$$\frac{(\neg tR),\ (tL \vee tR)}{(tL)}$$

Produces a new clause (tL). We can add it to the list as clause (8).

Known clauses:

1. $(w \vee tL \vee tR)$
2. $(\neg tL \vee \neg w)$
3. $(\neg tR \vee \neg w)$
4. $(\neg tR)$
5. $(\neg tL)$
6. $(\neg w)$

Added clauses:

7. $(tL \vee tR)$
8. (tL)

Should I walk: Proof by Resolution

Prove:

$$KB_N \wedge \neg Q \equiv$$

$$(w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (\neg tR)_4 \wedge (\neg tL)_5 \wedge (\neg w)_6$$

Resolution applied to clauses 5 and 8

$$\frac{(\neg tL), (tL)}{()}$$

Produces an empty clause / contradiction.

Stop.

Known clauses:

1. $(w \vee tL \vee tR)$
2. $(\neg tL \vee \neg w)$
3. $(\neg tR \vee \neg w)$
4. $(\neg tR)$
5. $(\neg tL)$
6. $(\neg w)$

Added clauses:

7. $(tL \vee tR)$
8. (tL)

Should I walk: Proof by Resolution

Prove:

$$KB_N \wedge \neg Q \equiv$$

$$(w \vee tL \vee tR)_1 \wedge (\neg tL \vee \neg w)_2 \wedge (\neg tR \vee \neg w)_3 \wedge (\neg tR)_4 \wedge (\neg tL)_5 \wedge (\neg w)_6$$

At this point, we tried to resolve all promising clause pairs and we reached an empty clause $\rightarrow$ KB entails Q.

Given PERCEPTS: $(\neg tR) \wedge (\neg tL)$

we should apply action walk (w) and go.

Known clauses:

1. $(w \vee tL \vee tR)$
2. $(\neg tL \vee \neg w)$
3. $(\neg tR \vee \neg w)$
4. $(\neg tR)$
5. $(\neg tL)$
6. $(\neg w)$

Added clauses:

7. $(tL \vee tR)$
8. (tL)

Street Crosser Agent: Summary

Applying resolution to all possible $\mathrm{PERCEPTS}$ **and** Q **(only one) combinations and decisions:**

- $\mathrm{PERCEPTS} \equiv (\neg tR) \wedge (\neg tL) \rightarrow$ **WALK**
- $\mathrm{PERCEPTS} \equiv (tR) \wedge (\neg tL) \rightarrow$ **DON'T WALK**
- $\mathrm{PERCEPTS} \equiv (\neg tR) \wedge (tL) \rightarrow$ **DON'T WALK**
- $\mathrm{PERCEPTS} \equiv (tR) \wedge (tL) \rightarrow$ **DON'T WALK**

allowed our agent to:

- **reason and make decisions**
- **learn: percepts $\rightarrow$ decision is new knowledge!**

Knowledge Base: But wait...

If I keep adding multiple new PERCEPTS to the knowledge base KB, for example:

$$\text{PERCEPTS1} \equiv (\neg tR) \wedge (\neg tL)$$

$$\text{PERCEPTS2} \equiv (tR) \wedge (tL)$$

I may end up with a contradiction in my KB, right?

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

Knowledge-based Agents

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

  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*

KBBEFORE

"time stamps"

CURRENTKB

new percept

CURRENTKB ⇔ KBBEFORE ∧ percept

Definite Clauses

A sentence can be called a definite clause if and only if it is a disjunction of literals of which EXACTLY one is positive. For example:

$$(\neg p \vee \neg q \vee r)$$

is a definite clause.

This:

$$(x \vee \neg y \vee z)$$

is NOT a definite clause (more than one positive literal)

Horn Clauses

A sentence can be called a Horn clause if and only if it is a disjunction of literals of which AT MOST one is positive. For example:

$$(\neg p \vee \neg q \vee r)$$

is a Horn clause. This:

$$(x \vee \neg y \vee z)$$

is NOT a Horn clause. However, this:

$$(\neg d \vee \neg e \vee \neg f)$$

is a Horn clause (goal clause $\rightarrow$ no positive literals).

Definite / Horn Clauses: Why Bother?

Reasons to use definite / Horn clauses:

- **resolution of two Horn clauses, yields a Horn clause**
- **definite clauses can be rewritten as implications:**

$$(\neg p \vee \neg q \vee r) \equiv (p \wedge q) \Rightarrow r$$

- **inference with Horn clauses can be realized using two human-friendly (-understandable) algorithms: forward- and backward-chaining**
- **deciding entailment with Horn clauses is** $O(|KB|)$

Definite / Horn Clauses: Why Bother?

Reasons to use definite / Horn clauses:

- **resolution of two Horn clauses, yields a Horn clause**
- **definite clauses can be rewritten as implications:**

$$(\neg p \vee \neg q \vee r) \equiv (p \wedge q) \Rightarrow r$$

- **inference with Horn clauses can be realized using two human-friendly (-understandable) algorithms: forward- and backward-chaining**
- **deciding entailment with Horn clauses is $O(|KB|)$**

Types of Horn Clauses

Types of Horn clauses (at most one positive literal):

Type of Horn clause	Disjunction form	Implication form	Read in English as
Definite clause	$(\neg p \vee \neg q \vee \ldots \vee \neg t \vee u)$	$(p \wedge q \wedge \ldots \wedge t) \Rightarrow u$	assume that, if *p* and *q* and ... and *t* all hold, then also *u* holds Rules | If .... then
Fact / Unit Clause	u	$\top \Rightarrow u$	assume that *u* holds
Goal clause	$(\neg p \vee \neg q \vee \ldots \vee \neg t)$	$(p \wedge q \wedge \ldots \wedge t) \Rightarrow \perp$	show that *p* and *q* and ... and *t* all hold

$$(\neg p \vee \neg q \vee \ldots \vee \neg t \vee u) \equiv \neg(p \wedge \neg q \wedge \ldots \wedge \neg t) \vee u$$

Because (Implication elimination reversed) $\neg a \vee b \equiv a \Rightarrow b$:

$$\neg(p \wedge \neg q \wedge \ldots \wedge \neg t) \vee u \equiv (p \wedge \neg q \wedge \ldots \wedge \neg t) \Rightarrow u$$

Also: $(\neg p \vee \neg q \vee \ldots \vee \neg t \vee u) \equiv (\text{head/consequence} \vee \text{body/premise})$

Definite Clause and Modus Ponens

Modus Ponens

$P \Rightarrow Q$

Q

$\therefore\ Q$

Modus Ponens (textbook)

$$\frac{(P \Rightarrow Q),\ (Q)}{(Q)}$$

Definite Clause and Modus Ponens

Modus Ponens

$(p \wedge \neg q \wedge \ldots \wedge \neg t) \Rightarrow u$

u

$\therefore\ u$

Modus Ponens (textbook)

$$\frac{((p \wedge \neg q \wedge \ldots \wedge \neg t) \Rightarrow u),\ (u)}{(u)}$$

Forward Chaining Algorithm

Entailment can be verified with Forward Chaining:

- **set up your Knowledge Base** KB
- **set up your query** Q
- **start with known <u>facts</u> (say** A **and** B**):**
  - A **and** B **are automatically considered "inferred"**
  - **are they a part of some implication** $\mathrm{A} \wedge \mathrm{B} \Rightarrow \mathrm{X}$**?**
  - **if yes,** X **is now considered "inferred"**
- **Repeat until:**
  - Q **is "inferred", or**
  - **no further inferences can be made**

Forward Chaining: Pseudocode

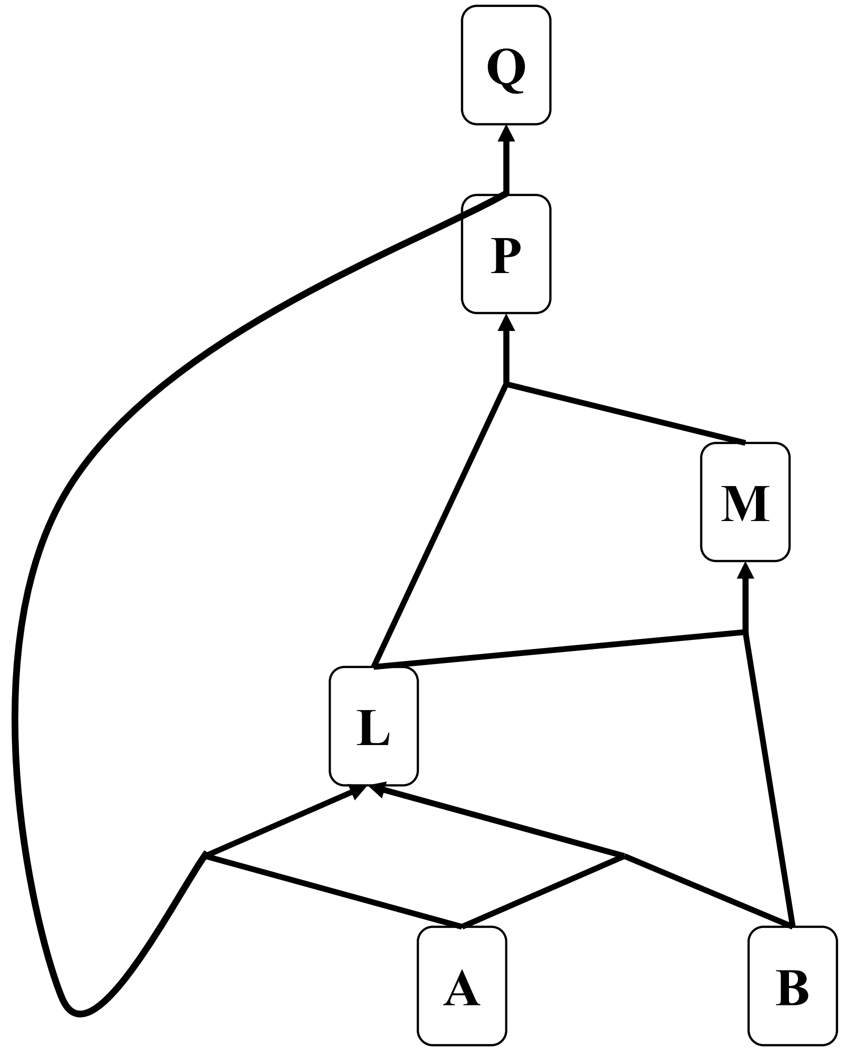
```
function PL-FC-ENTAILS?(KB, q) returns true or false
  inputs: KB, the knowledge base, a set of propositional definite clauses
          q, the query, a proposition symbol
  count ← a table, where count[c] is initially the number of symbols in clause c's premise
  inferred ← a table, where inferred[s] is initially false for all symbols
  queue ← a queue of symbols, initially symbols known to be true in KB

  while queue is not empty do
     p ← POP(queue)
     if p = q then return true
     if inferred[p] = false then
        inferred[p] ← true
        for each clause c in KB where p is in c.PREMISE do
           decrement count[c]
           if count[c] = 0 then add c.CONCLUSION to queue
  return false
```

Forward Chaining Algorithm

Knowledge Base KB:

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

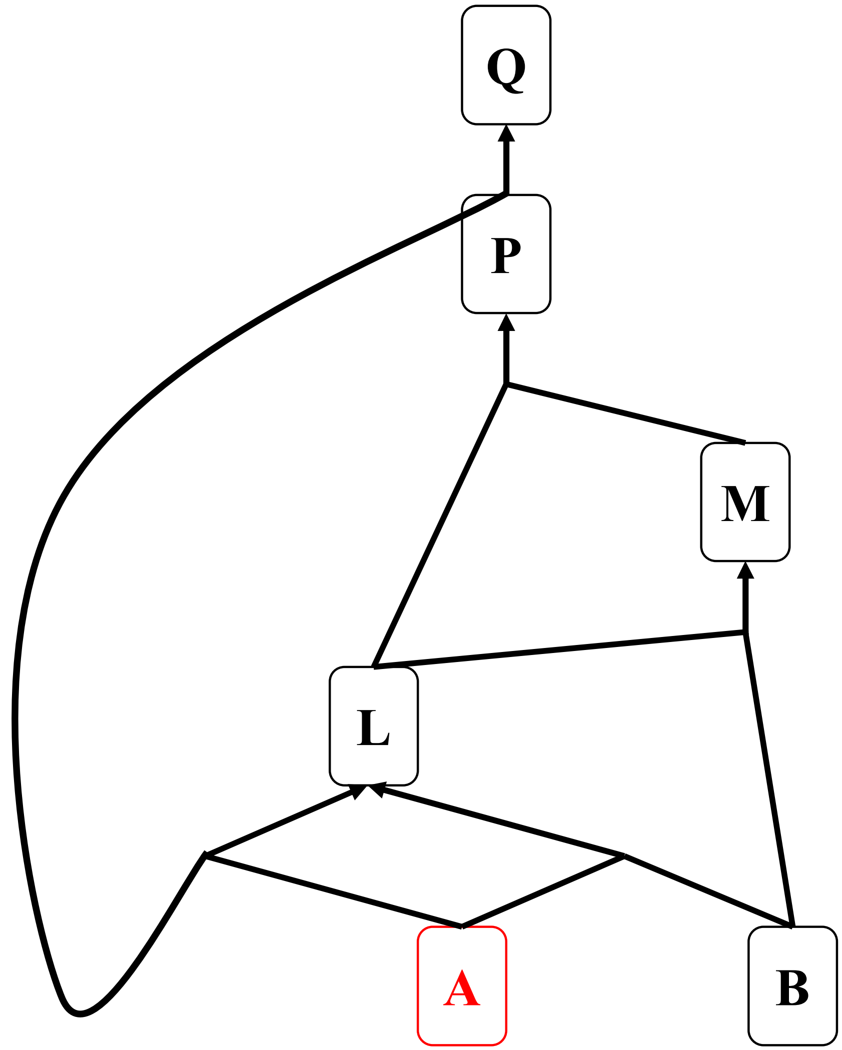
Inferred



Forward Chaining Algorithm

Knowledge Base KB:

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B



Inferred

A (because it is a fact)

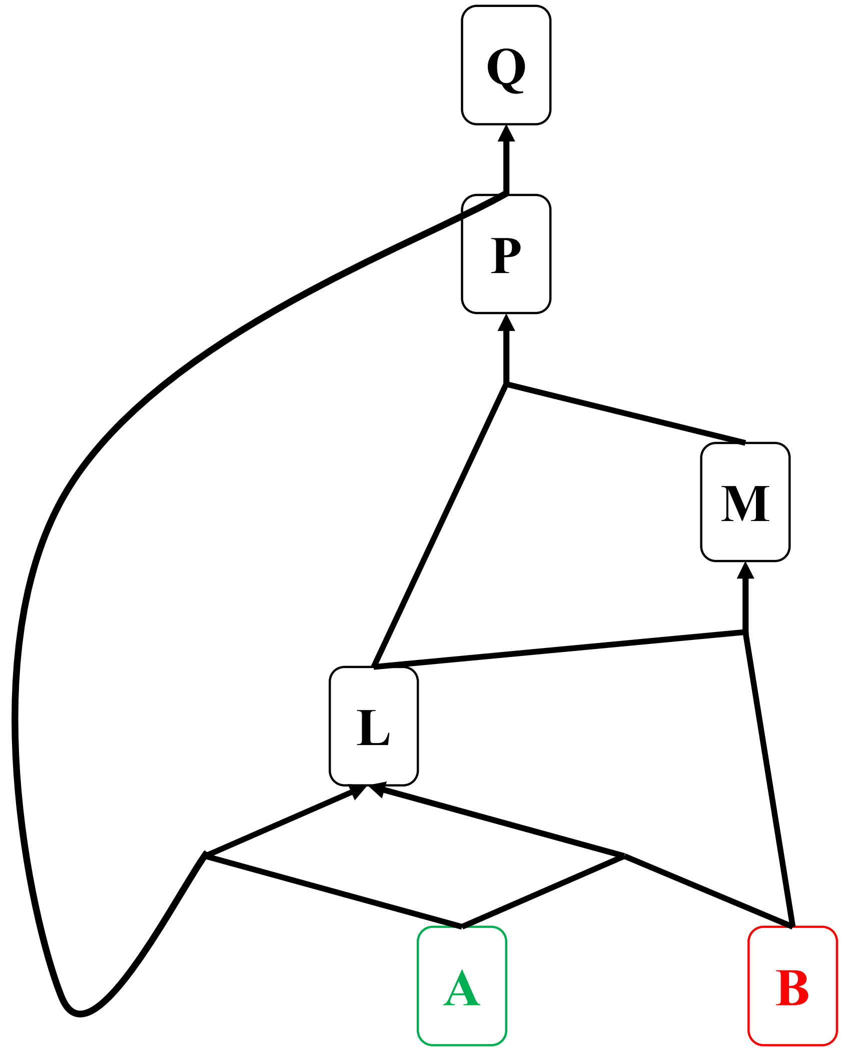
Forward Chaining Algorithm

Knowledge Base KB:

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Inferred:

A
B (because it is a fact)



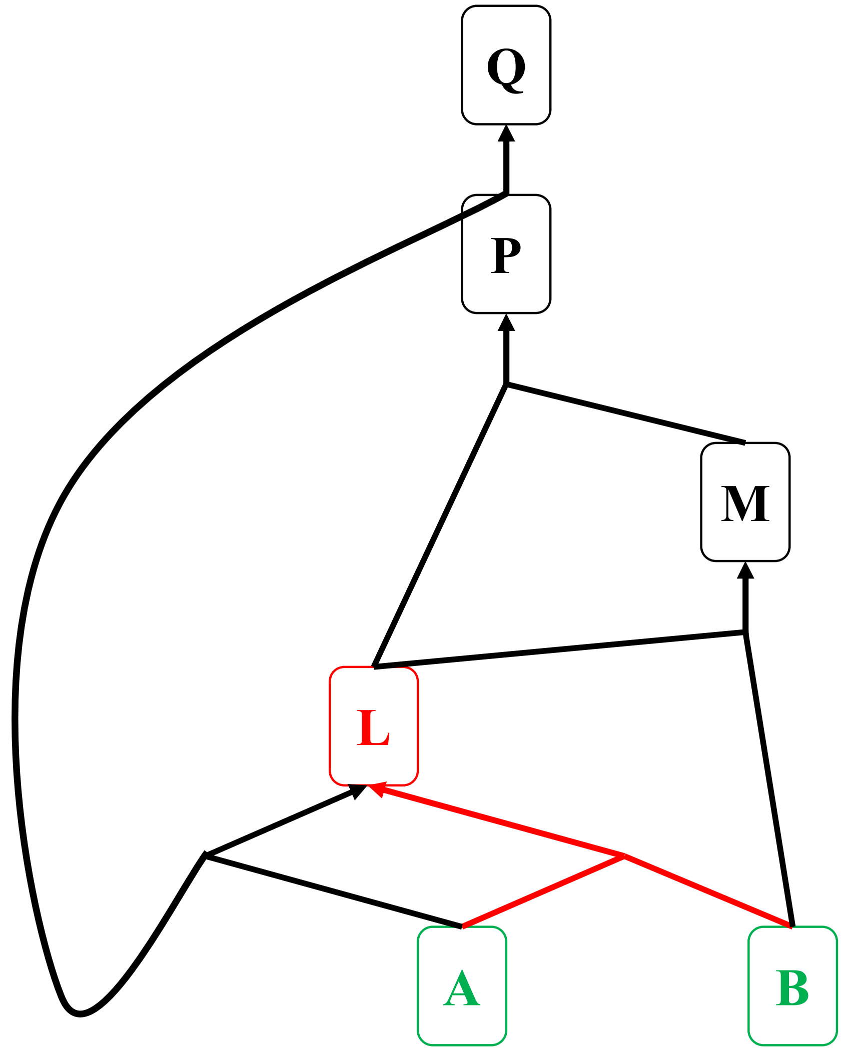
Forward Chaining Algorithm

Knowledge Base KB:

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Inferred:

A
B
L (because $A \wedge B \Rightarrow L$)



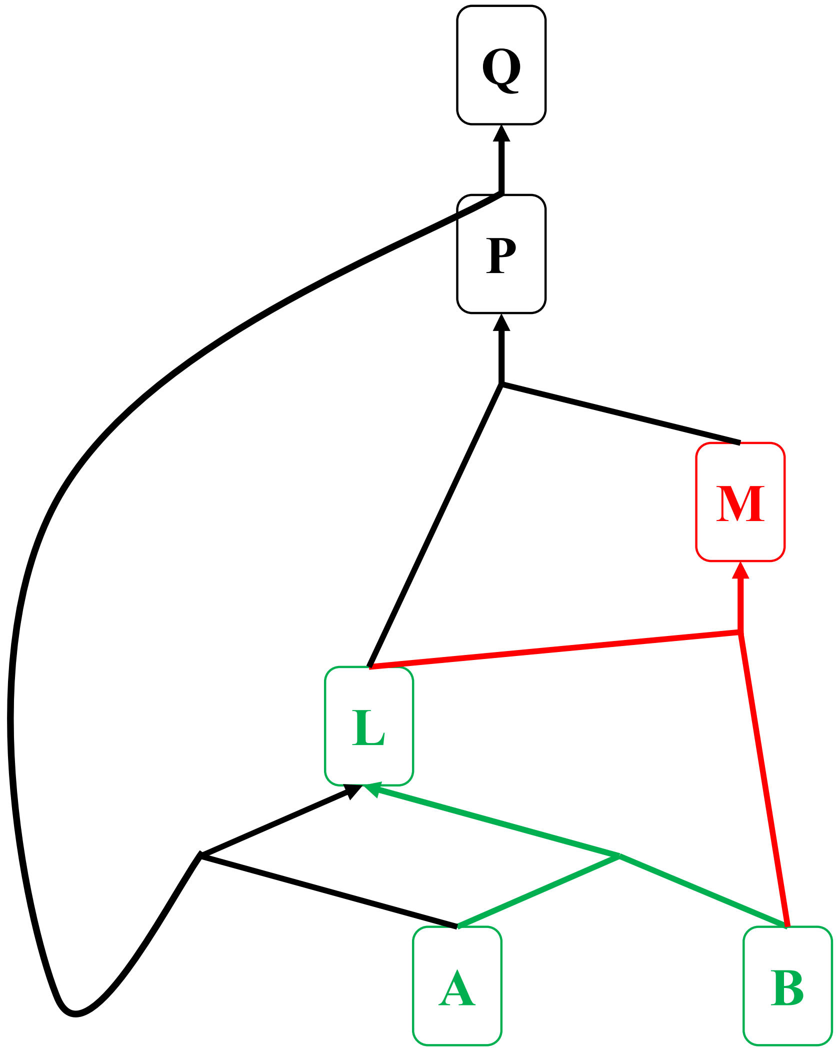
Forward Chaining Algorithm

Knowledge Base KB:

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Inferred:

A
B
L (because $A \wedge B \Rightarrow L$)
M (because $B \wedge L \Rightarrow M$)



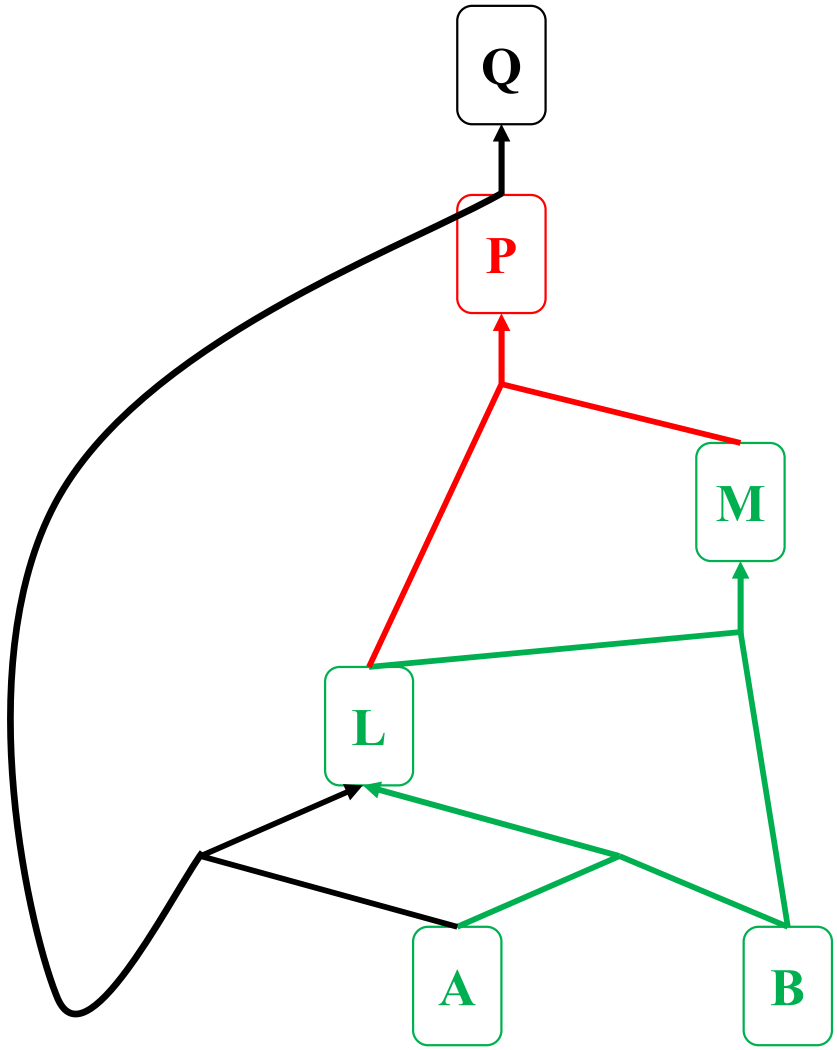
Forward Chaining Algorithm

Knowledge Base KB:

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Inferred:

A
B
L (because $A \wedge B \Rightarrow L$)
M (because $B \wedge L \Rightarrow M$)
P (because $L \wedge M \Rightarrow P$)



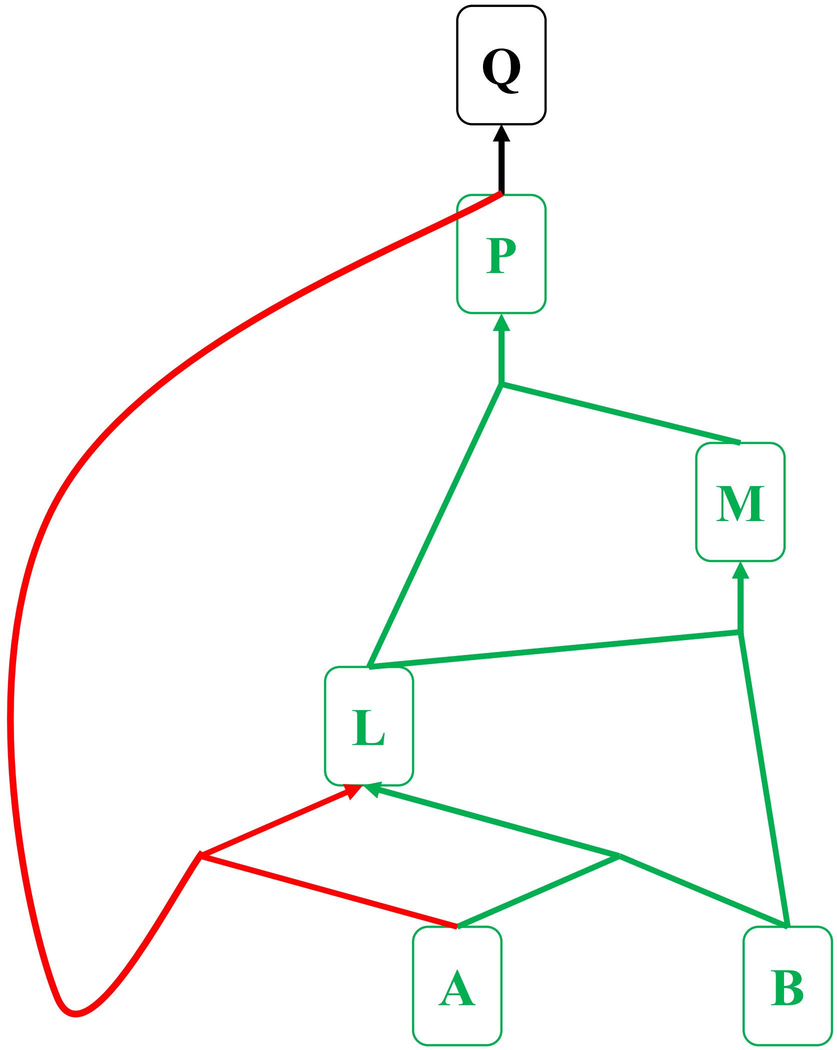
Forward Chaining Algorithm

Knowledge Base KB:

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$ (note: L is already inferred)
$A \wedge B \Rightarrow L$
A
B

Inferred:

A
B
L (because $A \wedge B \Rightarrow L$)
M (because $B \wedge L \Rightarrow M$)
P (because $L \wedge M \Rightarrow P$)



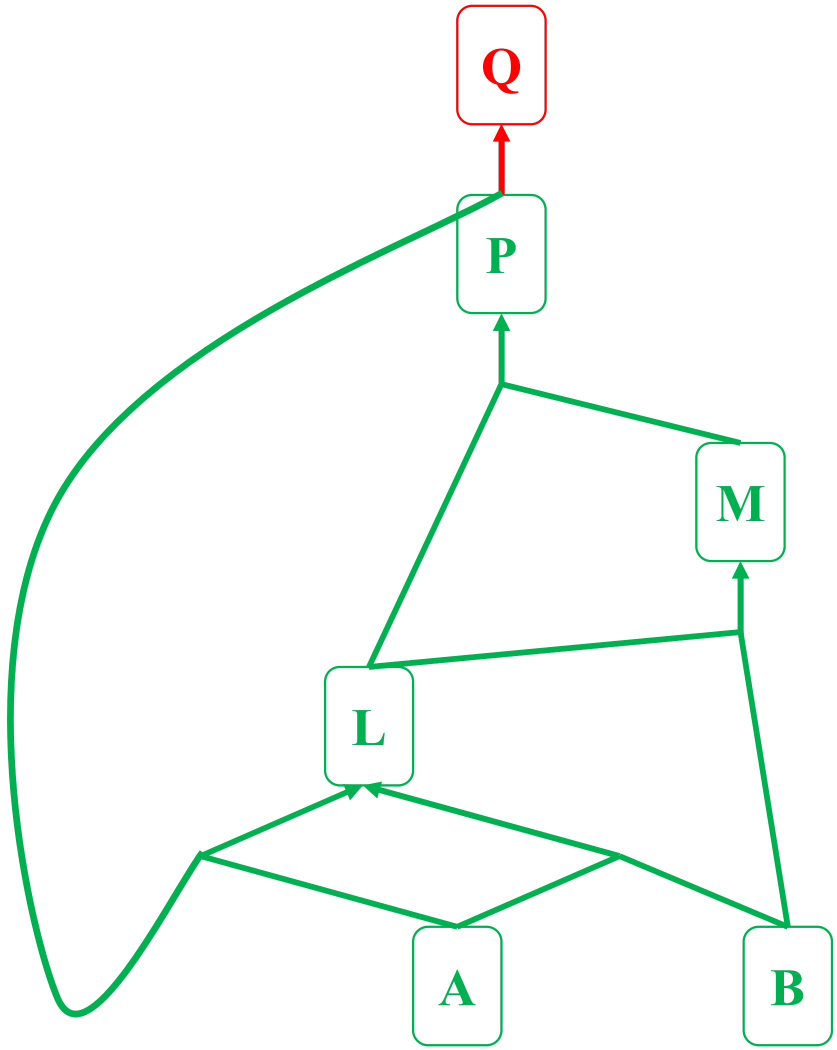
Forward Chaining Algorithm

Knowledge Base KB:

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Inferred:

A
B
L (because $A \wedge B \Rightarrow L$)
M (because $B \wedge L \Rightarrow M$)
P (because $L \wedge M \Rightarrow P$)
Q (because $P \Rightarrow Q$)



Forward Chaining Algorithm

Knowledge Base KB:

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
A
B

Inferred:

A
B
L (because $A \wedge B \Rightarrow L$)
M (because $B \wedge L \Rightarrow M$)
P (because $L \wedge M \Rightarrow P$)
Q (because $P \Rightarrow Q$)
Q is inferred, therefore <u>KB entails Q</u>