

CS 480

Introduction to Artificial Intelligence

February 22nd, 2022

Announcements / Reminders

- **Midterm: Thursday, February 24th!**
- **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**

Propositional Logic and KB-Agents

**Propositional
Logic:
Syntax**

**Propositional
Logic:
Semantics**

**Propositional
Logic:
Inference and
Proof Systems**

**KB-Agents:
Inference
algorithms**

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**”

Model / “Possible World”

A **model** (a “possible world”) is a single truth assignment / interpretation.

If a sentence U is **true** in model K , K satisfies U .

$M(U)$: set of **ALL** models of U (that satisfy U)

Now:

$KB \models Q$ if and only if $M(KB) \subseteq M(Q)$

$KB \models Q$ is **true** if and only if **in EVERY model** in which KB is **true**, Q is also **true**.

Logical Entailment with Truth Table

$$p \Rightarrow q \quad \text{KB}$$

$$p \Rightarrow \neg r$$

$$\neg p \Rightarrow \neg r$$

$$\text{KB} \Leftrightarrow (p \Rightarrow q) \wedge (p \Rightarrow \neg r) \wedge (\neg p \Rightarrow \neg r)$$

$$Q \Leftrightarrow \neg r$$

$$\therefore \neg r \quad Q$$

Model	p	q	r	P1:p \Rightarrow q	P2:q \Rightarrow \neg r	P3: \neg p \Rightarrow \neg r	KB	Q
M1	true	true	true	true	false	true	false	false
M2	true	true	false	true	true	true	true	true
M3	true	false	true	false	true	true	false	false
M4	true	false	false	false	true	true	false	true
M5	false	true	true	true	false	false	false	false
M6	false	true	false	true	true	true	true	true
M7	false	false	true	true	true	false	false	false
M8	false	false	false	true	true	true	true	true

Entailment: Model Checking

$$p \Rightarrow q \quad \text{KB}$$

$$p \Rightarrow \neg r$$

$$\neg p \Rightarrow \neg r$$

$$\therefore \neg r \quad Q$$

$$\text{KB} \Leftrightarrow (p \Rightarrow q) \wedge (p \Rightarrow \neg r) \wedge (\neg p \Rightarrow \neg r) \mid Q \Leftrightarrow \neg r$$

Models where KB is true: $M(\text{KB}) = \{M2, M6, M8\}$

Models where Q is true: $M(Q) = \{M2, M4, M6, M8\}$

Model	p	q	r	P1: $p \Rightarrow q$	P2: $q \Rightarrow \neg r$	P3: $\neg p \Rightarrow \neg r$	KB	Q
M1	true	true	true	true	false	true	false	false
M2	true	true	false	true	true	true	true	true
M3	true	false	true	false	true	true	false	false
M4	true	false	false	false	true	true	false	true
M5	false	true	true	true	false	false	false	false
M6	false	true	false	true	true	true	true	true
M7	false	false	true	true	true	false	false	false
M8	false	false	false	true	true	true	true	true

Entailment: Model Checking

$$p \Rightarrow q \quad \text{KB}$$

$$p \Rightarrow \neg r$$

$$\neg p \Rightarrow \neg r$$

$$\therefore \neg r \quad Q$$

$$\text{KB} \Leftrightarrow (p \Rightarrow q) \wedge (p \Rightarrow \neg r) \wedge (\neg p \Rightarrow \neg r) \mid Q \Leftrightarrow \neg r$$

Models where KB is true: $M(\text{KB}) = \{M2, M6, M8\}$

Models where Q is true: $M(Q) = \{M2, M4, M6, M8\}$

$M(\text{KB}) \subseteq M(Q)$ so Q follows KB

Model	p	q	r	P1: $p \Rightarrow q$	P2: $q \Rightarrow \neg r$	P3: $\neg p \Rightarrow \neg r$	KB	Q
M1	true	true	true	true	false	true	false	false
M2	true	true	false	true	true	true	true	true
M3	true	false	true	false	true	true	false	false
M4	true	false	false	false	true	true	false	true
M5	false	true	true	true	false	false	false	false
M6	false	true	false	true	true	true	true	true
M7	false	false	true	true	true	false	false	false
M8	false	false	false	true	true	true	true	true

Logical Entailment

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

$$\text{KB} \models \text{Q}$$

One more way to look at it:

If **KB** entails **Q**,

- “the truth of **KB** guarantees truth of **Q**”
- “the falsity of **KB** guarantees falsity of **Q**”

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:

$$KB \models Q$$

KB	Q	$KB \models Q$
true	true	true
true	false	false
false	true	false
false	false	false

Entailment: Deriving Conclusions

You can prove that:

$$KB \models Q$$

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

Entailment Proofs

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

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

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

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

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

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

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

Entailment Proofs

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

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

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

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

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

Entailment Proofs

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

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)

$KB \wedge \neg Q$ 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 as a Search Problem

Model checking can be considered a search problem. Searching a truth table for models in which **KB** entails **Q** (**Q** follows from **KB**). It is a $O(2^N)$ problem.

N Propositional Variables							KB \models Q	2 ^N Interpretations
p ₁	p ₂	p ₃	...	p _{N-1}	p _N			
true	true	true	...	true	true	false		
true	true	true	...	true	false	true		
true	true	false	...	false	true	false		
...		
false	false	true	...	true	false	true		
false	false	true	...	false	true	true		
false	false	false	...	false	false	false		

Programmers! What's the Difference?

& vs. && operator?

Programmers! What's the Difference?

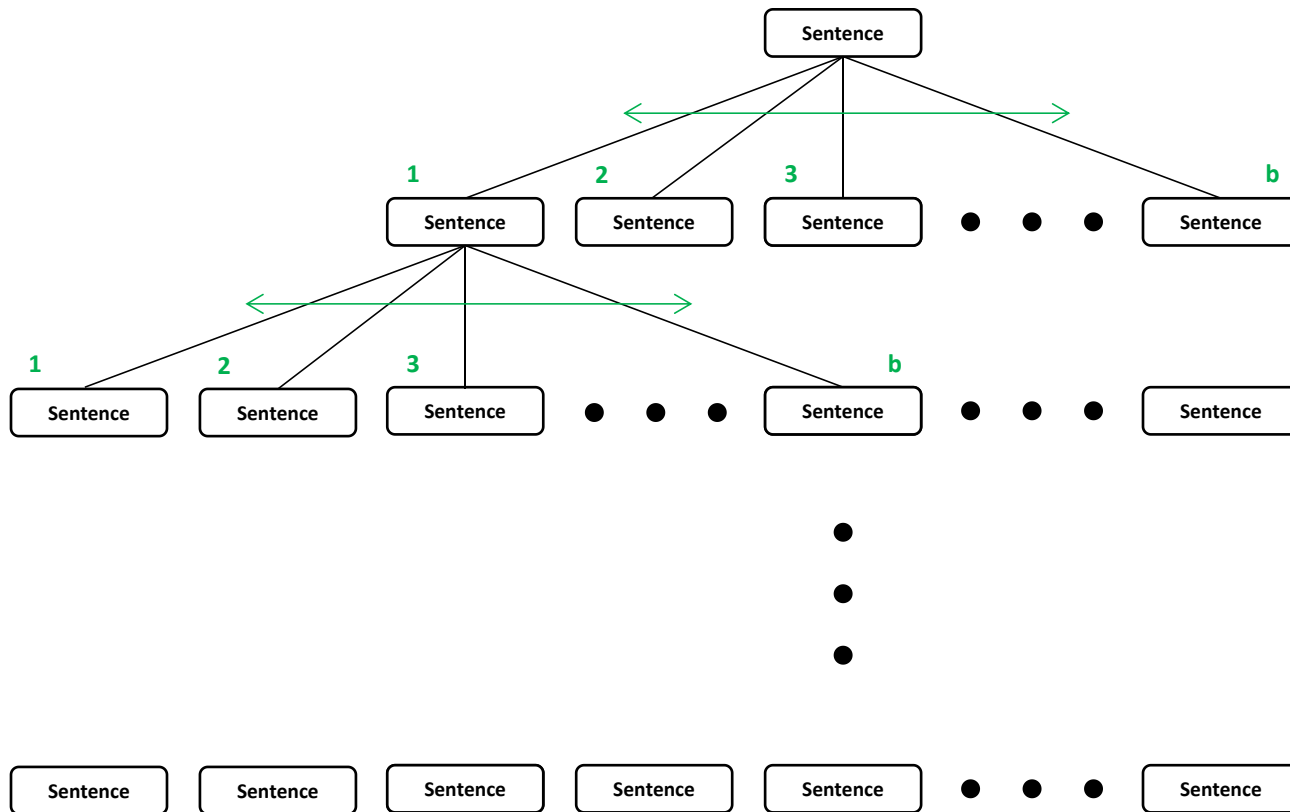
& vs. && operator?

What if I used?

$KB \equiv \text{PREMISE1} \ \&\& \ \text{PREMISE2} \ \&\& \ \dots \ \&\& \ \text{PREMISEN}$

What's the benefit?

Truth Table Enumeration as Search



Depth: 0
No assignment

Depth: 1
 p_1 : value assigned
partial assignment

Depth: 2
 p_2 : value assigned
partial assignment

Depth: **N**
 p_N : value assigned
complete assignment

Some truth assignments will quickly become **false**.
Not all propositional variables p_i need their values assigned to know that

$$C = \{ \dots \}$$


Truth Table Enumeration: Pseudocode

function TT-ENTAILS?(KB, α) **returns** *true* or *false*

inputs: KB , the knowledge base, a sentence in propositional logic
 α , the query, a sentence in propositional logic

$symbols \leftarrow$ a list of the proposition symbols in KB and α

return TT-CHECK-ALL($KB, \alpha, symbols, \{ \}$)

function TT-CHECK-ALL($KB, \alpha, symbols, model$) **returns** *true* or *false*

if EMPTY?($symbols$) **then**

if PL-TRUE?($KB, model$) **then return** PL-TRUE?($\alpha, model$)

else return *true* // when KB is false, always return true

else

$P \leftarrow$ FIRST($symbols$)

$rest \leftarrow$ REST($symbols$)

return (TT-CHECK-ALL($KB, \alpha, rest, model \cup \{P = true\}$)
 and
 TT-CHECK-ALL($KB, \alpha, rest, model \cup \{P = false\}$))

Returns true if EITHER “top” OR “bottom” recursive call returns true.

Evaluation

Evaluation is the process of **determining the truth values of compound/complex sentences** given a **truth assignment for the truth values of proposition constants/atomic sentences**. Consider the following truth assignment i:

$p^i = \text{true}, q^i = \text{false}, r^i = \text{true}$ Assignment

Let's evaluate the following complex sentence $(p \vee q) \wedge (\neg q \vee r)$:

$(p \vee q) \wedge (\neg q \vee r) \rightarrow (\text{true} \vee \text{false}) \wedge (\neg \text{false} \vee \text{true})$ Substitution

$(\text{true} \vee \text{false}) \wedge (\neg \text{false} \vee \text{true})$ Disjunction

$\text{true} \wedge (\neg \text{false} \vee \text{true})$ Negation

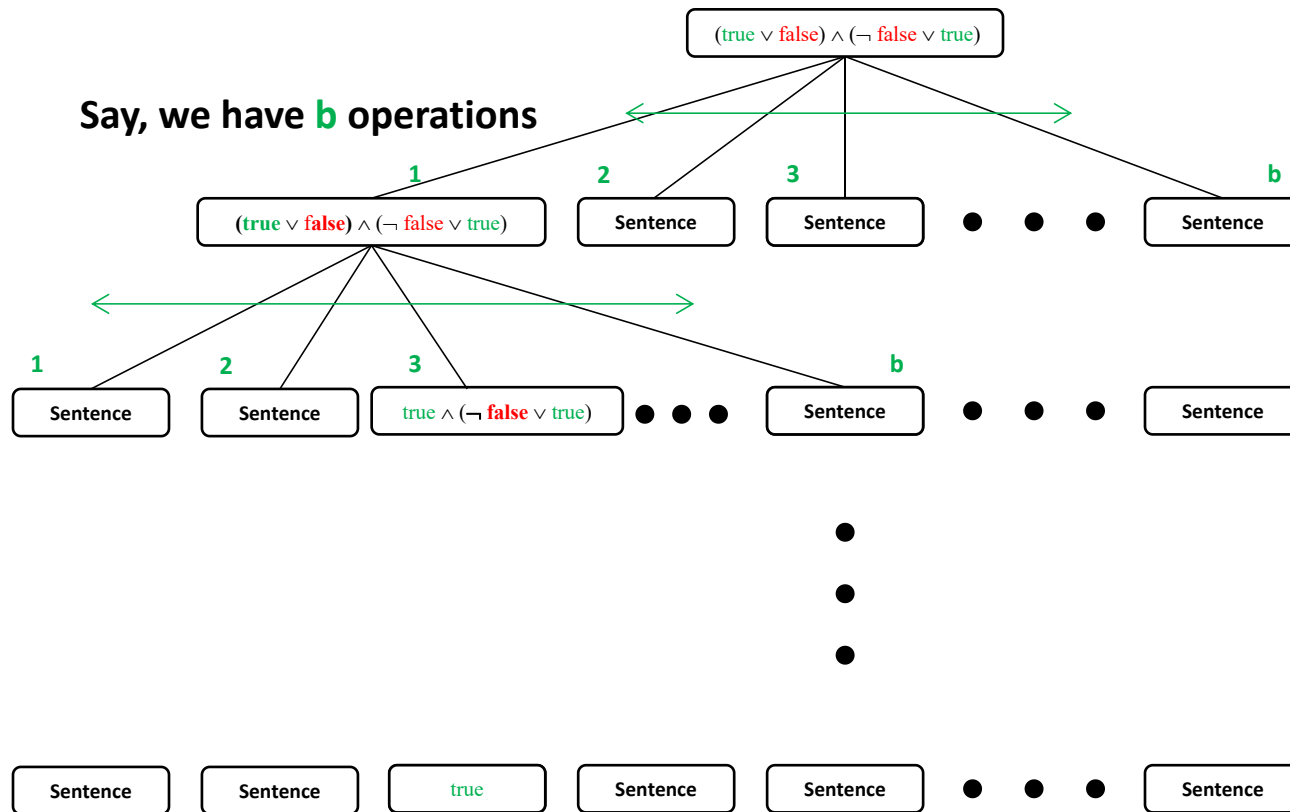
$\text{true} \wedge (\text{true} \vee \text{true})$ Disjunction

$\text{true} \wedge \text{true}$ Conjunction

true Interpretation

There is a path of operations that leads from substitution to the final interpretation.

Sentence Evaluation as Searching



Depth: 0
Substitution

Depth: 1
Disjunction

Depth: 2
Negation

Depth: **d**
Interpretation

Deduction / Proof

Laws/theorems in propositional logic can be used to prove additional theorems through a process known as **deduction**:

Prove that $((\neg m \vee n) \wedge \neg n) \Rightarrow \neg m$ **is a tautology:**

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

$$((\neg m \wedge \neg n) \vee \perp) \Rightarrow \neg m$$

$$(\neg m \wedge \neg n) \Rightarrow \neg m$$

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

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

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

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

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

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

$$T \vee n$$

$$n \vee T$$

$$T$$

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

by Negation law (contradiction) $p \wedge \neg p \Leftrightarrow \perp$

by Identity law $p \vee \perp \Leftrightarrow p$

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

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

by Double Negation law $\neg(\neg p) \Leftrightarrow p$

by Associative law $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$

by Commutative law $p \vee q \Leftrightarrow q \vee p$

by Associative law $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$

by Law of Excluded Middle $p \vee \neg p \Leftrightarrow T$

by Commutative law $p \vee q \Leftrightarrow q \vee p$

by Domination Law $p \vee T \Leftrightarrow T$

There is a path of operations to get from the beginning to the end



Deduction / Proof as Search

Laws/theorems in propositional logic can be used to prove additional theorems through a process known as **deduction**:

Prove that $((\neg m \vee n) \wedge \neg n) \Rightarrow \neg m$ **is a tautology:**

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

$$((\neg m \wedge \neg n) \vee \perp) \Rightarrow \neg m$$

$$(\neg m \wedge \neg n) \Rightarrow \neg m$$

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

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

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

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

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

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

$$T \vee n$$

$$n \vee T$$

$$T$$

How were the laws
chosen for each
step?

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

by Negation law (contradiction) $p \wedge \neg p \Leftrightarrow \perp$

by Identity law $p \vee \perp \Leftrightarrow p$

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

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

by Double Negation law $\neg(\neg p) \Leftrightarrow p$

by Associative law $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$

by Commutative law $p \vee q \Leftrightarrow q \vee p$

by Associative law $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$

by Law of Excluded Middle $p \vee \neg p \Leftrightarrow T$

by Commutative law $p \vee q \Leftrightarrow q \vee p$

by Domination Law $p \vee T \Leftrightarrow T$

There is a path of
operations to get
from the beginning
to the end



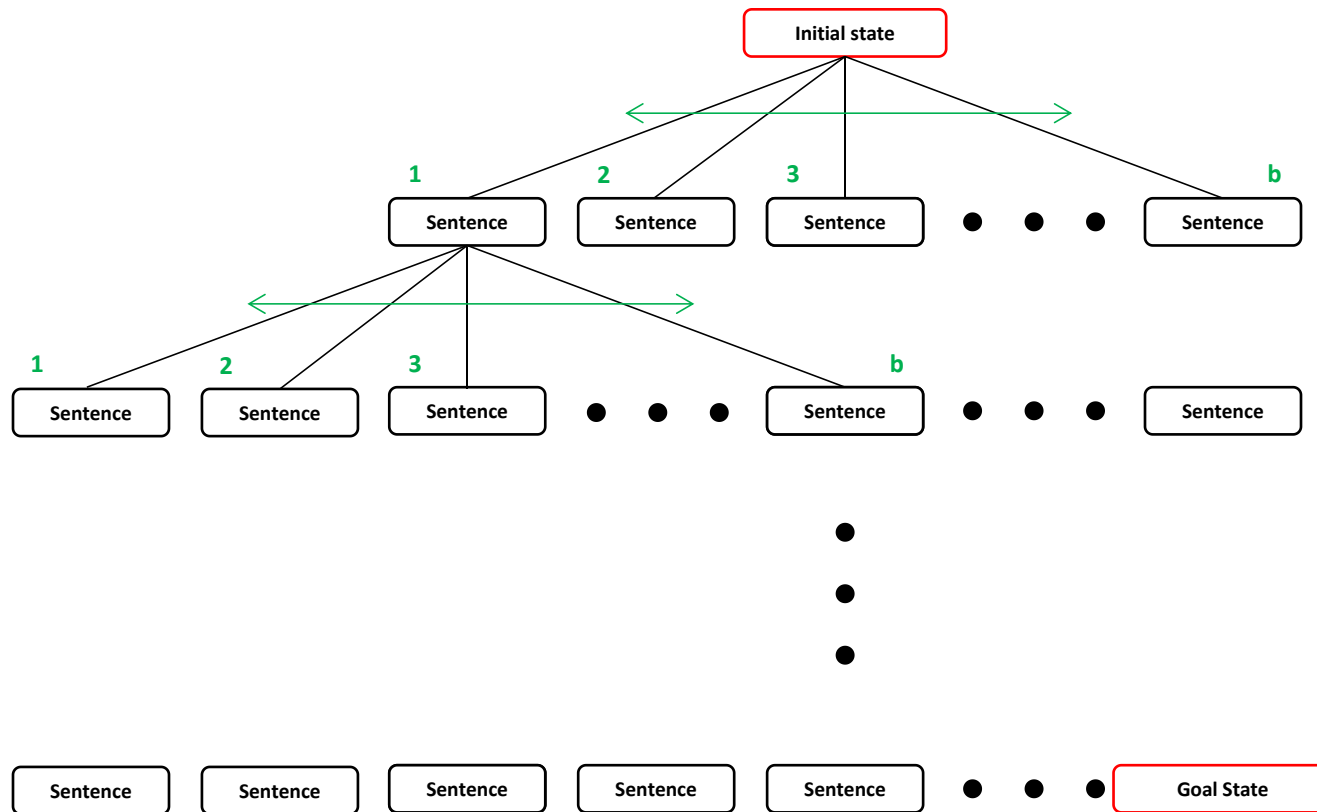
Proof as Search

Search algorithms can be used to find a sequence of steps that constitute a proof.

Just define the proof problem as a search problem:

- **INITIAL STATE:** initial knowledge base (sentence)
- **ACTIONS:** the set of all language rules
- **RESULT:** resulting sentence after applying a rule
- **GOAL:** a sentence that we are trying to prove

Deduction / Proof as Search



Depth: 0
Pick a rule/law

Depth: 1
Pick a rule/law

Depth: 2
Pick a rule/law

•
•
•

Depth: N
Pick a rule/law

Deduction / Proof

Laws/theorems in propositional logic can be used to prove additional theorems through a process known as **deduction**:

Prove that $((\neg m \vee n) \wedge \neg n) \Rightarrow \neg m$ is a tautology:

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

$$((\neg m \wedge \neg n) \vee \perp) \Rightarrow \neg m$$

$$(\neg m \wedge \neg n) \Rightarrow \neg m$$

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

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

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

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

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

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

$$T \vee n$$

$$n \vee T$$

$$T$$

Initial state

Goal state

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

by Negation law (contradiction) $p \wedge \neg p \Leftrightarrow \perp$

by Identity law $p \vee \perp \Leftrightarrow p$

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

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

by Double Negation law $\neg(\neg p) \Leftrightarrow p$

by Associative law $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$

by Commutative law $p \vee q \Leftrightarrow q \vee p$

by Associative law $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$

by Law of Excluded Middle $p \vee \neg p \Leftrightarrow T$

by Commutative law $p \vee q \Leftrightarrow q \vee p$

by Domination Law $p \vee T \Leftrightarrow T$

There is a path of operations to get from the beginning to the end

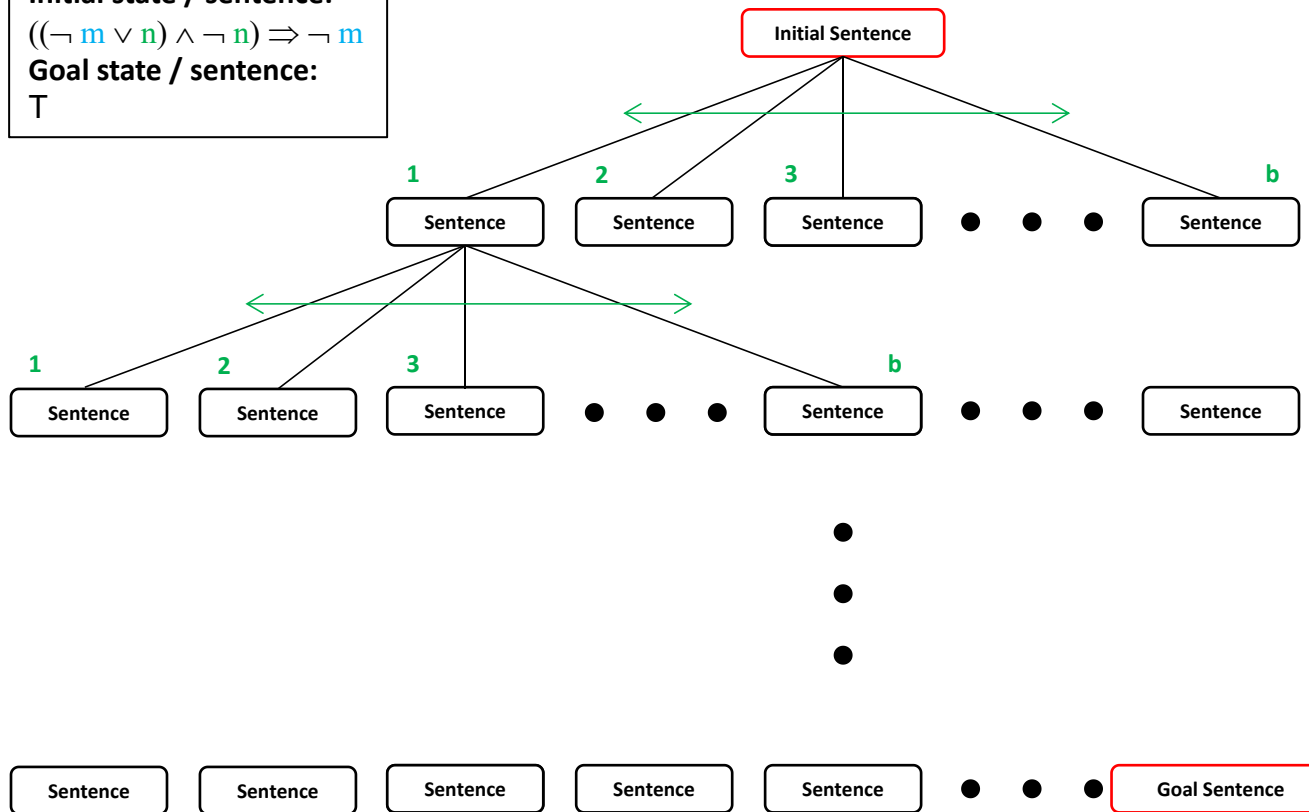
Deduction as Search

Initial state / sentence:

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

Goal state / sentence:

\top



Depth: 0
Pick a rule/law

Depth: 1
Pick a rule/law

Depth: 2
Pick a rule/law

Depth: N
Pick a rule/law

•
•
•

Model Checking: Q is Satisfiable

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

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

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

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

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

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

What Does It Mean?

Some queries Q can be proven to follow KB or not without interpreting KB and Q . For example:

- If Q is a tautology, Q will ALWAYS follow from KB no matter what KB is
- If Q is a contradiction, Q will NEVER follow from KB no matter what KB is

This can be decided at the syntax level through deduction.

Again: Tautology Proved by Deduction

Laws/theorems in propositional logic can be used to prove additional theorems through a process known as **deduction**:

Prove that $((\neg m \vee n) \wedge \neg n) \Rightarrow \neg m$ is a tautology:

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

$$((\neg m \wedge \neg n) \vee \perp) \Rightarrow \neg m$$

$$(\neg m \wedge \neg n) \Rightarrow \neg m$$

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

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

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

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

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

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

$$T \vee n$$

$$n \vee T$$

$$T$$

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

by Negation law (contradiction) $p \wedge \neg p \Leftrightarrow \perp$

by Identity law $p \vee \perp \Leftrightarrow p$

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

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

by Double Negation law $\neg(\neg p) \Leftrightarrow p$

by Associative law $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$

by Commutative law $p \vee q \Leftrightarrow q \vee p$

by Associative law $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$

by Law of Excluded Middle $p \vee \neg p \Leftrightarrow T$

by Commutative law $p \vee q \Leftrightarrow q \vee p$

by Domination Law $p \vee T \Leftrightarrow T$

What Does It Mean?

Some queries Q can be proven to follow KB or not without interpreting KB and Q . For example:

- If Q is a tautology, Q will ALWAYS follow from KB no matter what KB is
- If Q is a contradiction, Q will NEVER follow from KB no matter what KB is

This can be decided at the syntax level through deduction.

Proving Entailment: Two Levels

Syntax level



I was able to check entailment at
this level for a particular problem,
but will it always work?

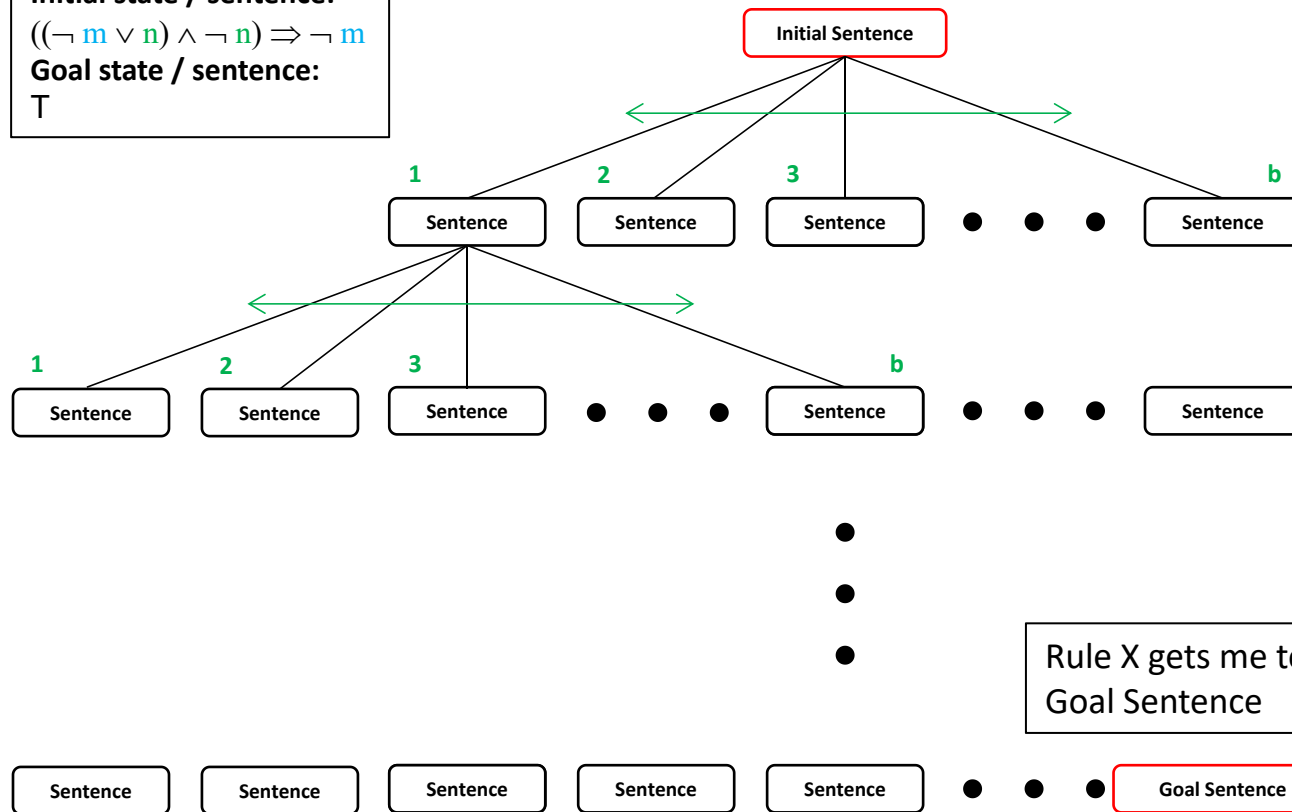
Deduction as Search

Initial state / sentence:

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

Goal state / sentence:

T



Depth: 0
Pick a rule/law

Depth: 1
Pick a rule/law

Depth: 2
Pick a rule/law

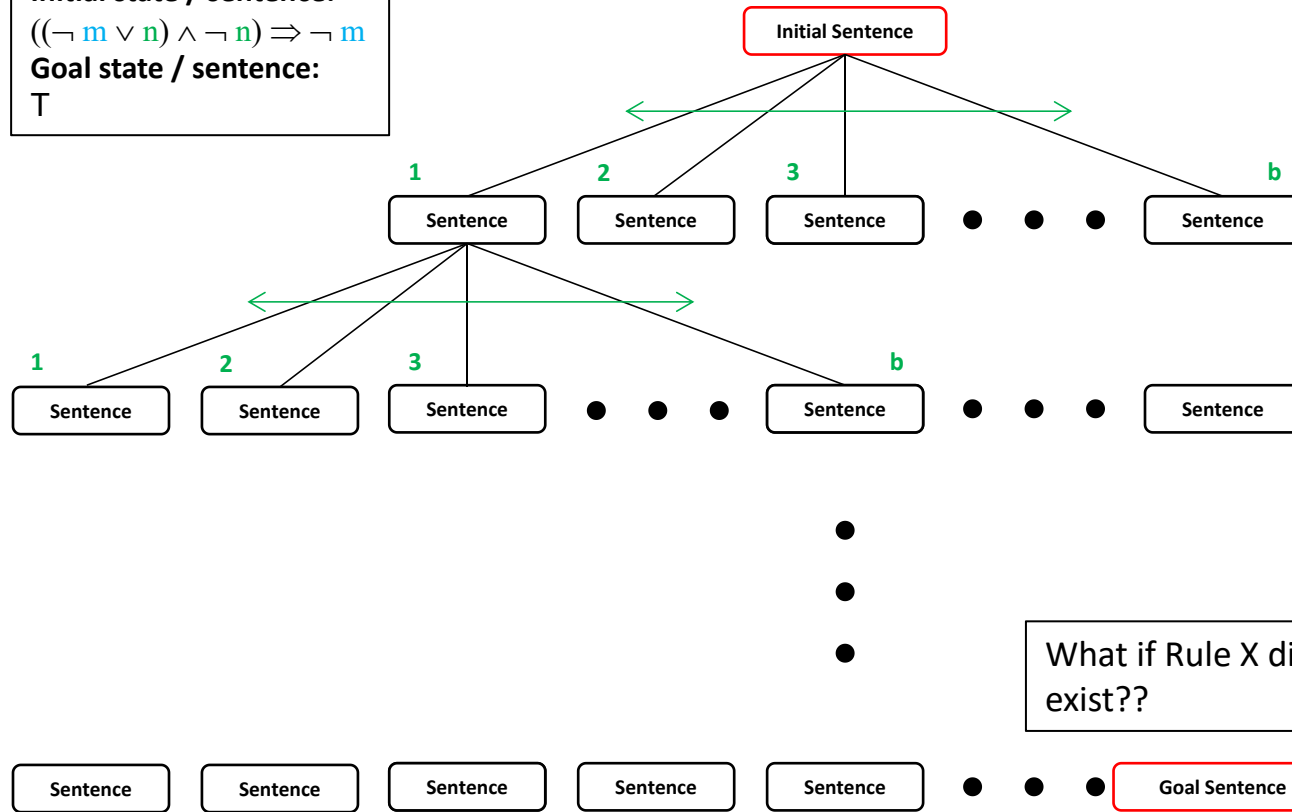
RULE X USED

Depth: N
Pick a rule/law

Deduction as Search

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

T



Depth: 0
Pick a rule/law

Depth: 1
Pick a rule/law

Depth: 2
Pick a rule/law

RULE X USED

Depth: N
Pick a rule/law

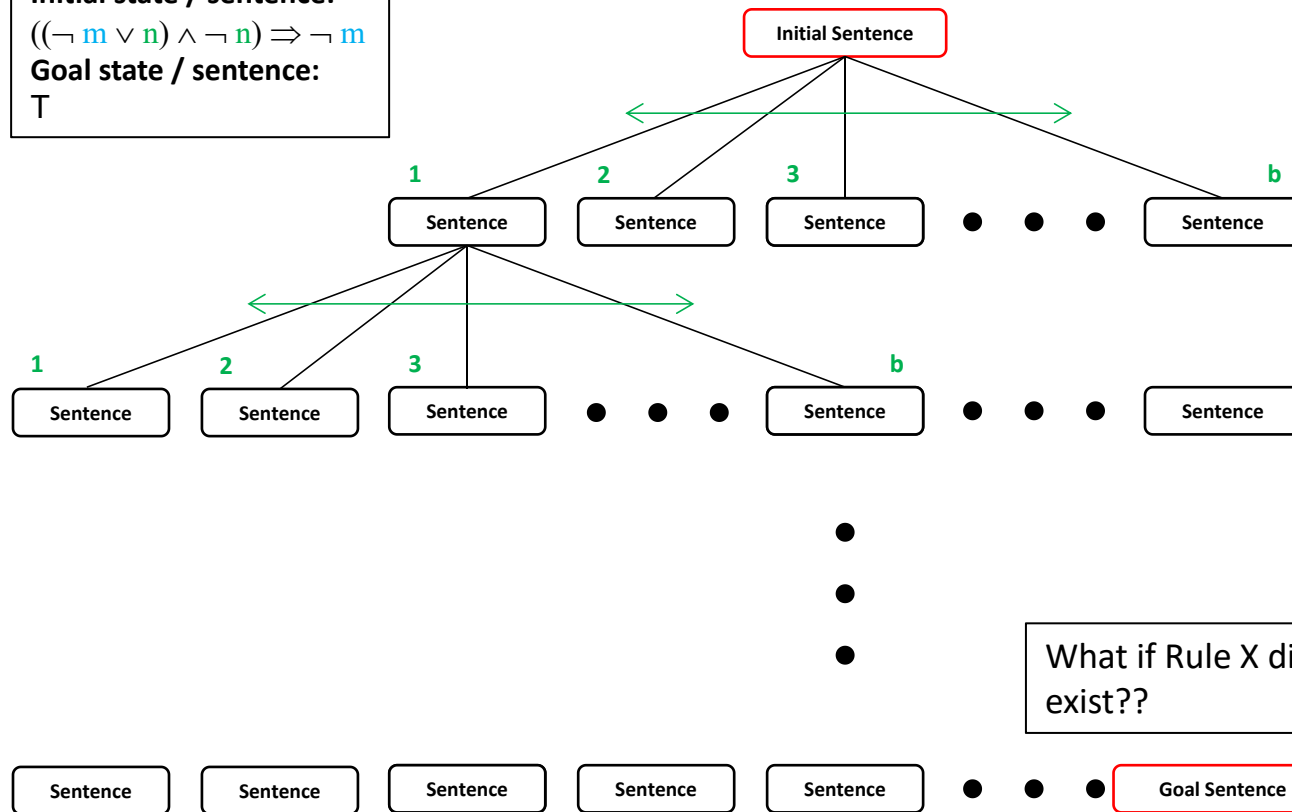
Deduction as Search

Initial state / sentence:

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

Goal state / sentence:

T



What if Rule X didn't exist??

Then this state is possible BUT UNREACHABLE!!

Depth: 0
Pick a rule/law

Depth: 1
Pick a rule/law

Depth: 2
Pick a rule/law

RULE X USED

Depth: N
Pick a rule/law

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

Completeness:

A complete calculus ALWAYS find 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**.

Entailment: Two Levels

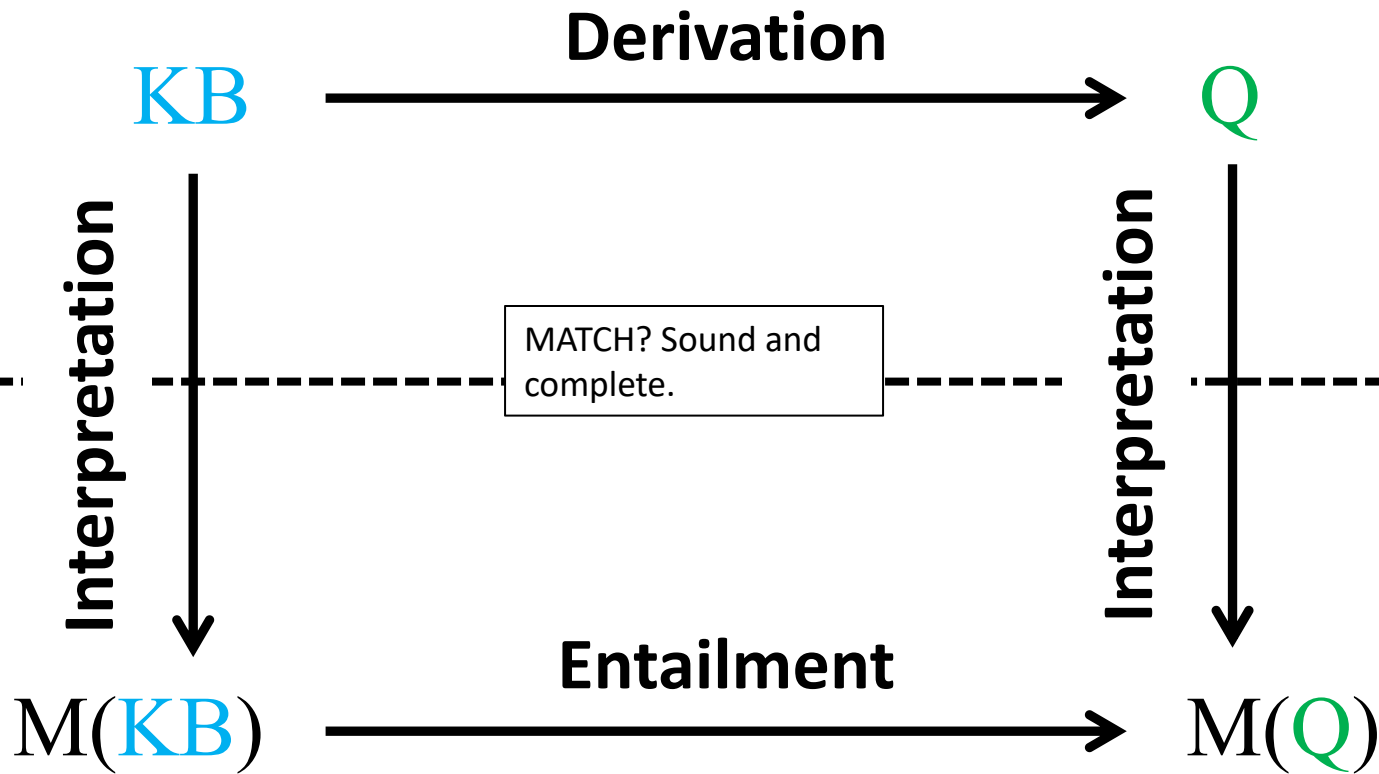
Syntax level



Semantic level

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 $P \Rightarrow Q$ P <hr/> $\therefore Q$	Modus Tollens $P \Rightarrow Q$ $\neg Q$ <hr/> $\therefore \neg P$	Hypothetical Syllogism (Transitivity) $P \Rightarrow Q$ $Q \Rightarrow R$ <hr/> $\therefore P \Rightarrow R$	Conjunction P Q <hr/> $\therefore P \wedge Q$
Addition P <hr/> $\therefore P \vee Q$	Simplification $P \wedge Q$ <hr/> $\therefore P$	Disjunctive Syllogism $P \vee Q$ $\neg P$ <hr/> $\therefore Q$	Resolution $P \vee Q$ $\neg P \vee R$ <hr/> $\therefore Q \vee R$

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

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**):

$$\text{KB} \models \text{Q}$$

by proving that:

$$(\text{KB} \wedge \neg \text{Q}) \Leftrightarrow \perp$$

(show that $\text{KB} \wedge \neg \text{Q}$ is a **contradiction / empty clause**)

Resolution: Two Forms of Notation

Resolution

$$P \vee Q$$

$$\neg P \vee R$$

$$\therefore Q \vee R$$

Resolution (textbook)

$$(P \vee Q), (\neg P \vee R)$$

$$(Q \vee R)$$

Resolution: Two Forms of Notation

Resolution

$$P \vee Q$$

$$\neg P \vee R$$

$$\therefore Q \vee R$$

Resolution (textbook)

$$(P \vee Q), (\neg P \vee R)$$

$$(Q \vee R)$$

derived clause (resolvent)

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

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	$\leftrightarrow, \equiv, \text{iff}$	if and only if
\top	Tautology	$T, 1, \blacksquare$	truth
\perp	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 K_i consists of a **disjunction**

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

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 $\neg (p \wedge q) \Leftrightarrow \neg q \vee \neg p$

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

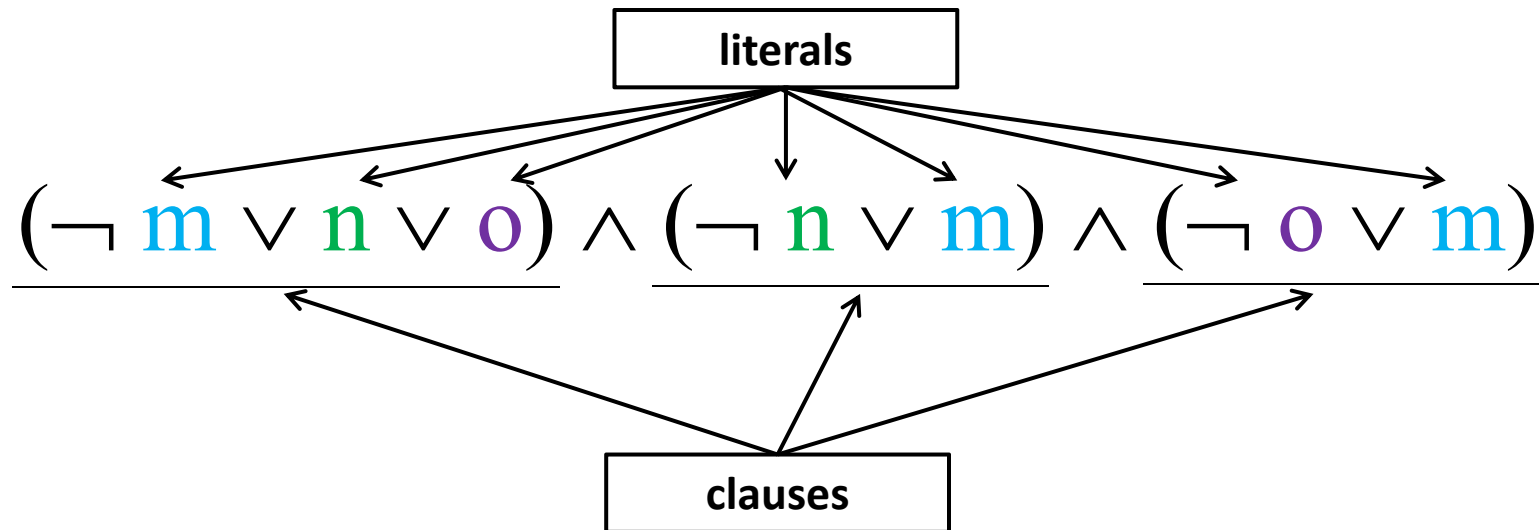
by Distributive law $p \vee (q \wedge r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$

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

Conjunctive Normal Form (CNF)

Example:

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



CNF Grammar

$CNFSentence \rightarrow Clause_1 \wedge \dots \wedge Clause_n$

$Clause \rightarrow Literal_1 \vee \dots \vee Literal_m$

$Fact \rightarrow Symbol$

$Literal \rightarrow Symbol \mid \neg Symbol$

$Symbol \rightarrow P \mid Q \mid R \mid \dots$

$HornClauseForm \rightarrow DefiniteClauseForm \mid GoalClauseForm$

$DefiniteClauseForm \rightarrow Fact \mid (Symbol_1 \wedge \dots \wedge Symbol_l) \Rightarrow Symbol$

$GoalClauseForm \rightarrow (Symbol_1 \wedge \dots \wedge Symbol_l) \Rightarrow False$

*** 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

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

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

Unit Resolution

General resolution rule allows clauses with arbitrary number of literals

The diagram illustrates the resolution rule. At the top, a box labeled "initial clauses" has two arrows pointing to the two clauses in the expression: $(a_1 \vee \dots \vee a_m \vee \mathbf{b})$ and $(\neg \mathbf{b} \vee c_1 \vee \dots \vee c_n)$. The literal \mathbf{b} in the first clause and $\neg \mathbf{b}$ in the second clause are highlighted in green. A horizontal line separates these from the result below. Below the line is the new clause: $(a_1 \vee \dots \vee a_m \vee c_1 \vee \dots \vee c_n)$. An arrow points from a box labeled "new clause" to this result.

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

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.

Unit Resolution

General resolution rule allows clauses with arbitrary number of literals

The diagram illustrates the resolution rule. At the top, a box labeled "initial clauses" has two arrows pointing to the two clauses in the expression: $(a_1 \vee \dots \vee a_m \vee \mathbf{b})$ and $(\neg \mathbf{b} \vee c_1 \vee \dots \vee c_n)$. The literal \mathbf{b} in the first clause and $\neg \mathbf{b}$ in the second clause are highlighted in green. A horizontal line separates these from the result below. Below the line is the derived clause: $(a_1 \vee \dots \vee a_m \vee c_1 \vee \dots \vee c_n)$. An arrow points from a box labeled "derived clause (resolvent)" to this result. The literal c_1 in the result is highlighted in green.

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

Literals \mathbf{b} and $\neg \mathbf{b}$ are **complimentary**. The clause $(\mathbf{b} \wedge \neg \mathbf{b})$ is a **contradiction** (an empty clause).

Unit Resolution

General resolution rule allows clauses with arbitrary number of literals

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

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

Occasionally, unit resolution will produce a new clause with the the following clause ($d \vee d$):

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

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

Resolution and Factorization

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

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

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:

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

So:

$$(a \Leftrightarrow \neg b) \equiv (a \Rightarrow \neg b) \wedge (\neg b \Rightarrow a) \quad \text{by Equivalence Law}$$

$$(a \Leftrightarrow \neg b) \equiv (\neg a \vee \neg b) \wedge (b \vee a) \quad \text{by Implication Law}$$

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

We obtain:

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

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:

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

Resolution applied to clauses 2 and 5

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

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

Produces new clause (9). 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)
9. $(b \vee \neg c)$

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 3 and 9

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

Produces new clause (10). 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)
9. $(b \vee \neg c)$
10. $(\neg c)$

Proof by Resolution: Example

Resolution steps:

Using resolution rule:

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

Resolution applied to clauses 8 and 10

$$(\neg c), (c)$$

\perp (empty clause)

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

Known clauses:

1. $(\neg a \vee \neg b)$
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)
9. $(b \vee \neg c)$
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**)?”

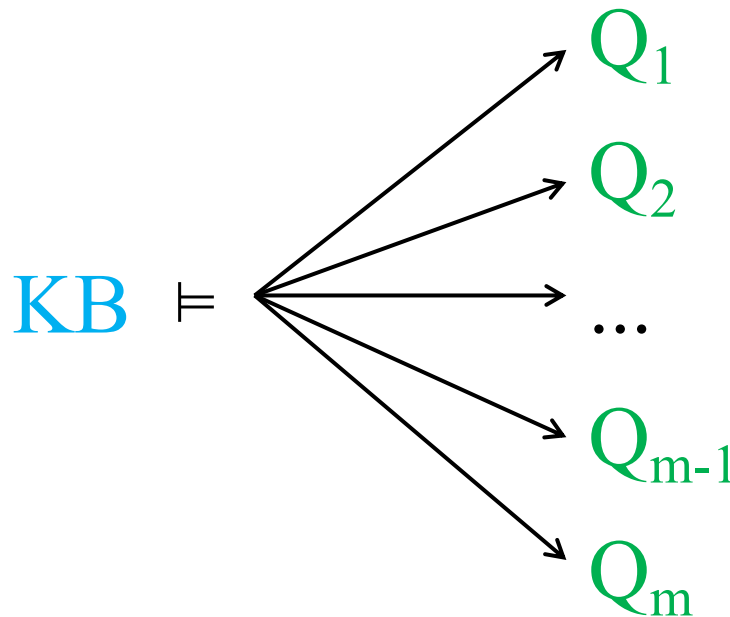
$$\text{KB} \models \text{Q}$$

But we could ask the following question:

“Which **Q**s follow from **KB**?”

Logical Entailment

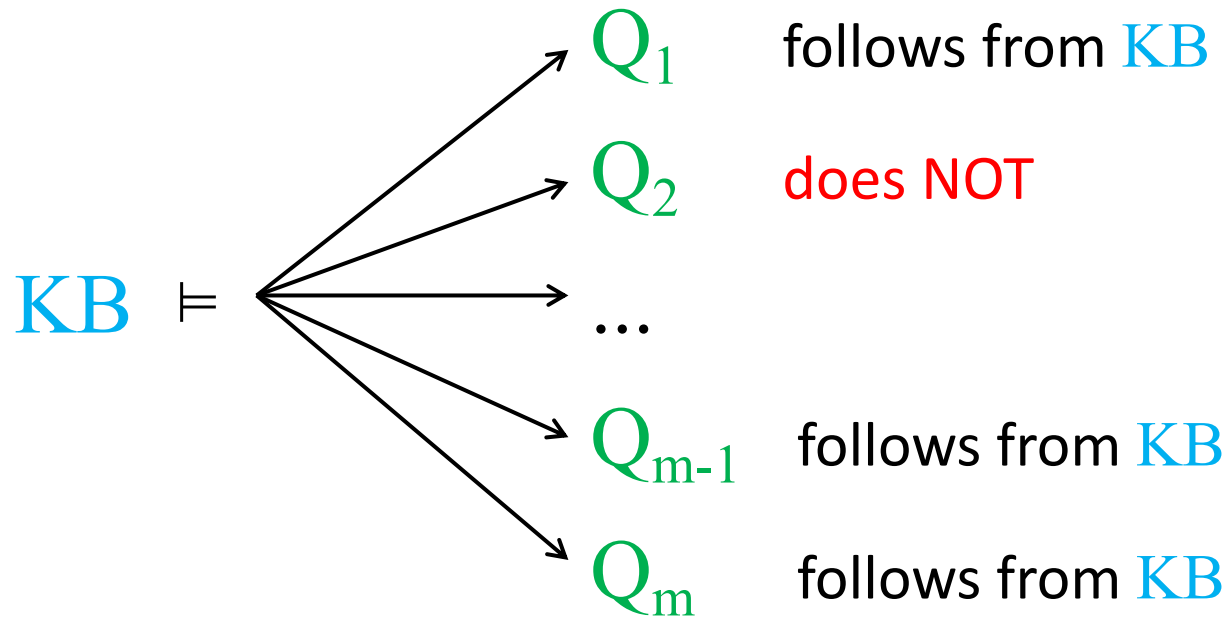
But we could ask the following question:
“Which Q s follow from KB ?”



Logical Entailment

But we could ask the following question:

“Which Q s follow 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 \leftarrow ASK(*KB*, MAKE-ACTION-QUERY(*t*))

TELL(*KB*, MAKE-ACTION-SENTENCE(*action*, *t*))

t \leftarrow *t* + 1

return *action*

CURRENTKB

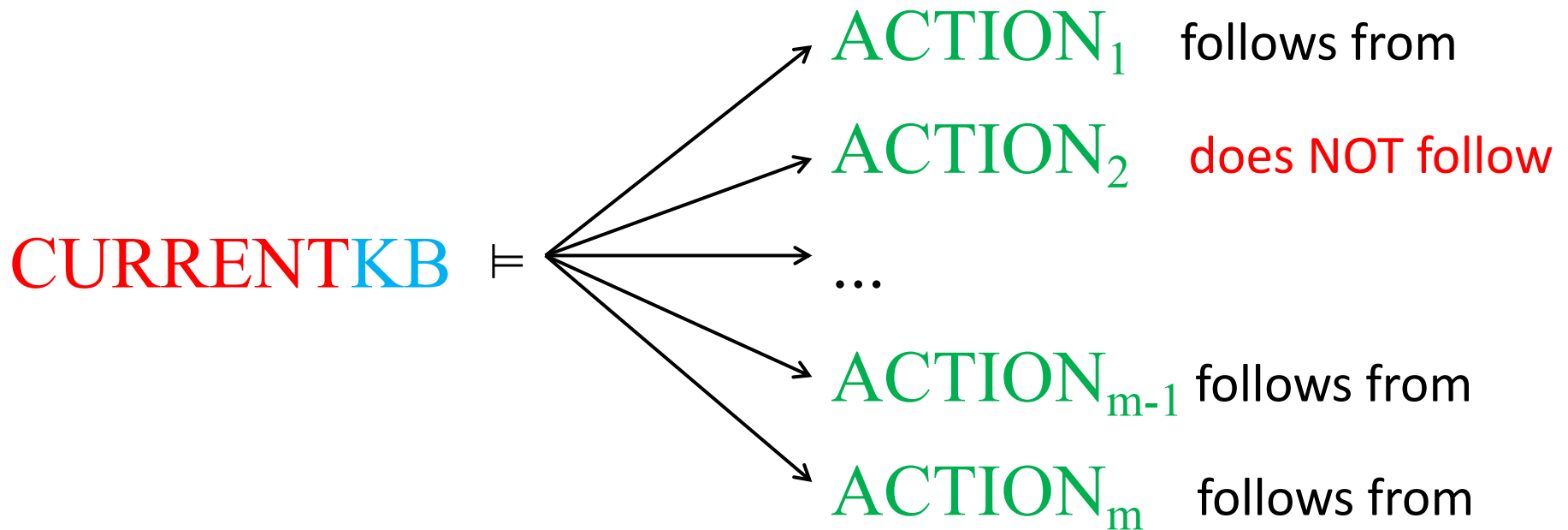
new *percept*

$\text{CURRENTKB} \Leftrightarrow \text{KBBEFORE} \wedge \text{percept}$

Logical Entailment with KB Agents

But we could ask the following question:

“Which **ACTION**s follow from **CURRENTKB**?”



Logical Entailment with KB Agents

But we could ask the following question:

“Which ACTIONs follow from CURRENTKB?”

