CS 480

Introduction to Artificial Intelligence

September 30th, 2021

Announcements / Reminders

- **Midterm: October 14th!**
  - **Online (NOT Beacon) section: please make arrangements. Contact Mr. Charles Scott (scott@iit.edu) if in doubt**
- **Programming Assignment #01:**
  - **due: October 17th, 11:00 PM CST**
- **Written Assignment #01: next week**
- **Please follow the Week 06 To Do List instructions**
- **Fall Semester midterm course evaluation reminder**
- **Grading TA assignment:**

  https://docs.google.com/spreadsheets/d/1Cav_GBTGC7fLGzxuBCAUmEuJYPeF-HMLCYvwPbq8Fus/edit?usp=sharing

Plan for Today

- **Propositional logic**

CORRECTION: Sentence Classes

SATISFIABLE

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

In plain English:
"You can find AT LEAST one assignment of logical values of true and false to individual propositional variables that will make this sentence true."

Example:

$p \Rightarrow q$

p	q	$p \Rightarrow q$
true	true	**true**
true	false	false
false	true	**true**
false	false	**true**

(LOGICALLY) VALID/TAUTOLOGY

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

In plain English:
"This sentence is ALWAYS true regardless of value assignment to individual propositional variables."

Example:

$p \vee \neg p$

p	$\neg p$	$p \wedge \neg p$
true	false	**true**
true	false	**true**
false	true	**true**
false	true	**true**

UNSATISFIABLE/CONTRADICTION

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

In plain English:
"This sentence is ALWAYS false regardless of value assignment to individual propositional variables."

Example:

$p \wedge \neg p$

p	$\neg p$	$p \wedge \neg p$
true	false	**false**
true	false	**false**
false	true	**false**
false	true	**false**

CORRECTION: Entailment: Model Checking

KB

$p \Rightarrow q$

$p \Rightarrow \neg r$

$\neg p \Rightarrow \neg r$

$\therefore \neg r$ Q

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

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

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

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

Model	p	q	r	P1:p⇒q	P2:q⇒¬r	P3:¬ p ⇒ ¬ 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

Propositional Logic and KB-Agents

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

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:

$$KB \models Q$$

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:

$$KB \vDash Q$$

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

Entailment Proofs

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

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

Prove that $(KB \land \neg Q) \Leftrightarrow \bot$ (Show that $KB \land \neg Q$ 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	$KB \Rightarrow Q$	$KB \land \neg Q$
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 $(KB \wedge \neg Q) \Leftrightarrow \bot$
(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

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

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

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

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

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

Prove that $(KB \wedge \neg Q) \Leftrightarrow \bot$
(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⇒q	q⇒¬r	¬p⇒¬r	KB	Q	KB ⇒ Q	KB ∧ ¬ 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.

2^N Possible Worlds (Models)

N Propositional Variables						
p_1	p_2	p_3	…	p_{N-1}	p_N	KB ⊨ Q
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

2^N Interpretations

Programmers! What's the Difference?

& vs. && operator?

Programmers! What’s the Difference?

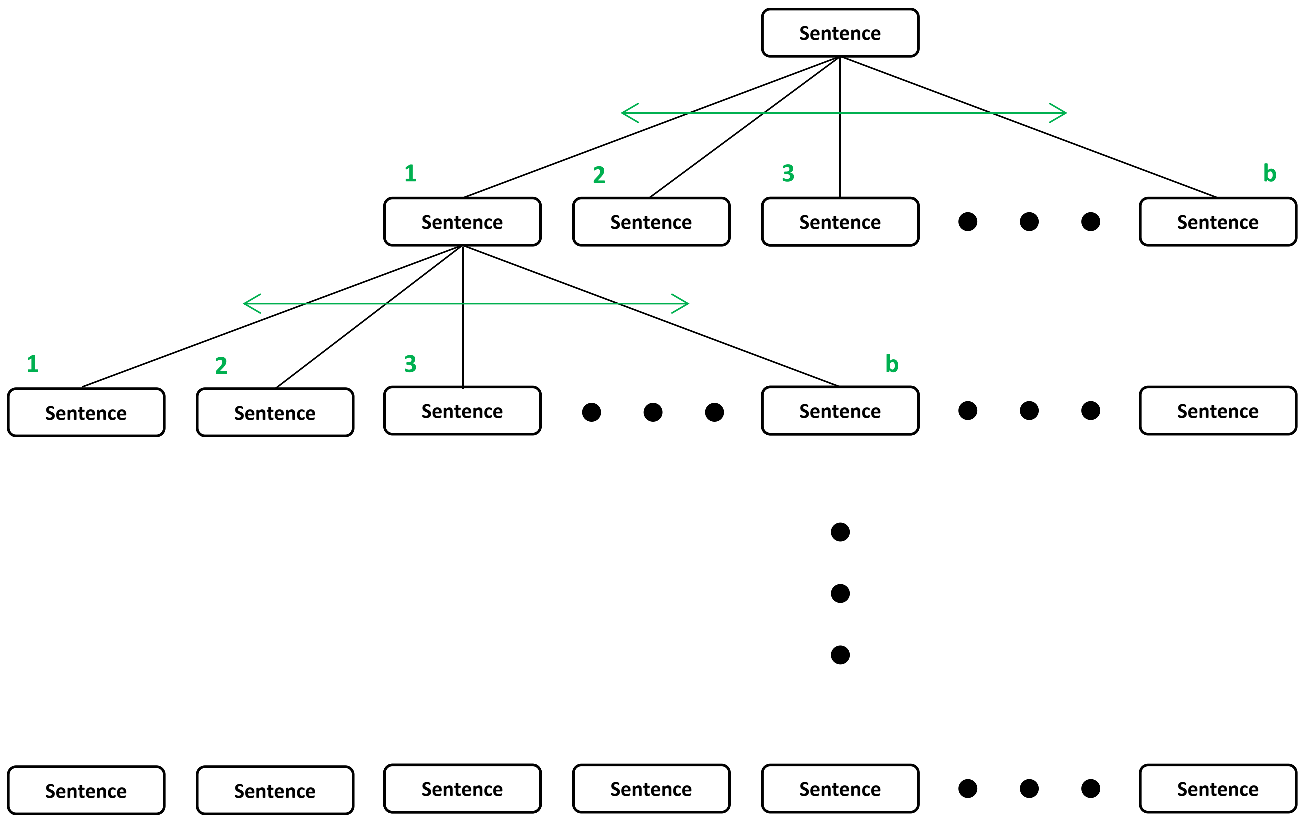
& vs. && operator?

What if I used?

KB ≡ PREMISE1 && PREMISE2 && … && 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

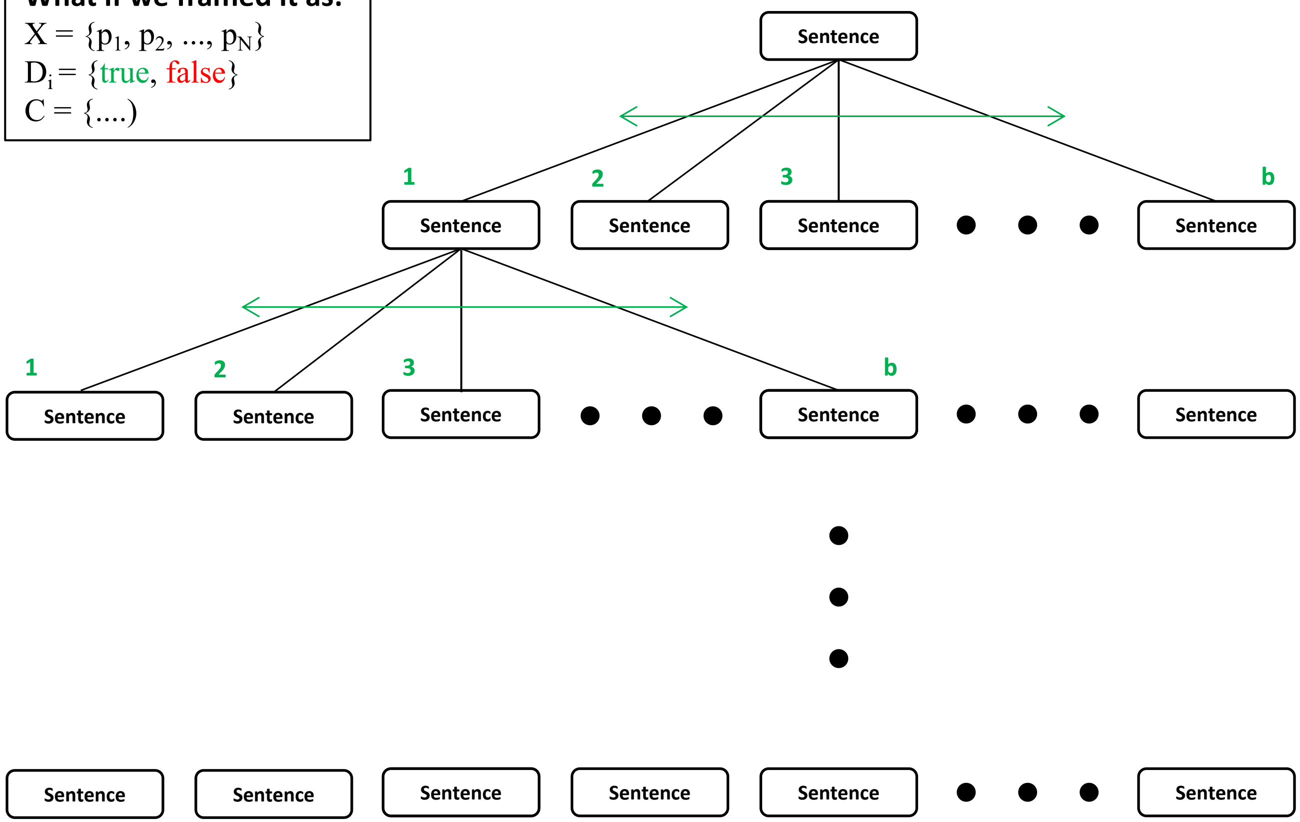
Truth Table Enumeration as Search

What if we framed it as:

$X = \{p_1, p_2, \ldots, p_N\}$

$D_i = \{$true, false$\}$

$C = \{\ldots.)$



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

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:

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

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

$(p \vee q) \wedge (\neg q \vee r) \rightarrow (true \vee false) \wedge (\neg false \vee true)$ Subsitute

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

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

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

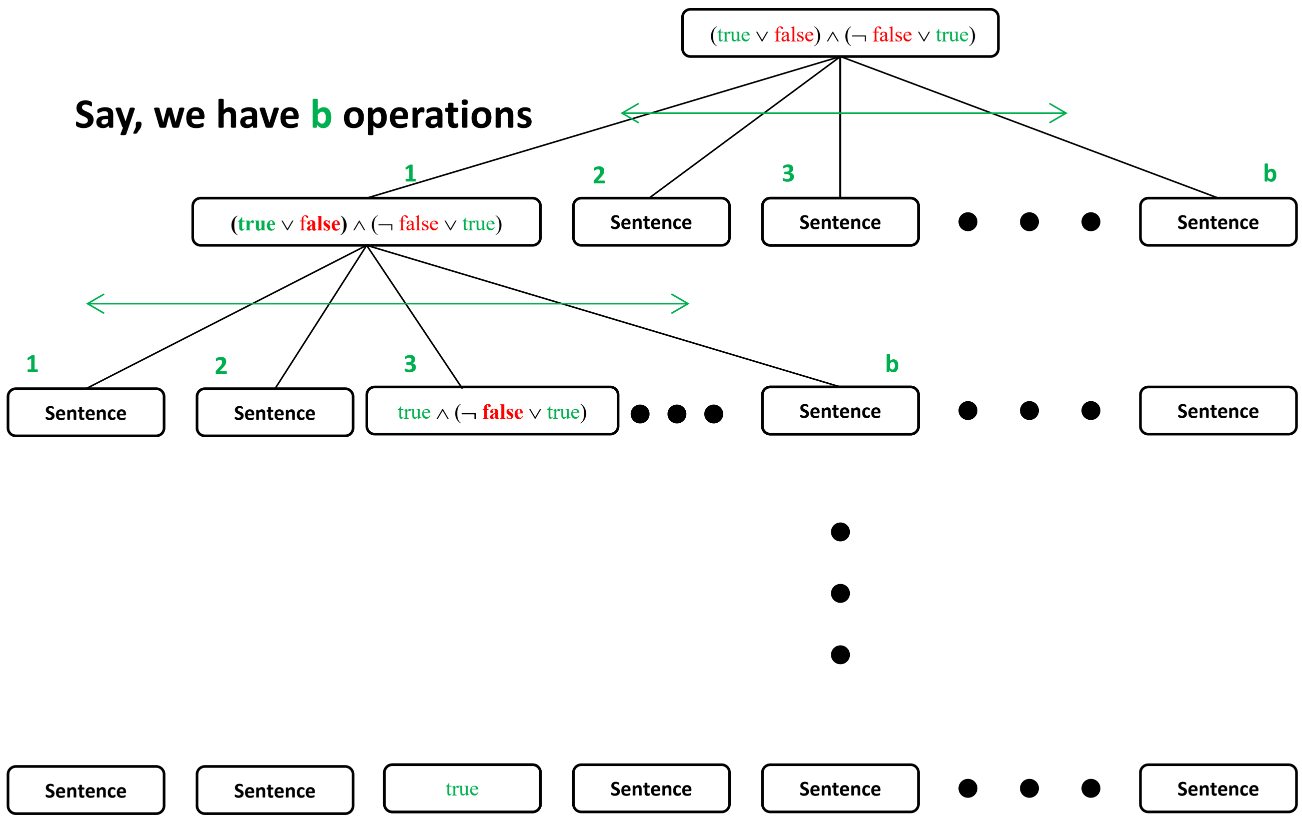
$\mathbf{true} \wedge \mathbf{true}$ Conjunction

$true$ Interpretation

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

Sentence Evaluation as Searching

Say, we have b operations



Depth: 0
Substition

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:**

$((\neg m \wedge \neg n) \vee (n \wedge \neg n)) \Rightarrow \neg m$	by Distributive law $p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$
$((\neg m \wedge \neg n) \vee \bot) \Rightarrow \neg m$	by Negation law (contradiction) $p \wedge \neg p \Leftrightarrow \bot$
$(\neg m \wedge \neg n) \Rightarrow \neg m$	by Identity law $p \vee \bot \Leftrightarrow p$
$\neg(\neg m \wedge \neg n) \vee \neg m$	by Implication law $\neg p \vee q \Leftrightarrow p \Rightarrow q$
$(\neg\neg m \vee \neg\neg n) \vee \neg m$	by De Morgan's law $\neg (p \wedge q) \Leftrightarrow \neg q \vee \neg p$
$(m \vee n) \vee \neg m$	by Double Negation law $\neg (\neg p) \Leftrightarrow p$
$m \vee (n \vee \neg m)$	by Associative law $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$
$m \vee (\neg m \vee n)$	by Commutative law $p \vee q \Leftrightarrow q \vee p$
$(m \vee \neg m) \vee n$	by Associative law $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$
$\top \vee n$	by Law of Excluded Middle $p \vee \neg p \Leftrightarrow \top$
$n \vee \top$	by Commutative law $p \vee q \Leftrightarrow q \vee p$
$\top$	by Domination Law $p \vee \top \Leftrightarrow \top$

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:**
$((\neg m \wedge \neg n) \vee (n \wedge \neg n)) \Rightarrow \neg m$	by Distributive law $p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$
$((\neg m \wedge \neg n) \vee \bot) \Rightarrow \neg m$	by Negation law (contradiction) $p \wedge \neg p \Leftrightarrow \bot$
$(\neg m \wedge \neg n) \Rightarrow \neg m$	by Identity law $p \vee \bot \Leftrightarrow p$
$\neg(\neg m \wedge \neg n) \vee \neg m$	by Implication law $\neg p \vee q \Leftrightarrow p \Rightarrow q$
$(\neg\neg m \vee \neg\neg n) \vee \neg m$	by De Morgan's law $\neg (p \wedge q) \Leftrightarrow \neg q \vee \neg p$
$(m \vee n) \vee \neg m$	by Double Negation law $\neg (\neg p) \Leftrightarrow p$
$m \vee (n \vee \neg m)$	by Associative law $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$
$m \vee (\neg m \vee n)$	by Commutative law $p \vee q \Leftrightarrow q \vee p$
$(m \vee \neg m) \vee n$	by Associative law $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$
$\top \vee n$	by Law of Excluded Middle $p \vee \neg p \Leftrightarrow \top$
$n \vee \top$	by Commutative law $p \vee q \Leftrightarrow q \vee p$
$\top$	by Domination Law $p \vee \top \Leftrightarrow \top$

How were the laws chosen for each step?

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

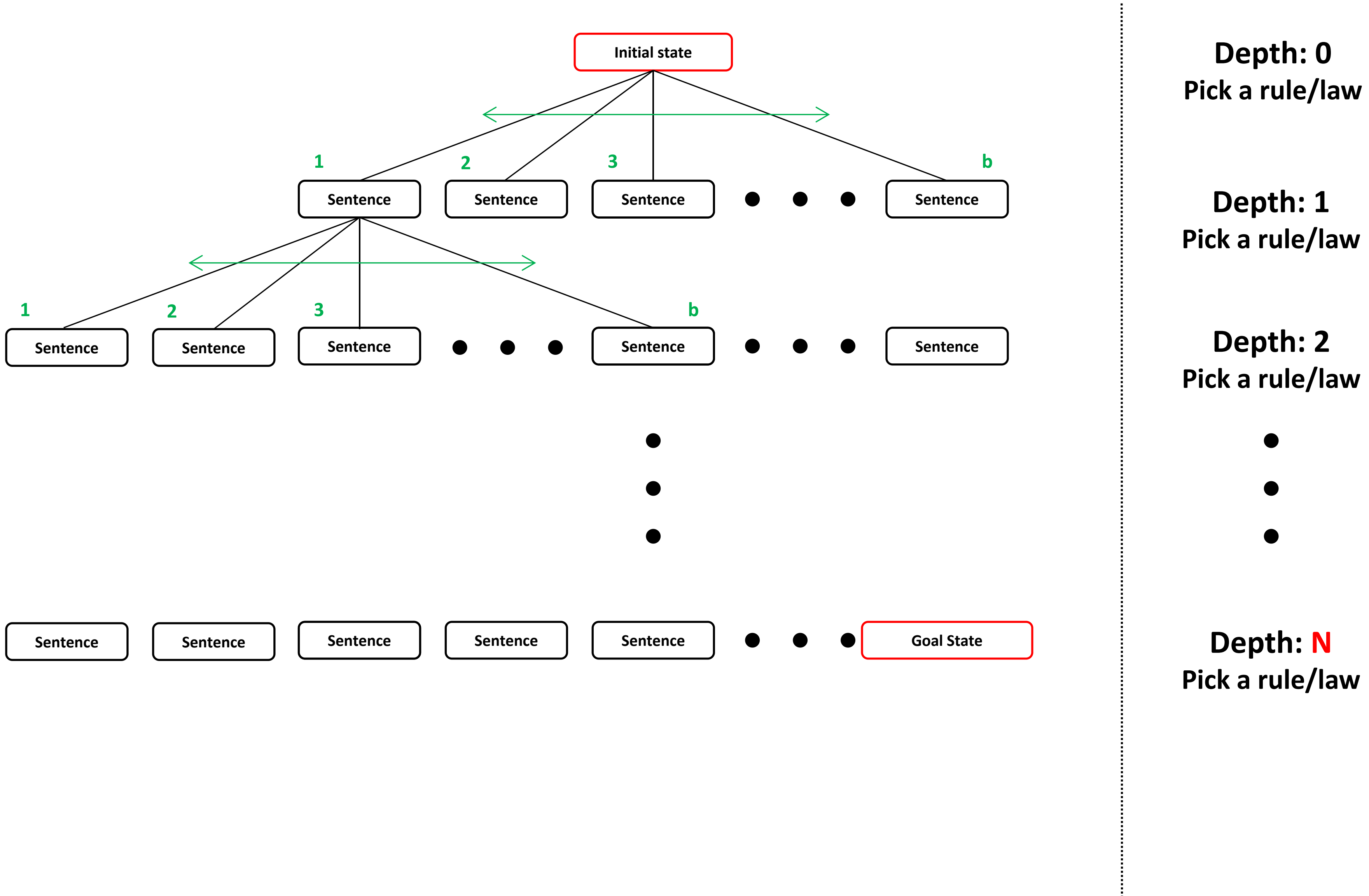
Proof as Search

Search algorithms can be used to find a sequence f steps that consitute 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



Deduction / Proof

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

Prove that $((\neg m \lor n) \land \neg n) \Rightarrow \neg m$ **is a tautology:**

$((\neg m \land \neg n) \lor (n \land \neg n)) \Rightarrow \neg m$	by Distributive law $p \land (q \lor r) \Leftrightarrow (p \land q) \lor (p \land r)$
$((\neg m \land \neg n) \lor \bot) \Rightarrow \neg m$	by Negation law (contradiction) $p \land \neg p \Leftrightarrow \bot$
$(\neg m \land \neg n) \Rightarrow \neg m$	by Identity law $p \lor \bot \Leftrightarrow p$
$\neg(\neg m \land \neg n) \lor \neg m$	by Implication law $\neg p \lor q \Leftrightarrow p \Rightarrow q$
$(\neg\neg m \lor \neg\neg n) \lor \neg m$	by De Morgan's law $\neg (p \land q) \Leftrightarrow \neg q \lor \neg p$
$(m \lor n) \lor \neg m$	by Double Negation law $\neg (\neg p) \Leftrightarrow p$
$m \lor (n \lor \neg m)$	by Associative law $(p \lor q) \lor r \Leftrightarrow p \lor (q \lor r)$
$m \lor (\neg m \lor n)$	by Commutative law $p \lor q \Leftrightarrow q \lor p$
$(m \lor \neg m) \lor n$	by Associative law $(p \lor q) \lor r \Leftrightarrow p \lor (q \lor r)$
$T \lor n$	by Law of Excluded Middle $p \lor \neg p \Leftrightarrow T$
$n \lor T$	by Commutative law $p \lor q \Leftrightarrow q \lor p$
T	by Domination Law $p \lor T \Leftrightarrow T$

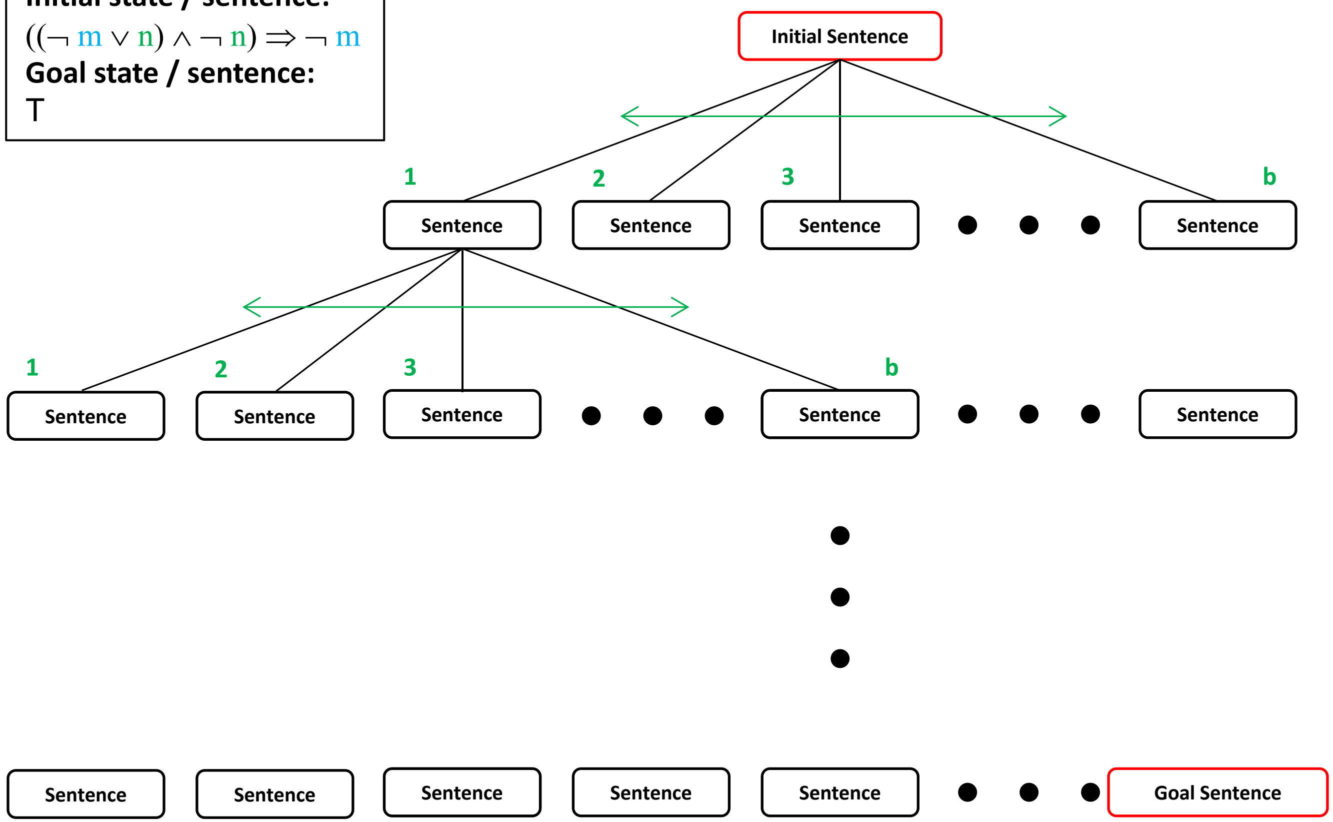
Initial state

Goal state

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

Deduction as Search

Initial state / sentence:
$((\neg m \lor n) \land \neg n) \Rightarrow \neg m$
Goal state / sentence:
T



Depth: 0
Pick a rule/law

Depth: 1
Pick a rule/law

Depth: 2
Pick a rule/law

Depth: N
Pick a rule/law

Model Checking: Q is Satisfiable

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

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

$KB \equiv P1 \wedge P2 \wedge P3$ **|** $Q \equiv$ **.....**

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**

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:**

$((\neg m \wedge \neg n) \vee (n \wedge \neg n)) \Rightarrow \neg m$	by Distributive law $p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$
$((\neg m \wedge \neg n) \vee \bot) \Rightarrow \neg m$	by Negation law (contradiction) $p \wedge \neg p \Leftrightarrow \bot$
$(\neg m \wedge \neg n) \Rightarrow \neg m$	by Identity law $p \vee \bot \Leftrightarrow p$
$\neg(\neg m \wedge \neg n) \vee \neg m$	by Implication law $\neg p \vee q \Leftrightarrow p \Rightarrow q$
$(\neg\neg m \vee \neg\neg n) \vee \neg m$	by De Morgan's law $\neg (p \wedge q) \Leftrightarrow \neg q \vee \neg p$
$(m \vee n) \vee \neg m$	by Double Negation law $\neg (\neg p) \Leftrightarrow p$
$m \vee (n \vee \neg m)$	by Associative law $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$
$m \vee (\neg m \vee n)$	by Commutative law $p \vee q \Leftrightarrow q \vee p$
$(m \vee \neg m) \vee n$	by Associative law $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$
$\mathrm{T} \vee n$	by Law of Excluded Middle $p \vee \neg p \Leftrightarrow \mathrm{T}$
$n \vee \mathrm{T}$	by Commutative law $p \vee q \Leftrightarrow q \vee p$
T	by Domination Law $p \vee \mathrm{T} \Leftrightarrow \mathrm{T}$

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

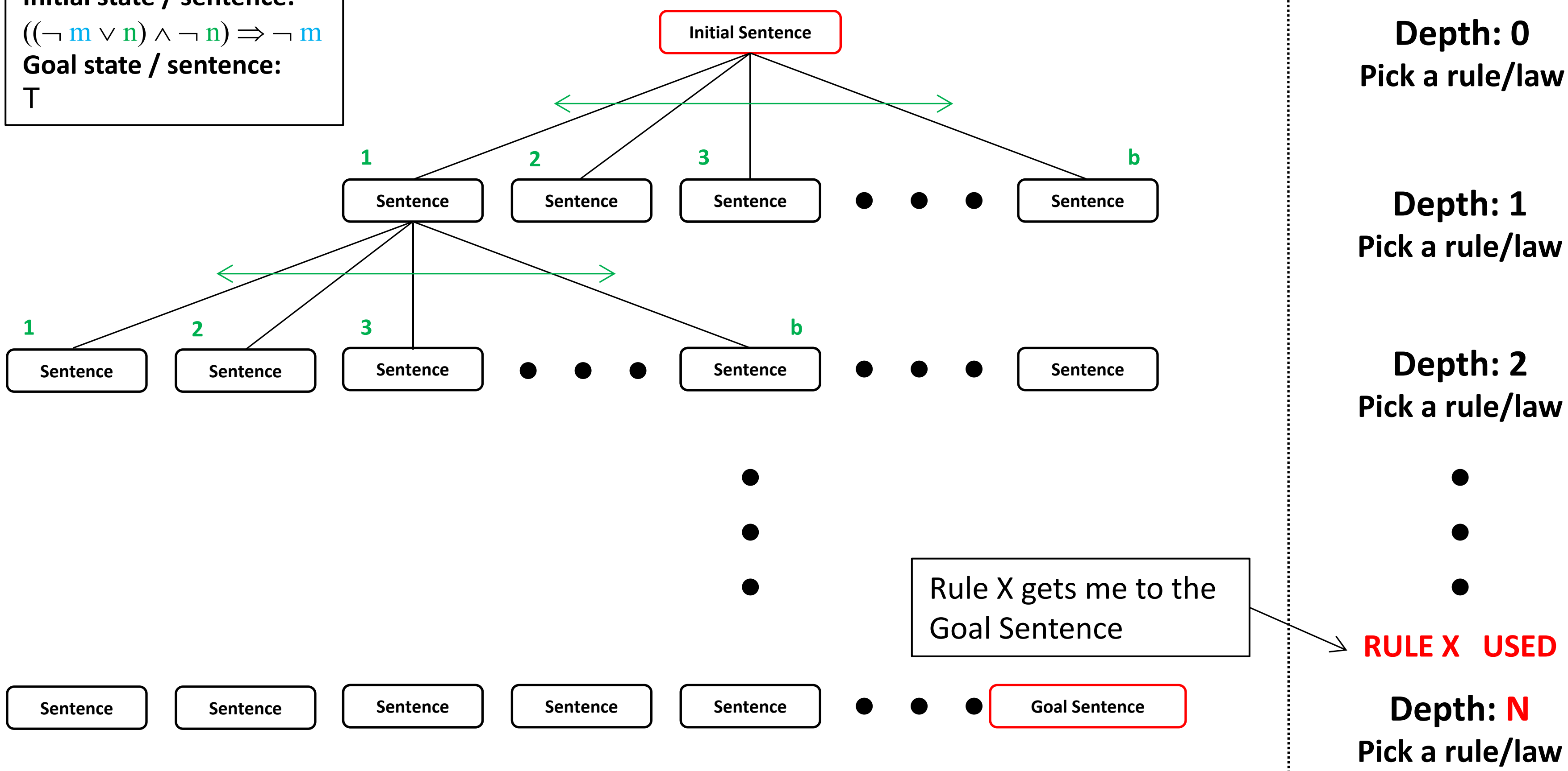
Syntax level

$$KB \xrightarrow{\text{Derivation}} Q$$

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



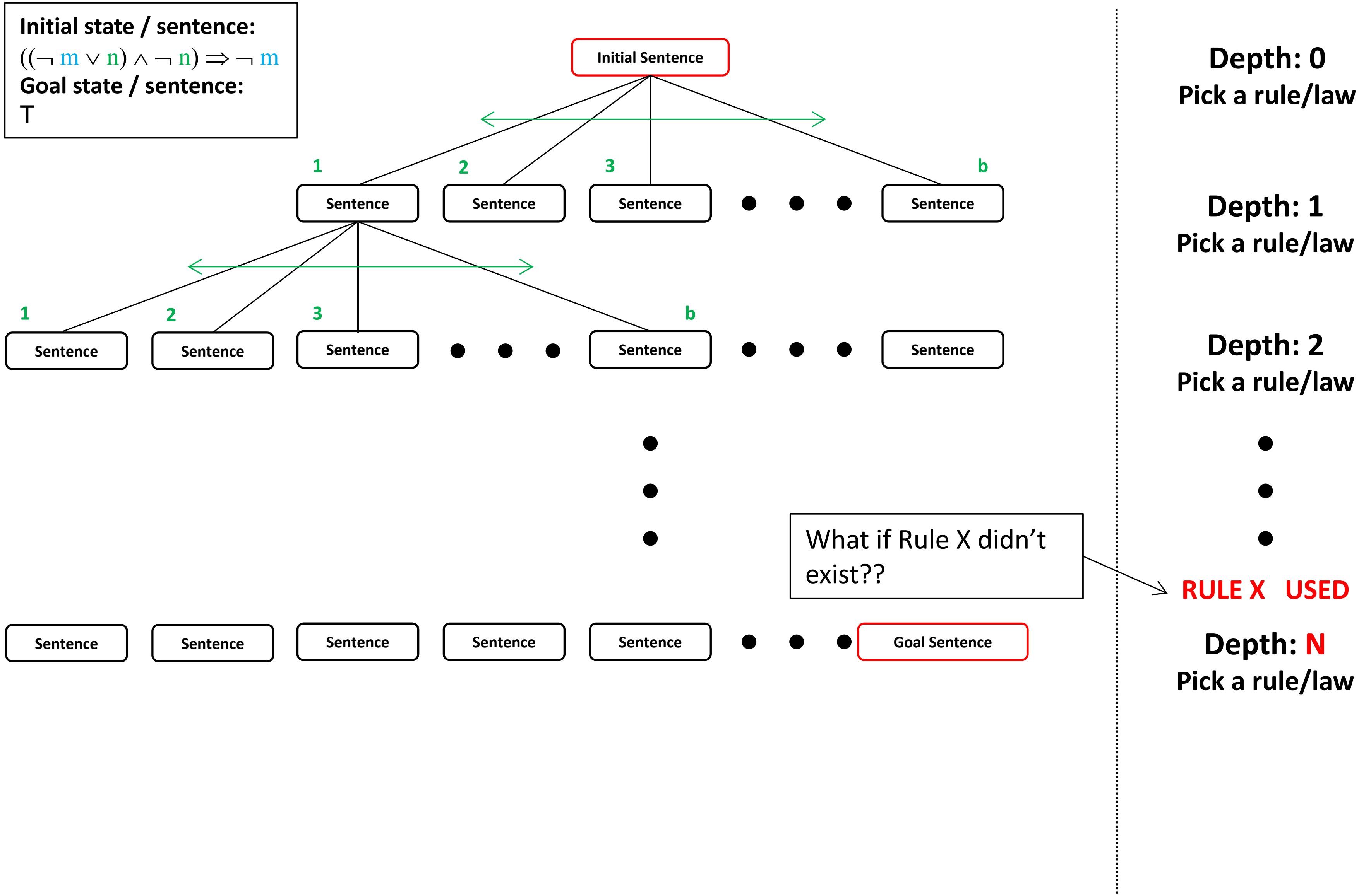
Deduction as Search

Initial state / sentence:

$((\neg m \lor n) \land \neg n) \Rightarrow \neg m$

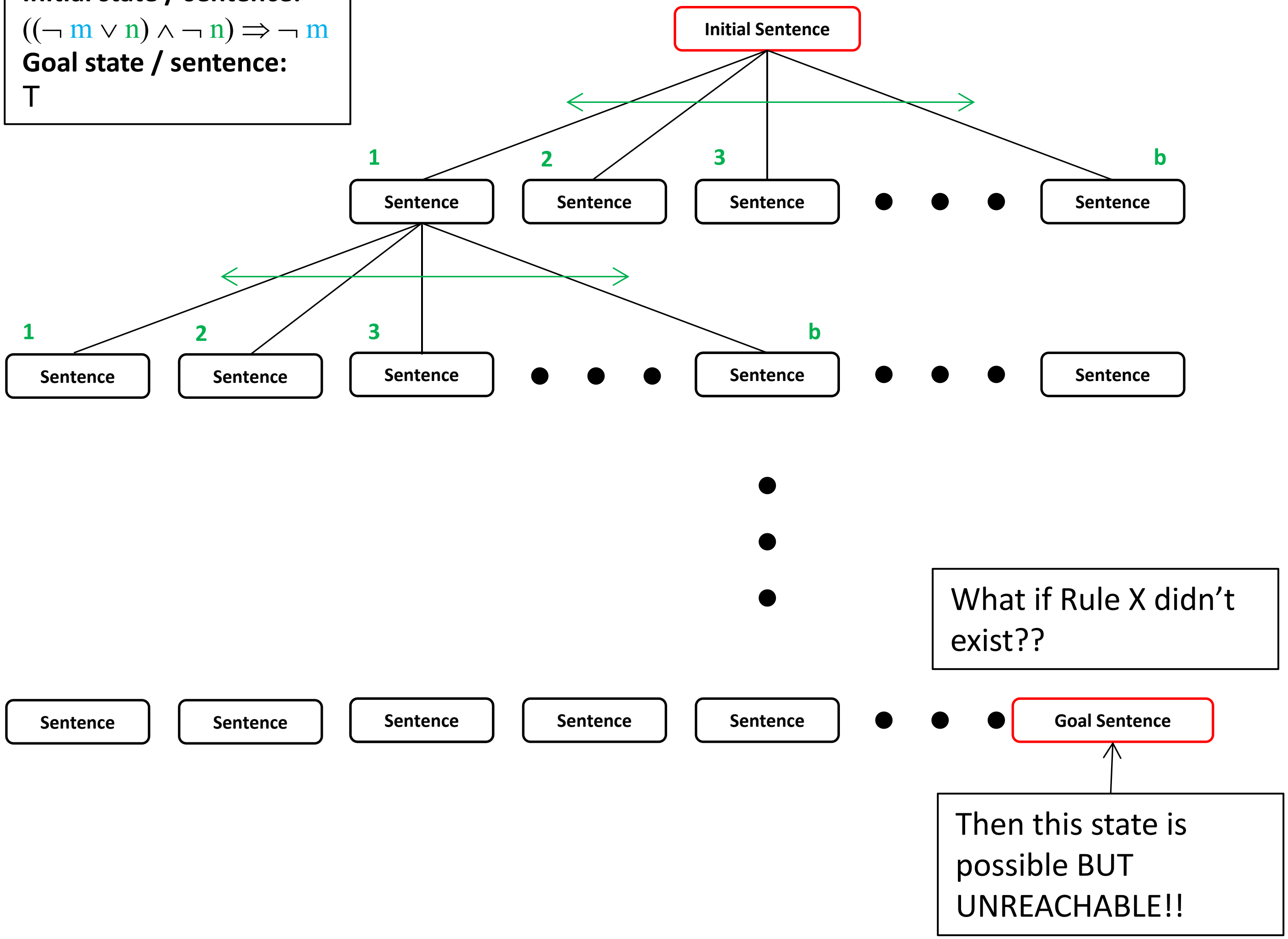
Goal state / sentence:

T



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

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 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

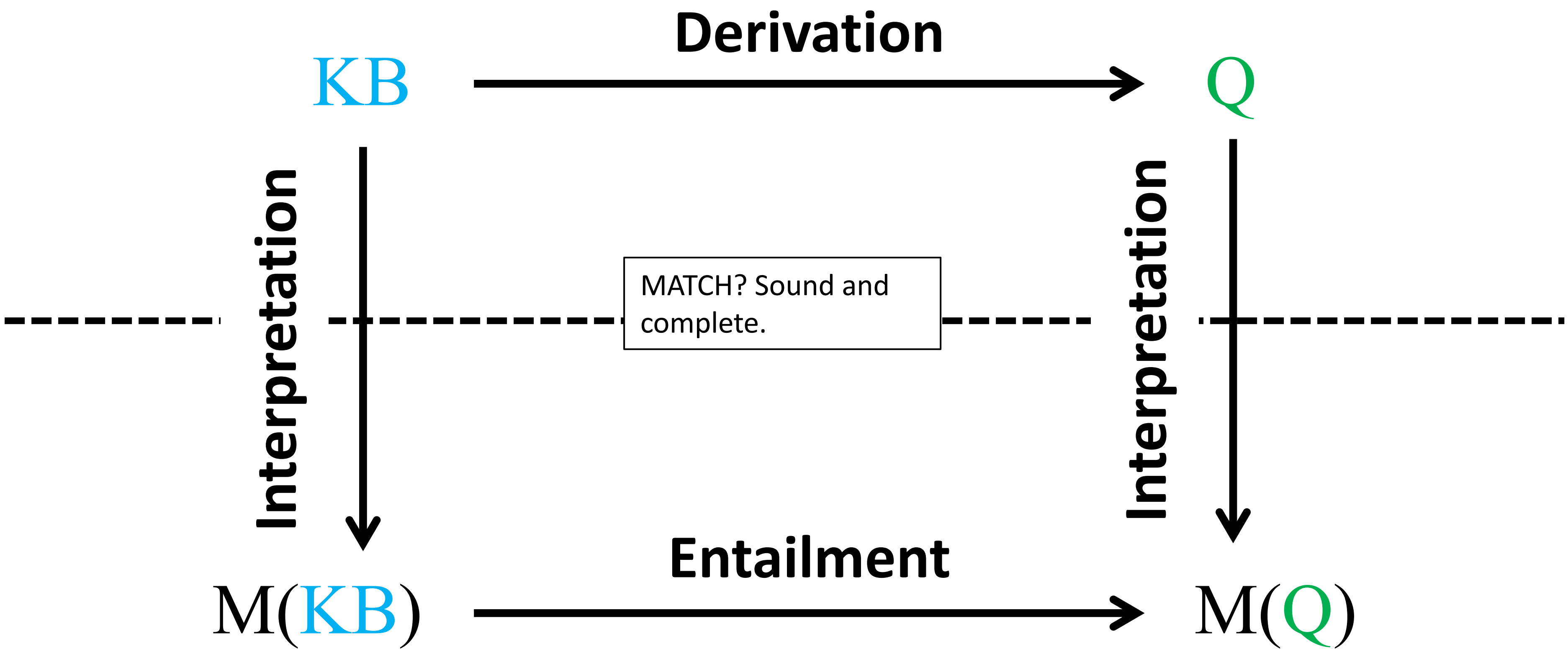
$$KB \xrightarrow{\text{Derivation}} Q$$

$$M(KB) \xrightarrow{\text{Entailment}} M(Q)$$

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

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

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

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

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

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

Proof by Resolution

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

$$KB \models Q$$

by proving that:

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

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

Resolution: Two Forms of Notation

Resolution

$P \lor Q$

$\neg P \lor R$

$\therefore\ Q \lor R$

Resolution (textbook)

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

Resolution: Two Forms of Notation

Resolution

$P \vee Q$

$\neg P \vee R$

$\therefore Q \vee R$

Resolution (textbook)

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

derived clause (resolvent)

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

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

* you can encounter it elsewhere in literature

Conjunctive Normal Form (CNF)

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

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

of clauses. A clause Ki consists of a disjunction

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

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

Conjunctive Normal Form (CNF)

Example:

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

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

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

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

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

we can remove parentheses

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

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

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

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

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

Conjunctive Normal Form (CNF)

Example:

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

literals

$$\underline{(\neg m \vee n \vee o)} \wedge \underline{(\neg n \vee m)} \wedge \underline{(\neg o \vee m)}$$

clauses

CNF Grammar

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

*** 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 b and ¬ b are complimentary. The resolution rule deletes a pair of complimentary literals from two clauses and combines the rest.

Unit Resolution

General resolution rule allows clauses with arbitrary number of literals

initial clauses

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

derived clause (resolvent)

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

Unit Resolution

General resolution rule allows clauses with arbitrary number of literals

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

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

Factorization

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

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

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

Resolution and Factorization

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

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

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

Proof by Resolution: Example

Consider the following problem:

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

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

Proof by Resolution: Example

Formalization step (English to Propositional Logic):

Propositional variables:

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

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

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

Sentences (bets):

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

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

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

Proof by Resolution: Example

Claim step (what are we trying to prove):

Claim: the three CANNOT all win their bets

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

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

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

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

We want to prove Claim by Contradiction:

Negated Claim: all three CAN win their bets

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

Proof by Resolution: Example

Convert negated claim to CNF step:

Original negated claim:

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

So:

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

$(a \Leftrightarrow \neg b) \equiv (\neg a \vee \neg b) \wedge (b \vee a)$ **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:

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

Resolution applied to clauses 1 and 6

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

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

Known clauses:

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

Added clauses:

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

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

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

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:

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

Resolution applied to clauses 3 and 9

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

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

Known clauses:

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

Added clauses:

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

Proof by Resolution: Example

Resolution steps:

Using resolution rule:

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

Resolution applied to clauses 8 and 10

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

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

Known clauses:

1. $(\neg a \vee \neg b)$
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** $KB \land \neg Q$

C. **convert** $KB \land \neg Q$ **into CNF ("standardized") form**

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

E. **Repeat (C) until:**

   a. **no new clause can be added (**KB **does NOT entail** Q**)**

   b. **last two clauses resolve to yield the empty clause (**KB **entails** Q**)**

Logical Entailment

So far, we have been asking the question:

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

$$KB \vDash Q$$

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?”

$$KB \models \begin{cases} \to Q_1 \\ \to Q_2 \\ \to \ldots \\ \to Q_{m-1} \\ \to Q_m \end{cases}$$

Logical Entailment

But we could ask the following question:

"Which Qs follow from KB?"

Q_1 follows from KB

Q_2 does NOT

$KB \vDash$ ...

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?"

ACTION$_1$ follows from

ACTION$_2$ does NOT follow

CURRENTKB $\models$ ...

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?"

$$(\mathrm{KBBEFORE} \wedge \mathrm{percept}) \vDash \begin{cases} \mathrm{ACTION}_1 & \text{follows from} \\ \mathrm{ACTION}_2 & \text{does NOT follow} \\ \ldots & \\ \mathrm{ACTION}_{m-1} & \text{follows from} \\ \mathrm{ACTION}_m & \text{follows from} \end{cases}$$