CS 480

Introduction to Artificial Intelligence

February 15, 2022

Announcements / Reminders

- **Midterm: February 24th!**
  - **Online section: please make arrangements. Contact Mr. Charles Scott (scott@iit.edu) if in doubt**
- **Written Assignment #01:**
  - **due: February 17th, 11:00 PM CST**
- **Written Assignment #02: will be posted this weekend**
- **Programming Assignment #01:**
  - **due: March 6th, 11:00 PM CST**

- **Please follow the Week 05 To Do List instructions**
- **Spring Semester midterm course evaluation reminder**
- **Grading TA assignment:**

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

Programming Assignment #01

- You CAN use textbook GitHub code:
  - https://github.com/aimacode/aima-python
  - make sure you REFERENCE/CITE it in your comment AND report document
- Use the Blackboard Discussion to ask for tips:
  - Please don't post significant portions of code there in response.
  - Tips and hints only

Teaching Assistants [UPDATE]

Name	e-mail	Office hours
Amit Nikam	anikam@hawk.iit.edu	Fridays 10:00 AM - 11:00 AM CST [Zoom and SB 108] Zoom link: https://iit-edu.zoom.us/j/87551262917?pwd=WG9BRkpyMFpreCtsTU45KzZMM2NCZz09
Juseung Lee	jlee302@hawk.iit.edu	Fridays 12:00 PM - 01:00 PM CST [Zoom ONLY] Zoom link: https://iit-edu.zoom.us/j/85493739308?pwd=VVBiQmRyVXlvdjRiN1NSTWV4QW5pdz09

TAs will:

- assist you with your assignments,
- hold office hours to answer your questions,
- grade your lab work (**a specific TA will be assigned to you**).

Take advantage of their time and knowledge!

DO NOT email them with questions unrelated to lab grading.

Make time to meet them during their office hours.

Add a **[CS480 Spring 2022]** prefix to your email subject when contacting TAs, please.

Plan for Today

- **Constraint Satisfaction Problems: continued**
- **Propositional logic**

Searching with Inference

```
function BACKTRACKING-SEARCH(csp) returns a solution or failure
  return BACKTRACK(csp, { })

function BACKTRACK(csp, assignment) returns a solution or failure
  if assignment is complete then return assignment
  var ← SELECT-UNASSIGNED-VARIABLE(csp, assignment)
  for each value in ORDER-DOMAIN-VALUES(csp, var, assignment) do
      if value is consistent with assignment  then
         add {var = value} to assignment
         inferences ← INFERENCE(csp, var, assignment)
         if inferences ≠ failure then
            add inferences to csp
            result ← BACKTRACK(csp, assignment)
            if result ≠ failure then return result
            remove inferences from csp
         remove {var = value} from assignment
  return failure
```

Apply local consistency checks and report failure if you know that following given path is going to dead end

Searching with Inference

Two key ideas:

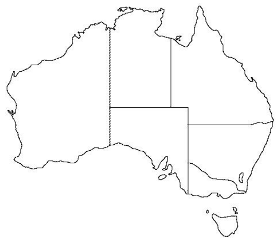
- **Forward checking**
- **Maintaining Arc Consistency**

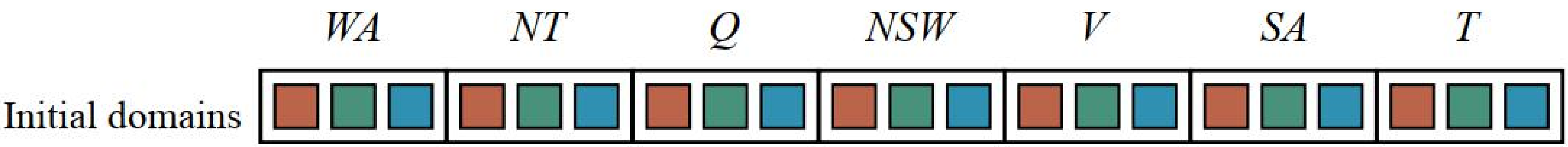
Forward Checking

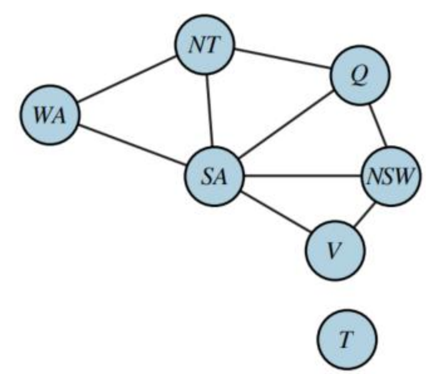
Idea:

After some value a is assigned to variable X, examine every unassigned variable Y connected to X by a constraint and delete values from Y's domain that are inconsistent with a

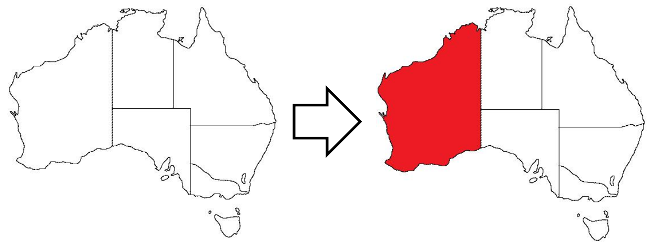
Forward Checking: Map of Australia

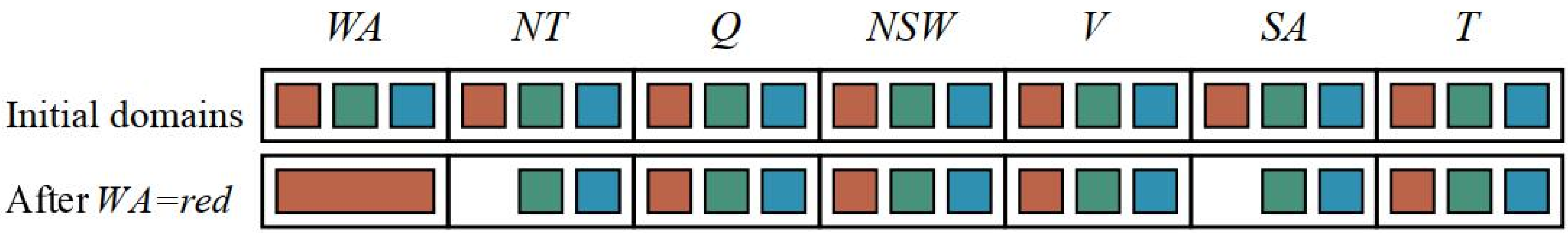


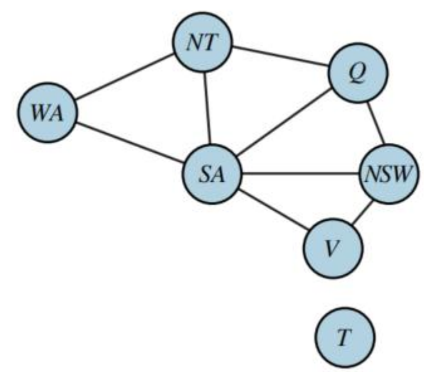




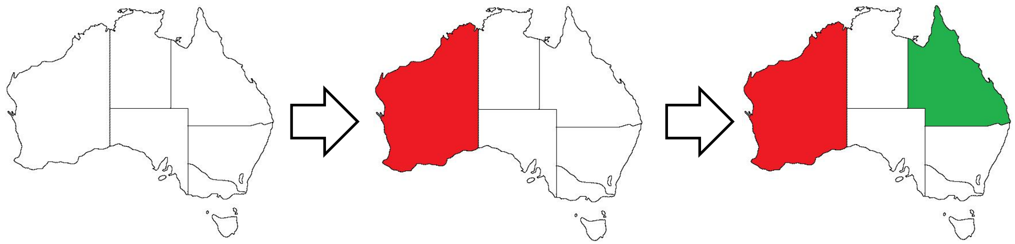
Forward Checking: Map of Australia

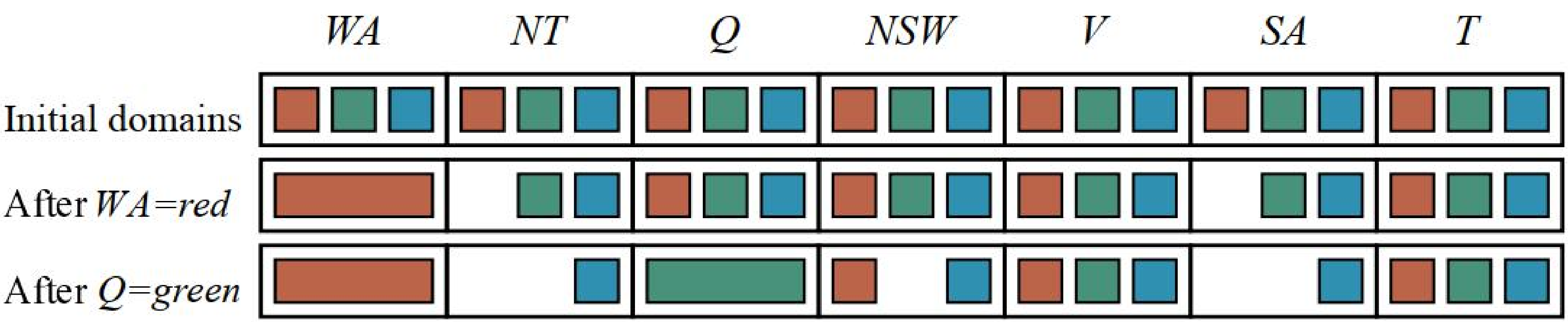


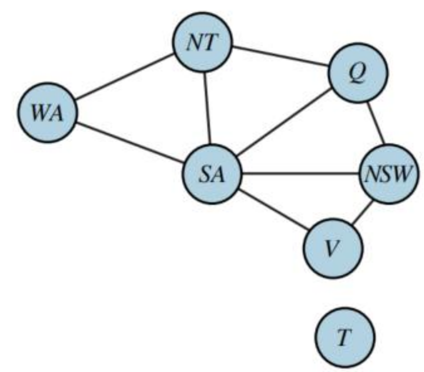




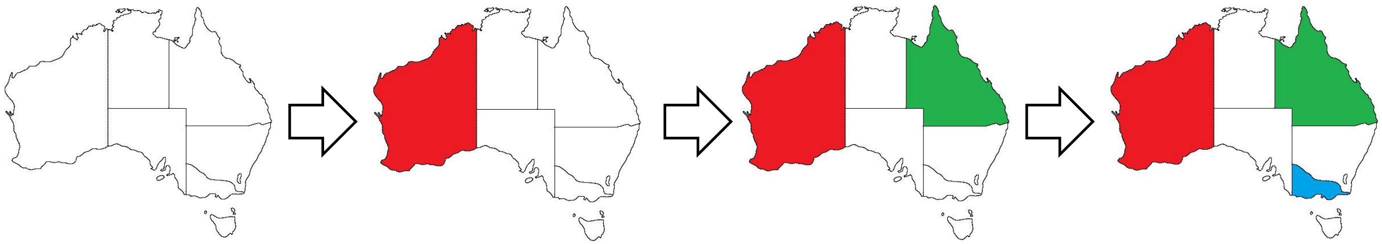
Forward Checking: Map of Australia

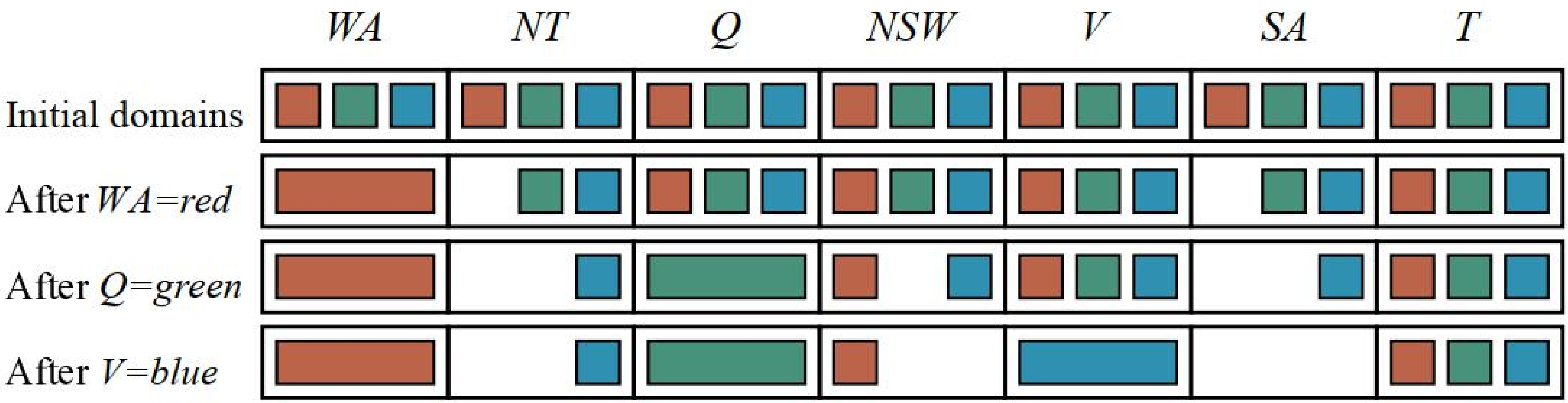


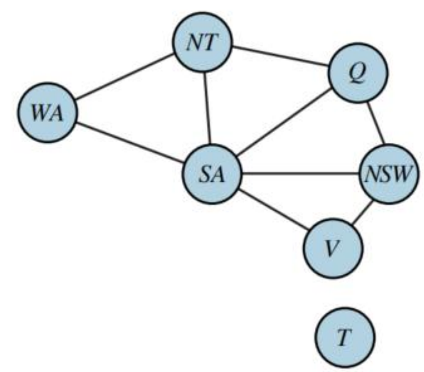




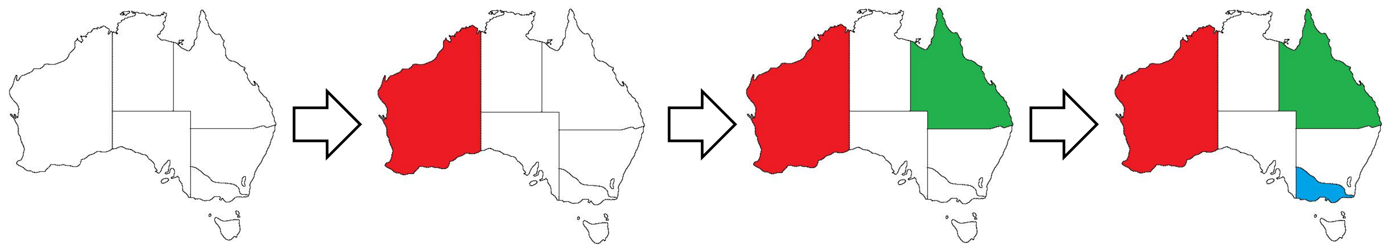
Forward Checking: Map of Australia

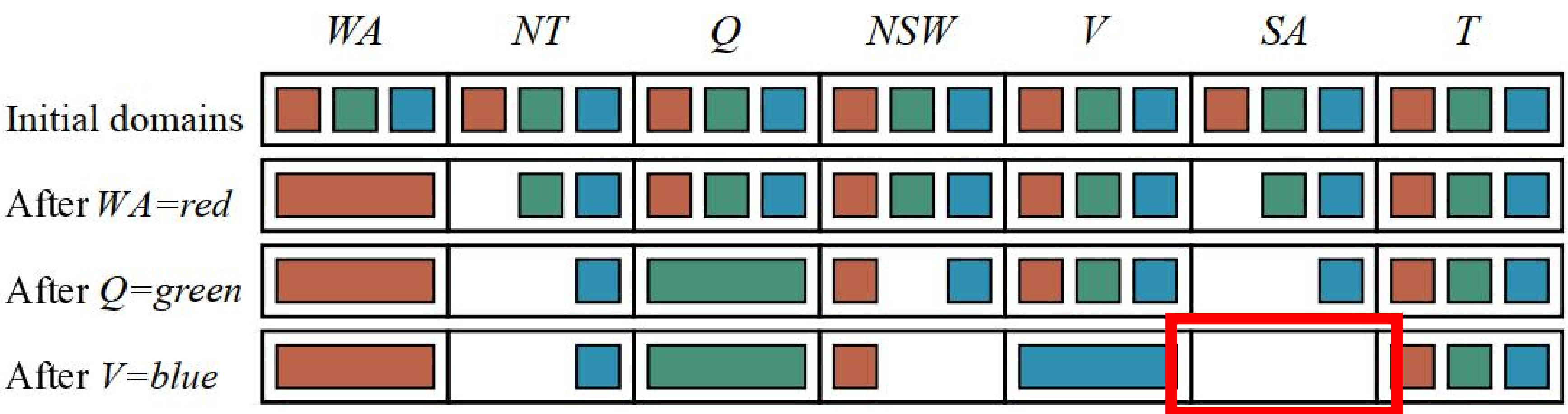


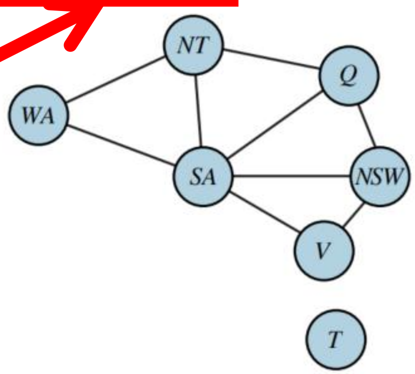




Forward Checking: Map of Australia

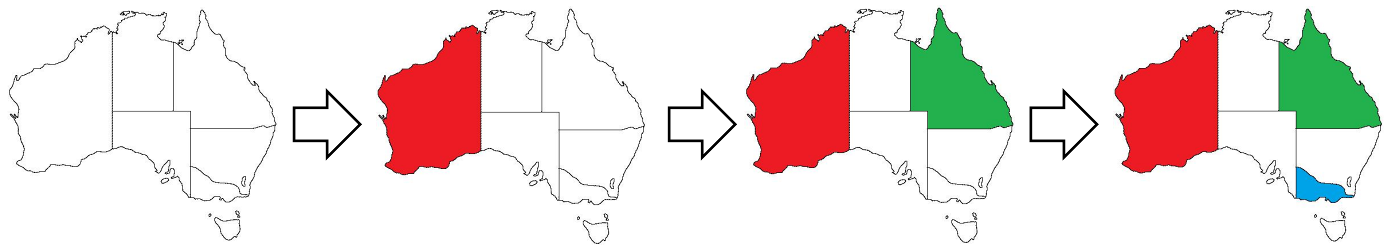


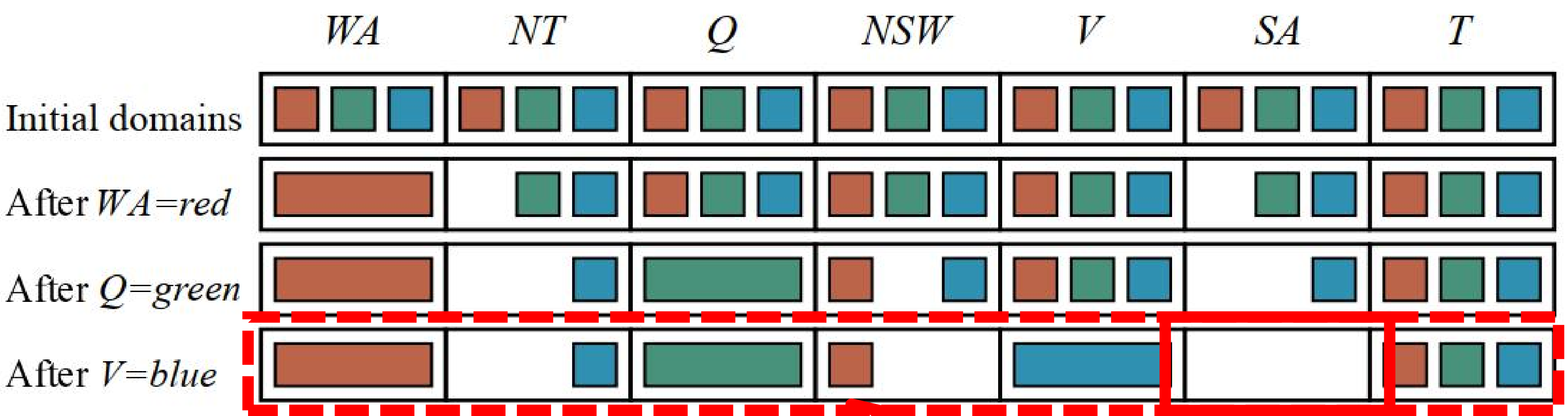


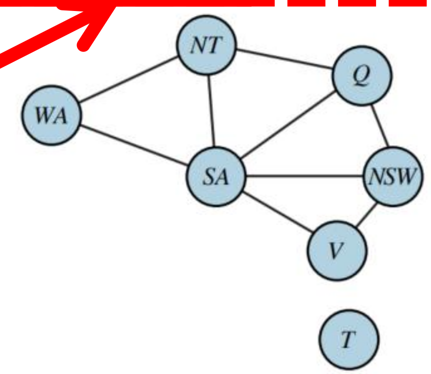


SA domain is empty!

Forward Checking: Map of Australia



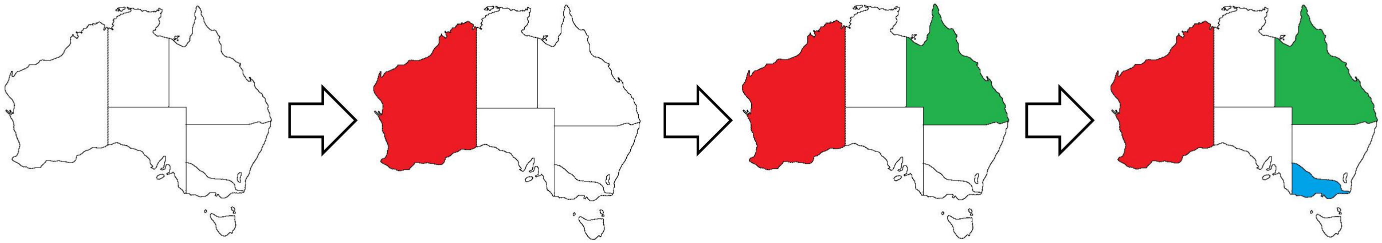


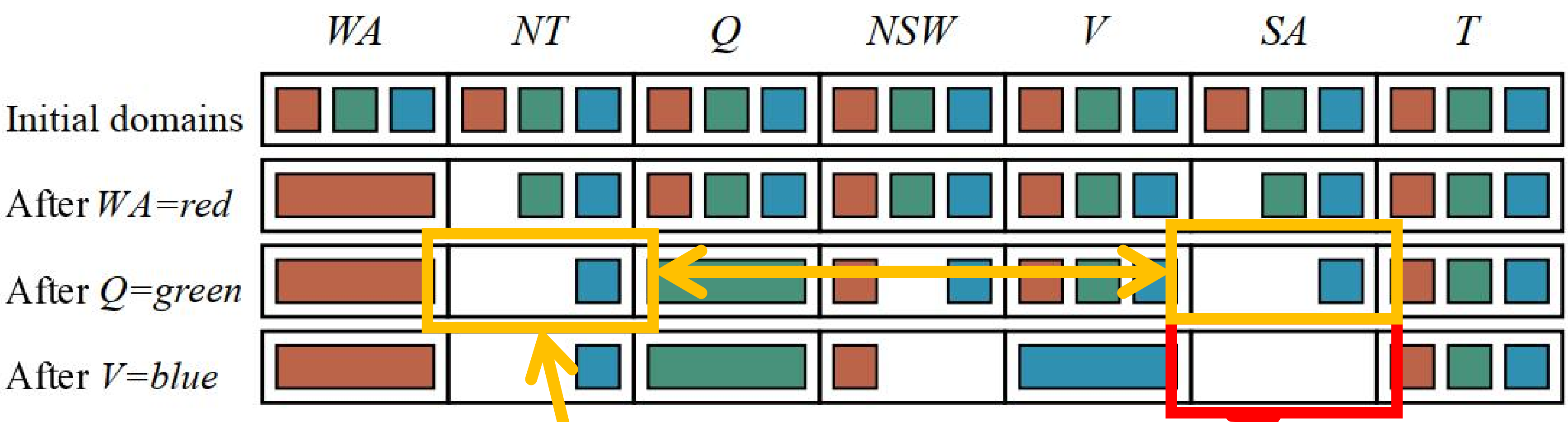


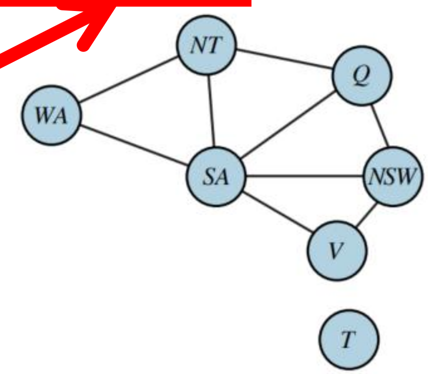
Inconsistent assignment

SA domain is empty!

Forward Checking: Map of Australia







Path inconsistency missed!

SA domain is empty!

AC-3 Algorithm: Pseudocode

function AC-3(*csp*) **returns** false if an inconsistency is found and true otherwise
  queue ← a queue of arcs, initially all the arcs in *csp*

  while *queue* is not empty **do**
    $(X_i, X_j) \leftarrow$ POP(*queue*)
    if REVISE(*csp*, X_i, X_j) **then**
      if size of $D_i = 0$ **then return** *false*
      for each X_k **in** X_i.NEIGHBORS - $\{X_j\}$ **do**
        add (X_k, X_i) to *queue*
  return *true*

Note: treat a constraint graph edge as two directional edges: constraint $X_i \neq X_j$ corresponds to edges (X_i, X_j) and (X_j, X_i)

function REVISE(*csp*, X_i, X_j) **returns** true iff we revise the domain of X_i
  revised ← *false*
  for each x **in** D_i **do**
    if no value y in D_j allows (x,y) to satisfy the constraint between X_i and X_j **then**
      delete x from D_i
      revised ← *true*
  return *revised*

Maintaing Arc-Consistency Algorithm

Idea:

After some value is assigned to variable X_i, infer by calling AC3 algorithm, but with a reduced number of edges / arcs for its queue:

- **only (X_i, X_j) arcs for all X_j variables that:**
  - **are constrained by X_i (neighbors of X_i on the constraint graph)**
  - **have no value assigned**

MAC Algorithm Call to AC3

function AC-3(*csp*) **returns** false if an inconsistency is found and true otherwise
  queue ← a queue of arcs, ~~initially all the arcs in~~ *csp*

  while *queue* is not empty **do**
    $(X_i,\ X_j) \leftarrow$ POP(*queue*)
    if REVISE(*csp*, X_i, X_j) **then**
      if size of $D_i = 0$ **then return** *false*
      for each X_k **in** X_i.NEIGHBORS - $\{X_j\}$ **do**
        add $(X_k,\ X_i)$ to *queue*
  return *true*

only (Xi, Xj) arcs for all Xj variables that:

- are constrained by X_i (neighbors of X_i on the constraint graph)
- have no value assigned

function REVISE(*csp*, X_i, X_j) **returns** true iff we revise the domain of X_i
  revised ← *false*
  for each x **in** D_i **do**
    if no value y in D_j allows (x,y) to satisfy the constraint between X_i and X_j **then**
      delete x from D_i
      revised ← *true*
  return *revised*

Intelligent Backtracking

- **Chronological Backtracking:**
  - **Backpropagation used it**
- **Backjumping:**
  - **maintains a conflict set for a node X: a set of assignments that are in conflict with some X domain value**
  - **backtracks to a variable assignment level where a conflict (it ruled out some potential value of X earlier)**
  - **Forward checking can help construct conflict set**

Search Problems: Summary

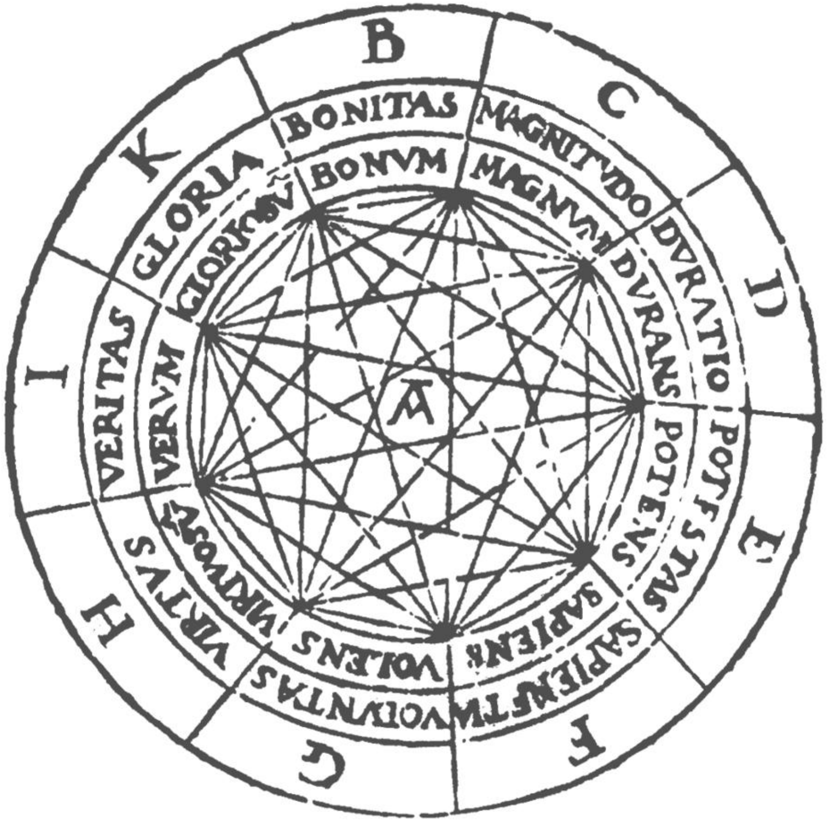
- **Initial problem analysis:**
  - **can it be represented with a state space?**
  - **what is the most useful state representation?**
  - **where, in the search tree, solution is expected? BFS or DFS?**
- **Do problem solutions need to be optimal?**
- **Do you care about time or space performance? Or both?**
- **Does your problem representation match known search algorithms?**
  - **Yes? Use it. No? See if you can make some simplifying assumptions and ask that question again**
- **Use all available knowledge about the problem to come with handy heuristics and use them to prune search tree**

Some CSP Challenges

- What if not all constraints can be satisfied?
  - Hard vs. soft constraints vs. preferences
  - Degree of constraint satisfaction concept
  - Cost of violating constraints
- What if constraints are of different forms?
  - Symbolic constraints
  - Logical constraints
  - Temporal constraints
  - Mixed constraints

Logical Agents and Reasoning

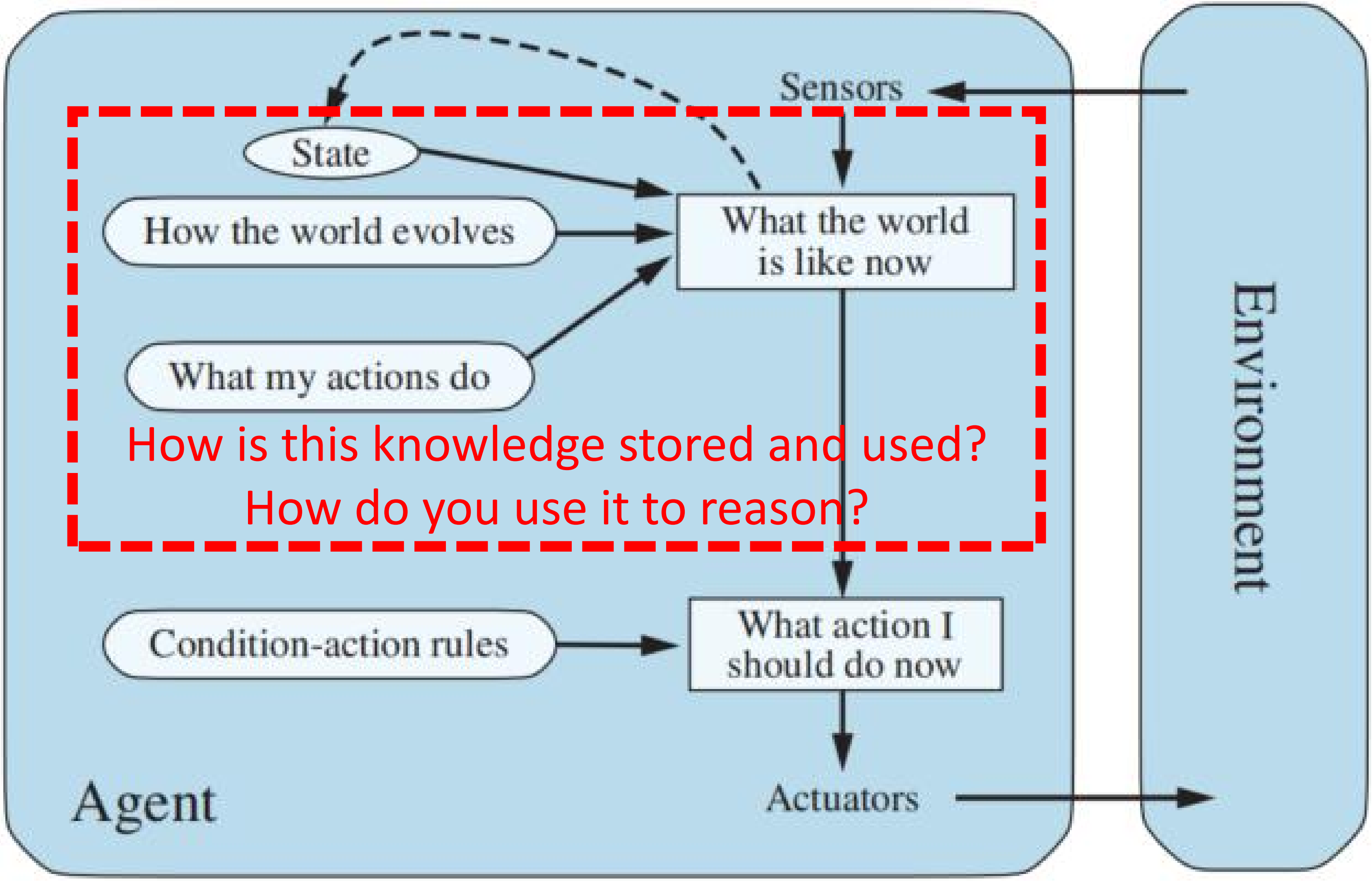
Llull’s Ars Magna (around 1305)



Catalan philosopher Ramon Llull in his book “Ars Magna”. It was an attempt to use logic to **artificially produce new knowledge by generating combinations of elemental truths** (a fixed set of preliminary ideas). Some consider it an **early step towards a “thinking machine”**.

Source: https://commons.wikimedia.org/wiki/File:Ramon_Llull_-_Ars_Magna_Fig_1.png

Knowledge-based Agent



Knowledge-based Agents

Knowledge-based agents use a process of reasoning over an internal representation of knowledge to decide what actions to take

Logic is one way to represent knowledge and reason:

- **Propositional logic**
- **First-order logic**

Knowledge-based Agents

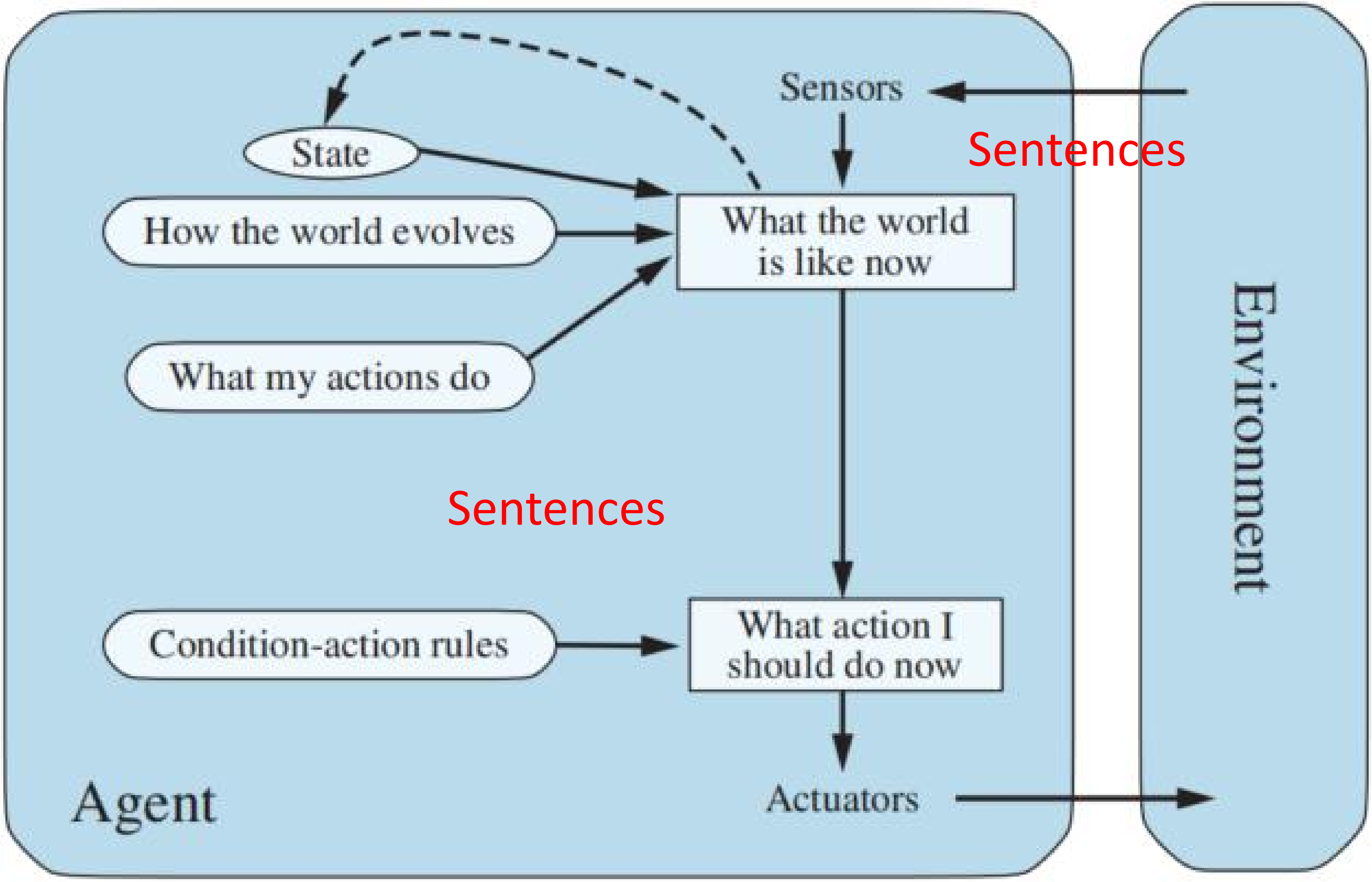
function KB-AGENT(*percept*) **returns** an *action*
    persistent: KB, a knowledge base
                  t, a counter, initially 0, indicating time

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

Knowledge-based Agents

- **Central component: Knowledge Base (KB)**
- **Knowledge Base is a set of sentences**
- **All Sentences are expressed in knowledge representation language**
- **Sentences can be:**
  - **given (axioms)**
  - **derived**
  - **used for inference**
- **KB can have background knowledge**

Knowledge-based Agent



Propositional Logic and KB-Agents

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

Propositional Logic

Propositional logic, also known as sentential logic and statement logic, is the branch of logic that studies ways of joining and/or modifying entire propositions, statements or sentences to form more complicated propositions, statements or sentences, as well as the logical relationships and properties that are derived from these methods of combining or altering statements.

Language: Syntax and Semantics

- **Syntax:**
  - **defines a set of rules for producing legal (well formed) sentences in a given language**
- **Semantics:**
  - **defines the "meaning" of a sentence → it has semantic value**
  - **NOT all legal sentences will have semantic value:**

  Example: Colorless green ideas sleep furiously

Proposition / Sentence

A proposition / sentence (also called a logical expression) is an assertion about the world in a mathematically defined knowledge representaton language. It can be true or false.

Examples:

John is sick

When it thunders, there is also lightning

Propositional Logic: Syntax

- **Logical constants:** true, false
- **Propositional symbols / variables:**
  - **atomic sentences:** p, q, r
  - **compound / complex sentences:** P, Q, R
- **Wrapping parentheses:** (…)
- **Sentences are combined by logical connectives:**

  $\neg \ \wedge \ \vee \ \Leftrightarrow \ \Rightarrow$
- **Literals:**
  - **atomic sentence** p **or negated atomic sequence** $\neg$p

Symbols: Refresher

Symbol	Name	Alternative symbols*	Should be read
$\neg$	Negation	$\sim$, !	not
$\wedge$	(Logical) conjunction	$\bullet$, &	and
$\vee$	(Logical) disjunction	$+$, $\Vert$	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

Creating Complex Sentences

atomic sentences

p _ q

logical connective

complex sentences

R _ S

logical connective

p, q, R, S - proposition (sentence) symbols / variables | _ logical connective: $\neg \wedge \vee \Leftrightarrow \Rightarrow$

Logical Connectives: $\neg \; \wedge \; \vee \; \Leftrightarrow \; \Rightarrow$

Negation (not)

literal (atomic sequence)

$$\neg p$$

negation

Logical conjunction (and)

conjuncts

$$p \wedge q$$

conjunction

Logical disjunction (or)

disjuncts

$$p \vee q$$

disjunction

Material implication and equivalence

premise* conclusion**

$$p \Rightarrow q$$

implication

$$p \Leftrightarrow q$$

biconditional

* also called antecedent | ** also called consequent

Operator Precedence

Operator Precedence

Higher precedence

$()$

$\neg$

$\wedge$

$\vee$

$\Rightarrow \; \Leftrightarrow$

Lower precedence

Precedence in Sentences

If in doubt: left can be rewritten as right

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

Well-formed Sentences

A well-formed sentence is a finite sequence of symbols from a given alphabet that is part of a formal language (grammatically correct)

- **well-formed propositional logic sentence:**

$$(((p \Rightarrow q) \land (r \Rightarrow s)) \lor (\neg q \land \neg s))$$

- **NOT well-formed propositional logic sentence:**

$$((p \Rightarrow q) \Rightarrow (qq))p))$$

BNF (Backus-Naur Form) Grammar

$$
\begin{array}{rcl}
Sentence & \rightarrow & AtomicSentence \mid ComplexSentence \\
AtomicSentence & \rightarrow & \boxed{True \mid False \mid P \mid Q \mid R \mid \ldots} \; * \\
ComplexSentence & \rightarrow & (\ Sentence\ ) \\
& \mid & \neg\ Sentence \\
& \mid & Sentence \land Sentence \\
& \mid & Sentence \lor Sentence \\
& \mid & Sentence \ \Rightarrow\ Sentence \\
& \mid & Sentence \ \Leftrightarrow\ Sentence \\
\\
\text{OPERATOR PRECEDENCE} & : & \neg, \land, \lor, \Rightarrow, \Leftrightarrow
\end{array}
$$

*** I will:**

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

English → Propositional Logic

Consider three atomic sentences in propositional logic: cool, funny, **and** popular. **Each can be assigned a truth value of** true **or** false.

Natural language encoded using propositional logic examples:

IF a person is cool OR funny, THEN she is popular.

$$(\text{cool} \vee \text{funny}) \Rightarrow \text{popular}$$

A person is popular ONLY IF she is EITHER cool OR funny.

$$\text{popular} \Rightarrow (\text{cool} \vee \text{funny})$$

A person is popular IF AND ONLY IF she is EITHER cool OR funny.

$$\text{popular} \Leftrightarrow (\text{cool} \vee \text{funny})$$

There is NO one who is both cool AND funny.

$$\neg(\text{cool} \wedge \text{funny})$$

Propositional Logic: Laws/Theorems

Equivalence	Law / Theorems
$p \vee q \Leftrightarrow q \vee p$ $p \wedge q \Leftrightarrow q \wedge p$	Commutative laws
$(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$ $(p \wedge q) \wedge r \Leftrightarrow p \wedge (q \wedge r)$	Associative laws
$p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$ $p \vee (q \wedge r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$	Distributive laws
$\neg (p \wedge q) \Leftrightarrow \neg q \vee \neg p$ $\neg (p \vee q) \Leftrightarrow \neg q \wedge \neg p$	De Morgan's laws
$p \wedge (p \vee q) \Leftrightarrow p$ $p \vee (p \wedge q) \Leftrightarrow p$	Absorption laws
$\neg (\neg p) \Leftrightarrow p$	Double Negation law (involution)
$p \wedge p \Leftrightarrow p$ $p \vee p \Leftrightarrow p$	Idempotent laws
$p \vee \neg p \Leftrightarrow \top$	Law of Excluded Middle (Negation law)
$p \wedge \neg p \Leftrightarrow \bot$	Contradiction (Negation law)
$p \wedge \top \Leftrightarrow p$ $p \vee \bot \Leftrightarrow p$	Identity laws
$p \wedge \bot \Leftrightarrow \bot$ $p \vee \top \Leftrightarrow \top$	Domination laws
$\neg p \vee q \Leftrightarrow p \Rightarrow q$	Implication law
$p \Rightarrow q \Leftrightarrow \neg q \Rightarrow \neg p$	Contraposition law
$(p \wedge q) \vee (\neg q \wedge \neg p) \Leftrightarrow (p \Leftrightarrow q)$ $(p \Rightarrow q) \wedge (q \Rightarrow p) \Leftrightarrow (p \Leftrightarrow q)$	Equivalence law

Deduction

Laws/theorems in propositional logic can be used to prove additional theorems througha 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 $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$
$T \vee n$	by Law of Excluded Middle $p \vee \neg p \Leftrightarrow T$
$n \vee T$	by Commutative law $p \vee q \Leftrightarrow q \vee p$
T	by Domination Law $p \vee T \Leftrightarrow T$

Deduction

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

Note that we only manipulated symbols at the syntactic level!

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 $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$
$T \vee n$	by Law of Excluded Middle $p \vee \neg p \Leftrightarrow T$
$n \vee T$	by Commutative law $p \vee q \Leftrightarrow q \vee p$
T	by Domination Law $p \vee T \Leftrightarrow T$

Propositional Logic and KB-Agents

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

Interpretation

The truth value assignment to propositional sentences is called an interpretation (an assertion about their truth in some possible world / model).

Definition: A mapping $I : \Sigma \rightarrow \{\text{true}, \text{false}\}$**, which assigns a truth value to every proposition variable, is called an interpretation.**

Sentence: $(p \vee q) \wedge (\neg q \vee r)$

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

Truth Values and Truth Tables

Propositional logic sentences can have a truth value assigned to them:

- **Atomic sentences:**
  - **either** true **or** false
- **Compound / complex sentence truth value can be established using a truth table:**

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \Rightarrow q$	$p \Leftrightarrow q$
true	true	false	true	true	true	true
true	false	false	false	true	false	false
false	true	true	false	true	true	false
false	false	true	false	false	true	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 $(p \vee q) \wedge (\neg q \vee r)$:

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

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 $(p \vee q) \wedge (\neg q \vee r)$**:**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$\textbf{true} \wedge \textbf{true}$ Conjunction

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 $(p \lor q) \land (\neg\, q \lor r)$:

$$(p \lor q) \land (\neg\, q \lor r) \rightarrow (true \lor false) \land (\neg\, false \lor true)$$ Subsitute

$$(\mathbf{true} \lor \mathbf{false}) \land (\neg\, false \lor true)$$ Disjunction

$$true \land (\neg\, \mathbf{false} \lor true)$$ Negation

$$true \land (\mathbf{true} \lor \mathbf{true})$$ Disjunction

$$\mathbf{true} \land \mathbf{true}$$ Conjunction

$$true$$ Interpretation

Complex Sentence: Truth Table

Consider a complex sentence R built with N propositional variables $p_1, p_2, p_3, \ldots, p_{N-1}, p_N$ and logical connectives ($\neg, \vee, \wedge, \Rightarrow, \Leftrightarrow$). Here is a corresponding truth table for sentence R.

	N Propositional Variables						Complex sentence R	
	p_1	p_2	p_3	…	p_{N-1}	p_N		
2^N Truth Assignments	true	true	true	…	true	true	false	2^N Interpretations of R
	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	

Complex Sentence: Truth Table

Consider a complex sentence R built with N propositional variables p_1, p_2, p_3, ..., p_{N-1}, p_N and logical connectives ($\neg$, $\vee$, $\wedge$, $\Rightarrow$, $\Leftrightarrow$). Each truth assignment is a different possible world.

	N Propositional Variables						Complex sentence R	
	p_1	p_2	p_3	...	p_{N-1}	p_N		
2^N Possible Worlds (Models)	true	true	true	...	true	true	false	2^N Interpretations of R
	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	

Sentence: Syntactic / Semantic Levels

Each propositional logic "exists" on two levels:

- **Syntactic: where a sentence is just a legal arrangement of language symbols. Sentence**

$$(p \lor q) \land (\neg\, q \lor r)$$

WITHOUT interpretation HAS NO MEANING

  - **we can manipulate symbols, but we CANNOT reason**

- **Semantic: where a sentence has a truth value is assigned to it (interpretation):**

$(p \lor q) \land (\neg\, q \lor r)$ **where** $p^i = \text{true}$, $q^i = \text{false}$, $r^i = \text{true}$

HAS MEANING (through interpretation) $\rightarrow$ it is true

Sentence: Syntactic / Semantic Levels

Each propositional logic "exists" on two levels:

- **Syntactic: where a sentence is just a legal arrangement of language symbols. Sentence**

$$(cool \vee funny) \Rightarrow popular$$

WITHOUT interpretation HAS NO MEANING

  - **we can't tell if a given person is** $popular$ **here**

- **Semantic: where a sentence has a truth value is assigned to it (interpretation):**

$(cool \vee funny) \Rightarrow popular$ **where** $cool = true,\ funny = false$

HAS MEANING $\rightarrow$ **we can deduce that a person is** $popular$

Sentence Semantical Equivalence

Two propositional logic sentences F and G are called <u>semantically</u> equivalent if they take on the same interpretation for all truth value assignments. If that is the case $\mathrm{F} \equiv \mathrm{G}$.

Example: sentence $\neg\ a \vee b$ is equivalent to sentence $a \Rightarrow b$. Proof with a truth table:

a	b	$\neg\ a$	$\neg\ a \vee b$	$\Leftrightarrow$	$a \Rightarrow b$
true	true	false	**true**		**true**
true	false	false	**false**	$\equiv$	**false**
false	true	true	**true**		**true**
false	false	true	**true**		**true**

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**

Propositional Logic and KB-Agents

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

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)?

(Automated) Proof System

In AI we are interested in taking existing knowledge (sentences in KB) and from that:

- **deriving new knowledge (new sentences)**
- **answering questions (query sentences)**

In Propositional Logic this means showing that some sentence Q follows from a Knowledge Base KB

where:

- Q **- some query sentence**
- KB **- knowledge base (a sentence made of sentences)**

Inference: Real-life Example

If it is raining, I will need an umbrella. It is raining. Therefore, I will need an umbrella.

Inference: Real-life Example

If it is raining, then I will need an umbrella. It is raining. Therefore, I will need an umbrella.

Inference: Real-life Example

If it is raining, then I will need an umbrella. It is raining. Therefore, I will need an umbrella.

Propositional Logic: An Argument

An argument A in propositional logic has the following form:

$$
\begin{array}{ll}
\mathrm{A:} & \mathrm{P1} \\
 & \mathrm{P2} \\
 & \ldots \\
 & \mathrm{PN} \\
\hline
 & \therefore\ \mathrm{C}
\end{array}
$$

PREMISES

CONCLUSION

An argument A is said to be valid if the implication formed by taking the conjunction of the premiseses (antecedent) and the conclusion C (consequent),

$(\mathrm{P1} \wedge \mathrm{P2} \wedge \mathrm{P3} \wedge \ldots \wedge \mathrm{PN}) \Rightarrow \mathrm{C}$ **is a tautology.**

Propositional Logic: An Argument

An argument A in propositional logic has the following form:

$$
\begin{array}{ll}
A: & P1 \\
 & P2 \\
 & \ldots \\
 & PN \\
\hline
 & \therefore\ C
\end{array}
$$

PREMISES

CONCLUSION

Premises are taken for granted (assumed to be true).

Inference: Real-life Example

If it is raining, then I will need an umbrella.

It is raining.

Therefore, I will need an umbrella.

Inference: Real-life Example

If it is raining, then I will need an umbrella.

It is raining. PREMISES

Therefore, I will need an umbrella. CONCLUSION

Inference: Real-life Example

If ______, then ____________. ← PREMISES

__________.

Therefore, ____________. ← CONCLUSION

Inference: Real-life Example

If __p__, then ______q______.

_____p_____. ← PREMISES

Therefore, ______q_____. ← CONCLUSION

p = “It is raining.”

q = “I will need an umbrella.”

PREMISE1 = “If it is raining, then I will need an umbrella.”

PREMISE2 = “It is raining.”

CONCLUSION = “I will need an umbrella.”

Inference: Real-life Example

If p then q is true

and p is true. PREMISES

Therefore, q is true. CONCLUSION

p = "It is raining."

q = "I will need an umbrella."

PREMISE1 = "If it is raining, then I will need an umbrella."

PREMISE2 = "It is raining."

CONCLUSION = "I will need an umbrella."

Inference Rules: Modus Ponens

$p \Rightarrow q$

p

PREMISES

$\therefore q$

CONCLUSION

p = "It is raining."

q = "I will need an umbrella."

PREMISE1 = $p \Rightarrow q$

PREMISE2 = p

CONCLUSION = q

Inference Rules: Modus Ponens

$p \Rightarrow q$

p

PREMISES

$\therefore q$

CONCLUSION

p = "It is raining."

q = "I will need an umbrella."

PREMISE1 = $p \Rightarrow q$

PREMISE2 = p

CONCLUSION = q

IF PREMISES ARE TRUE, THEREFORE THE CONCLUSION MUST ALSO BE TRUE

Inference: Modus Ponens

$p \Rightarrow q$ ← PREMISES

p

$\therefore q$ ← CONCLUSION

p = "It is raining."

q = "I will need an umbrella."

PREMISE1 = $p \Rightarrow q$

PREMISE2 = p

CONCLUSION = q

PROPOSITIONAL VARIABLES		IMPLICATION
p	q	$p \Rightarrow q$
true	true	true
true	false	false
false	true	true
false	false	true

Inference Rules: Modus Ponens

$p \Rightarrow q$

PREMISES

p

$\therefore q$

CONCLUSION

p = **"It is raining."**

q = **"I will need an umbrella."**

PREMISES = PREMISE1 AND PREMISE2 = $(p \Rightarrow q) \land p$

CONCLUSION = q

Inference Rules: Modus Ponens

$p \Rightarrow q$ ← PREMISES

p

$\therefore q$ ← CONCLUSION

PROPOSITIONAL VARIABLES		INDIVIDUAL PREMISE		PREMISES	CONCLUSION
p	q	P1: $p \Rightarrow q$	P2: p	$P1 \wedge P2{:}(p \Rightarrow q) \wedge p$	q
true	true	true	true	true	true
true	false	false	true	false	false
false	true	true	false	false	true
false	false	true	false	false	false

Inference Rules: Modus Ponens

$p \Rightarrow q$

p

PREMISES

$\therefore q$

CONCLUSION

PROPOSITIONAL VARIABLES		INDIVIDUAL PREMISE		PREMISES	CONCLUSION
p	q	P1: $p \Rightarrow q$	P2: p	$P1 \wedge P2{:}(p \Rightarrow q) \wedge p$	q
true	true	true	true	true	true
true	false	false	true	false	false
false	true	true	false	false	true
false	false	true	false	false	false

Inference Rules: Modus Ponens

$p \Rightarrow q$

p

PREMISES

$\therefore q$

CONCLUSION

IF **PREMISES** ARE **TRUE**, THEREFORE THE **CONCLUSION MUST ALSO BE TRUE**

INDIVIDUAL PREMISE		PREMISES	CONCLUSION
P1: $p \Rightarrow q$	P2: p	P1 $\wedge$ P2:$(p \Rightarrow q) \wedge p$	q
true	true	true	true
false	true	false	false
true	false	false	true
true	false	false	false

Inference Rules: Summary

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	$P \wedge Q$	$P \vee Q$ $\neg P \quad \vee$	$P \vee Q$ $\neg P \vee R$
$\therefore P \vee Q$	$\therefore P$	$\therefore Q$	$\therefore Q \vee R$

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

Argument Validity: Truth Table Proof

$p \Rightarrow q$

$p \Rightarrow \neg r$

$\neg p \Rightarrow \neg r$

$\therefore \neg r$

p	q	r	P1: $p \Rightarrow q$	P2: $q \Rightarrow \neg r$	P3: $\neg p \Rightarrow \neg r$	$P1 \wedge P2 \wedge P3$	$(P1 \wedge P2 \wedge P3) \Rightarrow \neg r$
true	true	true	true	false	true	false	**true**
true	true	false	true	true	true	true	**true**
true	false	true	false	true	true	false	**true**
true	false	false	false	true	true	false	**true**
false	true	true	true	false	false	false	**true**
false	true	false	true	true	true	true	**true**
false	false	true	true	true	false	false	**true**
false	false	false	true	true	true	true	**true**

Argument Validity: Truth Table Proof

$p \Rightarrow q$

$p \Rightarrow \neg r$

$\neg p \Rightarrow \neg r$

$\therefore \neg r$

$A \Leftrightarrow ((p \Rightarrow q) \wedge (p \Rightarrow \neg r) \wedge (\neg p \Rightarrow \neg r) \Rightarrow \neg r)$

An argument A is valid if it is a tautology.

p	q	r	P1: $p \Rightarrow q$	P2: $q \Rightarrow \neg r$	P3: $\neg p \Rightarrow \neg r$	$P1 \wedge P2 \wedge P3$	$(P1 \wedge P2 \wedge P3) \Rightarrow \neg r$
true	true	true	true	false	true	false	**true**
true	true	false	true	true	true	true	**true**
true	false	true	false	true	true	false	**true**
true	false	false	false	true	true	false	**true**
false	true	true	true	false	false	false	**true**
false	true	false	true	true	true	true	**true**
false	false	true	true	true	false	false	**true**
false	false	false	true	true	true	true	**true**

Argument Validity: Truth Table Proof

$p \Rightarrow q$

$p \Rightarrow \neg r$

$\neg p \Rightarrow \neg r$

$\therefore \neg r$

$A \Leftrightarrow ((P1) \wedge (P2) \wedge (P3) \Rightarrow \neg r)$

An argument A is valid if it is a tautology.

Argument A is valid, because it is a tautology

(always true regardless of p, q, r truth assignments)

p	q	r	$P1: p \Rightarrow q$	$P2: q \Rightarrow \neg r$	$P3: \neg p \Rightarrow \neg r$	$P1 \wedge P2 \wedge P3$	A
true	true	true	true	false	true	false	**true**
true	true	false	true	true	true	true	**true**
true	false	true	false	true	true	false	**true**
true	false	false	false	true	true	false	**true**
false	true	true	true	false	false	false	**true**
false	true	false	true	true	true	true	**true**
false	false	true	true	true	false	false	**true**
false	false	false	true	true	true	true	**true**

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:

$$\mathrm{KB} \models \mathrm{Q}$$

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

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:

$$KB \vDash Q \text{ if and only if } M(KB) \subseteq M(Q)$$

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

Logical Entailment with Truth Table

$$\boxed{\begin{array}{ll} p \Rightarrow q & \text{KB} \\ p \Rightarrow \neg r & \\ \neg p \Rightarrow \neg r & \end{array}}$$

$$\therefore \boxed{\neg r \quad Q}$$

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

$Q \Leftrightarrow \neg r$

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

KB

$p \Rightarrow q$

$p \Rightarrow \neg r$

$\neg p \Rightarrow \neg r$

$\therefore \neg r$ Q

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

Models where KB **is** true**:** M(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

KB

$p \Rightarrow q$

$p \Rightarrow \neg r$

$\neg p \Rightarrow \neg r$

$\therefore \neg r$ Q

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

Models where KB **is** true**:** M(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:$p \Rightarrow q$	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

$KB \Rightarrow Q$ is a Tautology Proof

$p \Rightarrow q$

$p \Rightarrow \neg r$

$\neg p \Rightarrow \neg r$

$\therefore \neg r$

KB ⇒ Q is true for ALL models / interpreations

KB ⇒ Q is a tautology

p	q	r	P1:p⇒q	P2:q⇒¬r	P3:¬ p ⇒ ¬ r	KB	KB ⇒ Q
true	true	true	true	false	true	false	**true**
true	true	false	true	true	true	true	**true**
true	false	true	false	true	true	false	**true**
true	false	false	false	true	true	false	**true**
false	true	true	true	false	false	false	**true**
false	true	false	true	true	true	true	**true**
false	false	true	true	true	false	false	**true**
false	false	false	true	true	true	true	**true**

Enumeration: Issues

Consider a complex sentence R built with N propositional variables p_1, p_2, p_3, ..., p_{N-1}, p_N and logical connectives ($\neg$, $\vee$, $\wedge$, $\Rightarrow$, $\Leftrightarrow$). Each truth assignment is a different possible world.

	N Propositional Variables						Complex sentence R	
	p_1	p_2	p_3	...	p_{N-1}	p_N		
2^N Possible Worlds (Models)	true	true	true	...	true	true	false	2^N Interpretations of R
	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	

Can we do better?
Can we automate the process?

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:

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

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