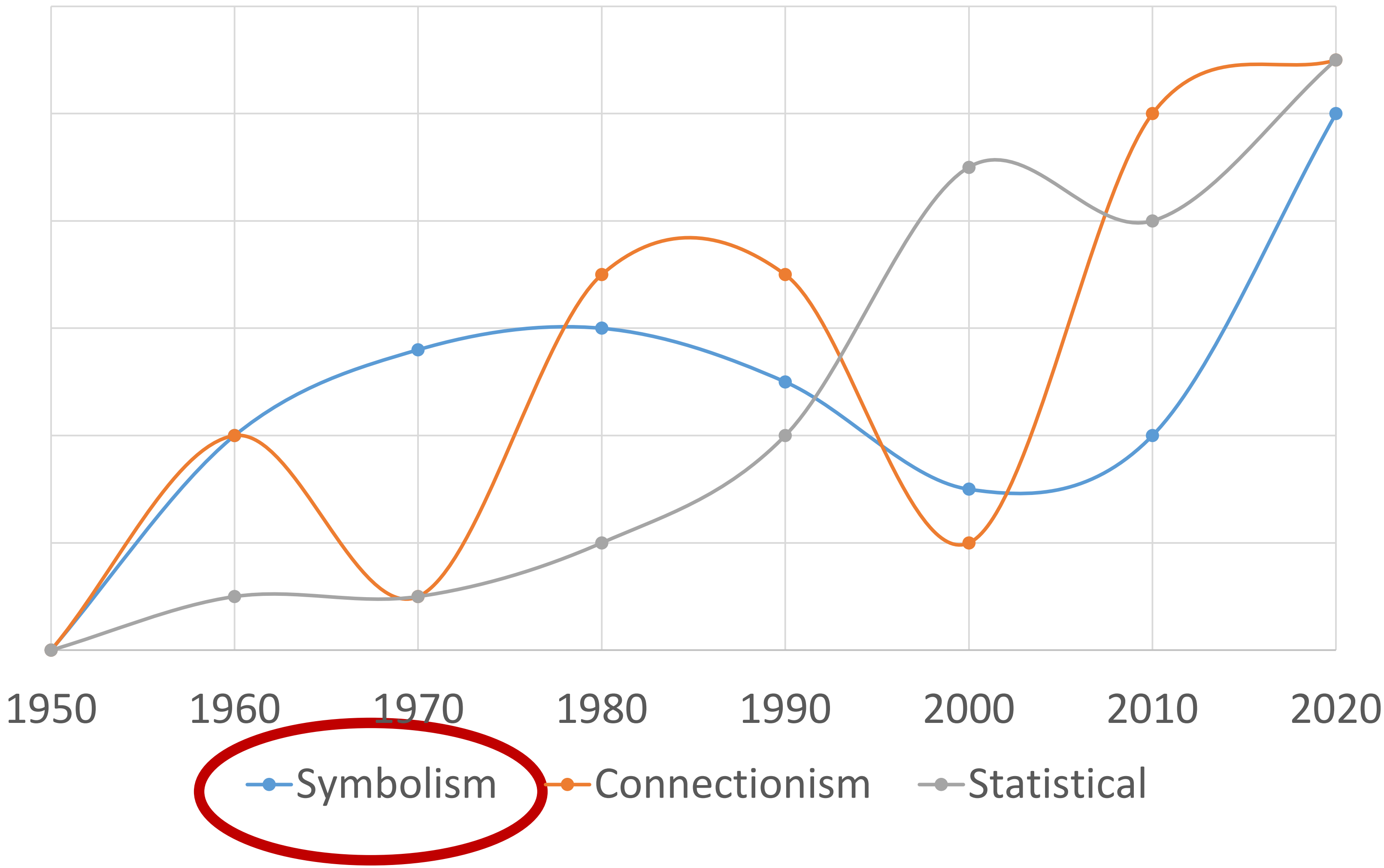
Three types of (strong) AI approaches



Propositional Logic

AIMA Chapter 7

Outline

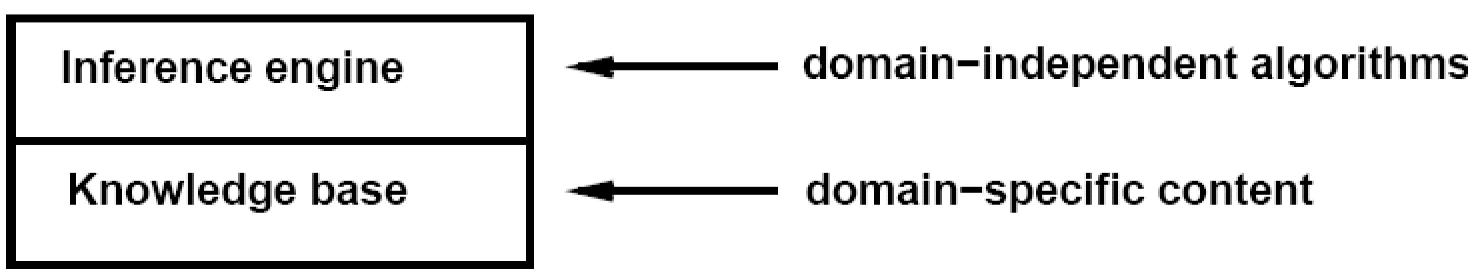
- Logic
- Propositional logic
  - Syntax
  - Semantics
  - Inference
- Horn logic
  - Inference
- An example application

Logic-based Symbolic AI

- Logic
  - Formal language in which knowledge can be expressed
  - A means of carrying out reasoning in the language

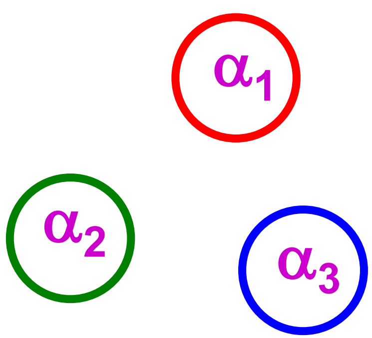
Logic-based Symbolic AI

- Logic (Knowledge-Based) AI
  - Knowledge base
    - set of sentences in a formal language to represent knowledge about the “world”
  - Inference engine
    - answers any answerable question following the knowledge base

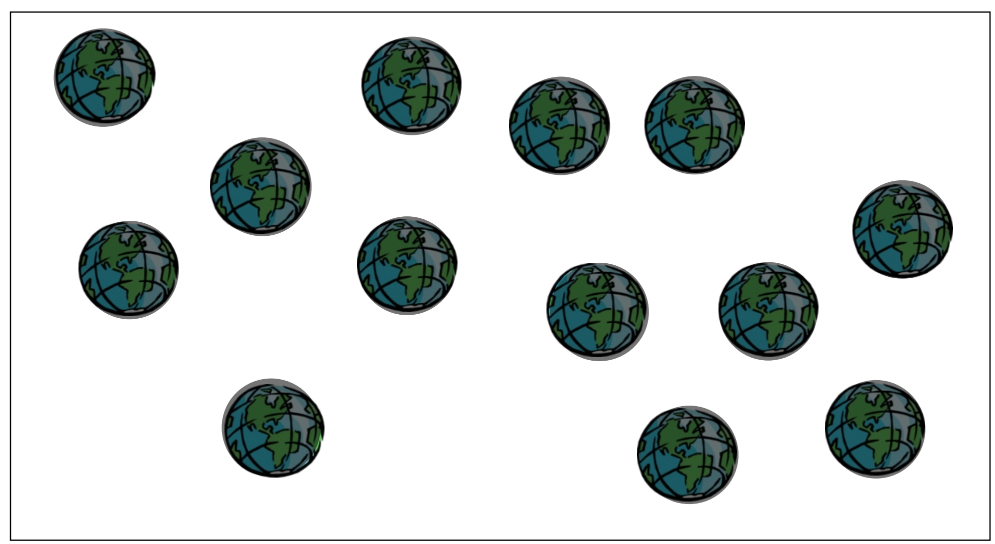


Formal Language

- Components of a formal language in a logic
  - Syntax: What sentences are allowed?
  - Semantics:
    - Which sentences are true/false in each model (possible world)?



Syntaxland



Semanticsland

Formal Language

- Example: the language of arithmetic
  - Syntax
    - $x+2 \geq y$ is a sentence
    - x2+y > {} is not a sentence
  - Semantics
    - $x+2 \geq y$ is true in a world where $x = 7$, $y = 1$
    - $x+2 \geq y$ is false in a world where $x = 0$, $y = 6$

Propositional Logic

Propositional logic: Syntax

- Propositional logic is the "simplest" logic
  - The proposition symbols P1, P2, etc. are sentences
  - If S is a sentence, $\neg$S is a sentence (negation)
  - If S1 and S2 are sentences, S1 $\wedge$ S2 is a sentence (conjunction)
  - If S1 and S2 are sentences, S1 $\vee$ S2 is a sentence (disjunction)
  - If S1 and S2 are sentences, S1 $\Rightarrow$ S2 is a sentence (implication)
  - If S1 and S2 are sentences, S1 $\Leftrightarrow$ S2 is a sentence (biconditional)

$\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$ are called logic connectives or operators

Sometimes $\rightarrow$ and $\leftrightarrow$ are used

Examples of PL sentences

- P means “It is hot.”
- Q means “It is humid.”
- R means “It is raining.”
- $(P \wedge Q) \Rightarrow R$
  - “If it is hot and humid, then it is raining”
- $Q \Rightarrow P$
  - “If it is humid, then it is hot”

Propositional logic: Semantics

- Each model specifies true/false for each proposition symbol
  - E.g. P_1 P_2 P_3

P_1	P_2	P_3
false	true	false

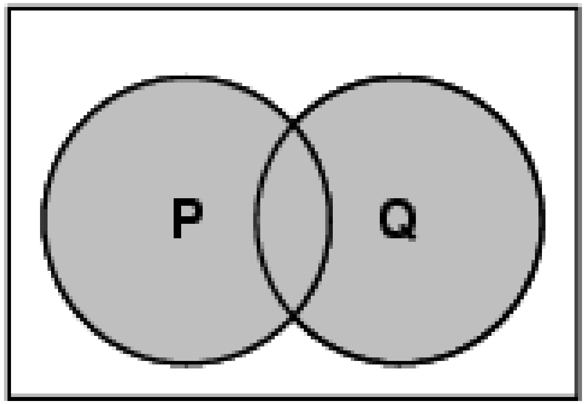
- Rules for evaluating truth with respect to a model m:
  - $\neg$S is true iff S is false
  - S1 $\wedge$ S2 is true iff S1 is true and S2 is true
  - S1 $\vee$ S2 is true iff S1 is true or S2 is true
  - S1 $\Rightarrow$ S2 is true iff S1 is false or S2 is true
  - S1 $\Leftrightarrow$ S2 is true iff S1$\Rightarrow$S2 is true and S2$\Rightarrow$S1 is true

Truth tables for connectives

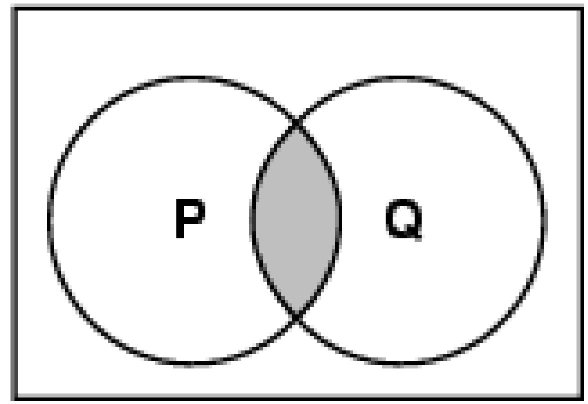
P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
$false$	$false$	$true$	$false$	$false$	$true$	$true$
$false$	$true$	$true$	$false$	$true$	$true$	$false$
$true$	$false$	$false$	$false$	$true$	$false$	$false$
$true$	$true$	$false$	$true$	$true$	$true$	$true$

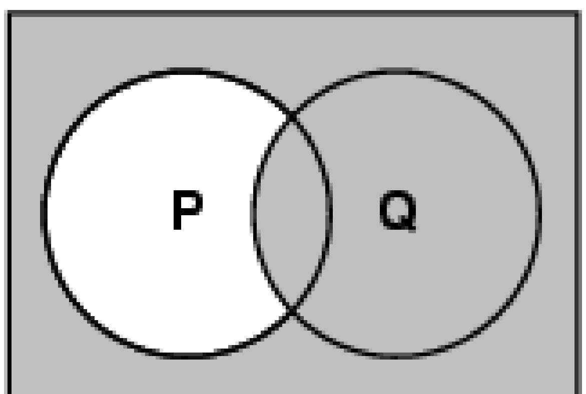
Venn Diagrams

$P \vee Q$

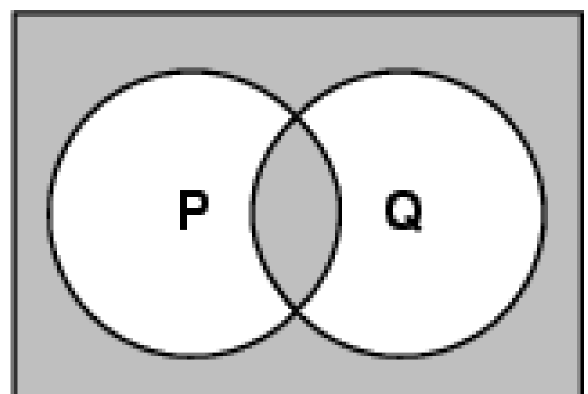


$P \wedge Q$





$P \Rightarrow Q$



$P \Leftrightarrow Q$

Material Implication

- S1 $\Rightarrow$ S2 is true iff S1 is false or S2 is true
- Given the following propositions, is “S1 $\Rightarrow$ S2” true?
  - S1 means “the moon is made of green cheese”
  - S2 means “the world is coming to an end”

- Material implication does not capture the meaning of “if... then”.
- See “Paradoxes of material implication” in Wikipedia

Logical equivalence

- Two sentences are logically equivalent iff true in same models

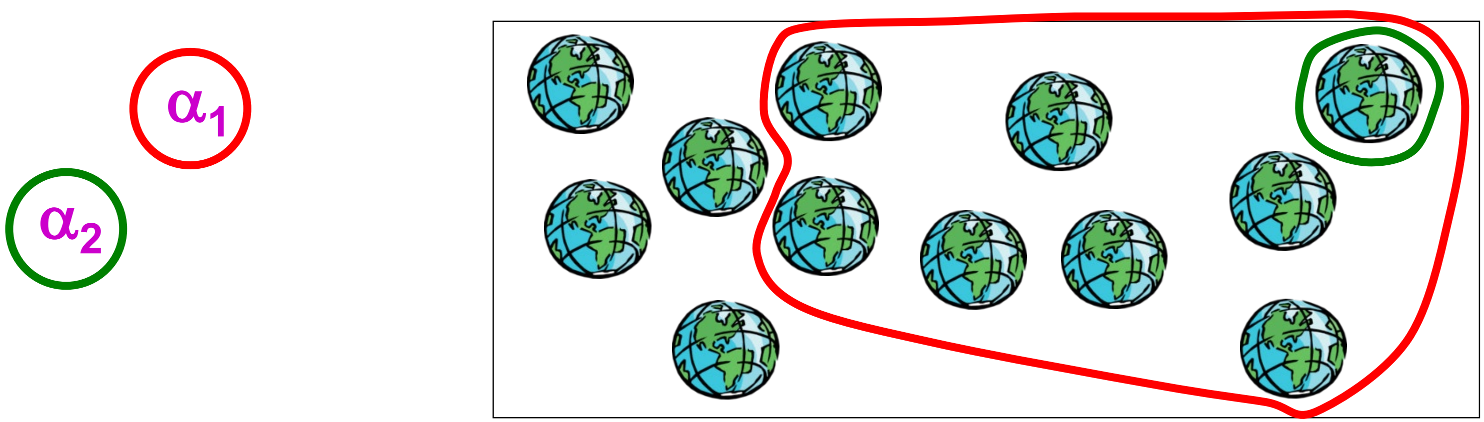
$$
\begin{array}{rcll}
(\alpha \wedge \beta) & \equiv & (\beta \wedge \alpha) & \text{commutativity of } \wedge \\
(\alpha \vee \beta) & \equiv & (\beta \vee \alpha) & \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) & \equiv & (\alpha \wedge (\beta \wedge \gamma)) & \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) & \equiv & (\alpha \vee (\beta \vee \gamma)) & \text{associativity of } \vee \\
\neg(\neg\alpha) & \equiv & \alpha & \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\beta \Rightarrow \neg\alpha) & \text{contraposition} \\
(\alpha \Rightarrow \beta) & \equiv & (\neg\alpha \vee \beta) & \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) & \equiv & ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) & \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) & \equiv & (\neg\alpha \vee \neg\beta) & \text{de Morgan} \\
\neg(\alpha \vee \beta) & \equiv & (\neg\alpha \wedge \neg\beta) & \text{de Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) & \equiv & ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) & \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) & \equiv & ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) & \text{distributivity of } \vee \text{ over } \wedge
\end{array}
$$

Validity and satisfiability

- A sentence is valid if it is true in all models
  - e.g., $A \vee \neg A$, $A \Rightarrow A$, $(A \wedge (A \Rightarrow B)) \Rightarrow B$
- A sentence is satisfiable if it is true in some model
  - e.g., $A \vee B$, C
- A sentence is unsatisfiable if it is true in no models
  - e.g., $A \wedge \neg A$
- Obviously, S is valid iff. $\neg S$ is unsatisfiable

Inference: entailment

- Entailment: $\alpha \models \beta$ ("α entails β" or "β follows from α") means in every world where α is true, β is also true
  - i.e., the α-worlds are a subset of the β-worlds [models(α) $\subseteq$ models(β)]
- In the example, $\alpha 2 \models \alpha 1$



Inference: proof

- A proof (α |- β) is a demonstration of entailment from α to β
  - Method 1: model checking
    - Truth table enumeration to check if models(α) $\subseteq$ models(β)
    - Time complexity always exponential in n ☹

P1	**P2**	**…**	**Pn**	α	β
F	F	…	F	F	T
F	F	…	T	T	T
……					
T	T	…	F	T	T
T	T	…	T	F	F

Inference: proof

- A proof (α |- β) is a demonstration of entailment from α to β
  - Method 2: application of inference rules
    - Search for a finite sequence of sentences each of which is an axiom or follows from the preceding sentences by a rule of inference
    - Axiom: a sentence known to be true
    - Rule of inference: a function that takes one or more sentences (premises) and returns a sentence (conclusion)

Inference: soundness & completeness

- Sound inference
  - everything that can be proved is in fact entailed
- Complete inference
  - everything that is entailed can be proved

- Method 1 (enumeration) is obviously sound and complete
- For method 2 (applying inference rules), it is much less obvious
  - Example: arithmetic is found to be not complete! (Gödel's theorem, 1931)

Quiz

- What's the connection between complete inference algorithms and complete search algorithms?
- Answer 1: they both have the words "complete…algorithm"
- Answer 2: Formulate inference $\alpha \mid\!- \beta$ as a search problem
  - Initial state: KB contains α
  - Actions: apply any inference rule that matches KB, add conclusion
  - Goal test: KB contains β

  Hence any complete search algorithm can be used to produce a complete inference algorithm

Resolution: an inference rule in PL

- Conjunctive Normal Form (CNF)
  - conjunction of disjunctions of literals (clauses)
  - Ex
    - $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$
    - $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

Conversion to CNF

$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

1. Eliminate $\Leftrightarrow$, replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.

   $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$

2. Eliminate $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$.

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$

3. Move $\neg$ inwards using de Morgan's rules and double-negation:

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$

4. Apply distributivity law ($\wedge$ over $\vee$) and flatten:

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

Resolution: an inference rule in PL

- Resolution inference rule (for CNF):

  Suppose l_i is $\neg m_j$

$$\frac{l_1 \vee \ldots \vee l_k, \qquad\qquad m_1 \vee \ldots \vee m_n}{l_1 \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k \vee m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n}$$

  Examples:

$$\frac{P_{1,3} \vee P_{2,2}, \qquad P_{2,3} \vee \neg P_{2,2}}{P_{1,3} \vee P_{2,3}} \qquad\qquad \frac{P_1, \neg P_1}{\{\}}$$

- Resolution is sound and complete for propositional logic

Resolution algorithm

- The best way to prove $KB \models \alpha$?
  - Proof by contradiction, i.e., show $KB \wedge \neg\alpha$ is unsatisfiable
  1. Convert $KB \wedge \neg\alpha$ to CNF
  2. Repeatedly apply the resolution rule to add new clauses, until one of the two things happens
     a) Two clauses resolve to yield the empty clause, in which case KB entails α
     b) There is no new clause that can be added, in which case KB does not entail α

Resolution example

$KB = (B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})) \land \neg B_{1,1}$

$\alpha = \neg P_{1,2}$