MATHEMATICS
FOR
INFORMATION PROCESSING

Derek G. Bridge
(Office: V/X034; Email: dgb)

Autumn term 1996-97

Course arrangements

- Taught to:
  - B.A./B.Sc. in ITBML
  - M.Sc.I.P.
- Contact time:
  - *Lectures*
  - *Problem class*
  - *Surgery hour*
- Examination:
  - *Three $\times$ 30 min. mock exams*
  - *One $\times$ 90-minute exam*

Course arrangements, cont'd.

- MIP provides prerequisite material for many courses
- Course materials:
  - *'Expanded slides'*
  - *Problem sheets and model answers*
  - *'Examination Notes'*
  - *Books*
- How to study maths:
  - Read ahead
  - Summarise the 'Expanded slides', learning the essentials by heart
  - Do lots of exercises; then do them again

Models

> *A model is an artifact that represents a part of reality at some level of abstraction that can be used to explain or predict features of the modelled reality.*

Example:

> A map of some terrain is a model of that terrain.

Models, cont'd.

Three sets of rules

- **Formulation:** the constructs you can use and how to fit them together.

- **Evaluation:** the way you interpret your model in terms of the modelled reality.

- **Manipulation:** the reasoning that you can carry out on the model to give you new information about the modelled reality.

The formulation and evaluation rules together define a **language**

Models: familiar examples

- Maps, plans, blueprints, even drawings, paintings and photos.

- Wind tunnels, reduce-sized replicas of engines, prototypes of devices.

- Business games.

- Computer simulations of economies, of brains, of meteorological conditions, of Post Office queues and of warehouse stock re-ordering.

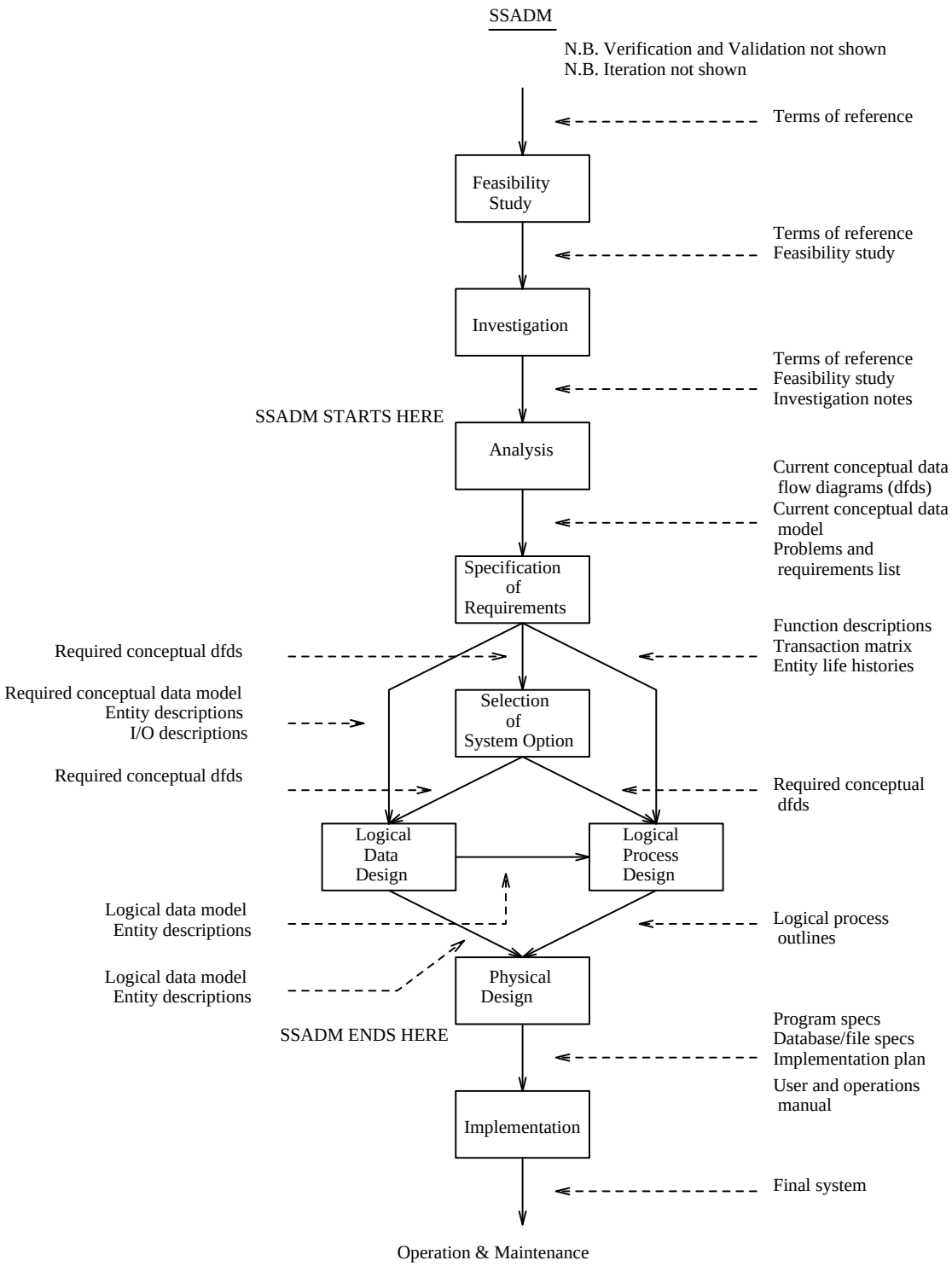
Of particular interest to us are those in which ‘symbols’ represent objects in the modelled reality.

Models: computerised information system development

A typical information systems development methodology involves a series of stages, each of which constructs a model.

These models are used

- to confirm understanding
- to analyse problems and propose solutions
- to aid the development of the next model


SSADM
N.B. Verification and Validation not shown
N.B. Iteration not shown
Terms of reference
Feasibility Study
Terms of reference
Feasibility study
Investigation
Terms of reference
Feasibility study
Investigation notes
SSADM STARTS HERE
Analysis
Current conceptual data flow diagrams (dfds)
Current conceptual data model
Problems and requirements list
Specification of Requirements
Function descriptions
Transaction matrix
Entity life histories
Required conceptual dfds
Required conceptual data model
Entity descriptions
I/O descriptions
Selection of System Option
Required conceptual dfds
Required conceptual dfds
Logical Data Design
Logical Process Design
Logical data model
Entity descriptions
Logical process outlines
Logical data model
Entity descriptions
Physical Design
SSADM ENDS HERE
Program specs
Database/file specs
Implementation plan
User and operations manual
Implementation
Final system
Operation & Maintenance

Models: computerised information systems development, cont'd.

In the development of (computerised) information systems, a failure to meet the customer's requirements (e.g. through a misunderstanding) can result in user dissatisfaction with the final system and even a final system that endangers life.

But, making changes becomes more costly as the development process proceeds.

One key to a solution then is a better characterisation of customer requirements from the outset.

We need models that are complete, consistent, non-redundant, unambiguous and precise

What language should we use to achieve this?

Models: a choice of language

- Using natural languages such as English:
  - expressive, require no special training
  - but ambiguous and vague
- Using diagrammatic notations:
  - easy to understand
  - but lack expressiveness, (often) lack a precise unambiguous meaning
- Using mathematics:
  - unambiguous and precise
  - but less expressive than natural languages and requires courses such as this one to educate its potential users.

Formal methods, formal languages and formal systems

Formal methods: sensible use of mathematics to assist in the process of information systems development.

Formal systems:

- **Formal languages:**
  - *Syntax:* precise formulation rules
  - *Semantics:* precise evaluation rules
- **Calculi:** one or more sets of precise manipulation rules (*deductive apparatus*)

Language and metalanguage

- **Object language:** the language under discussion

- **Metalanguage:** the language in which the object language is discussed

We will use English as the metalanguage for describing our object languages (which are various formal languages from parts of maths).

Conclusions

A firm foundation in mathematics is the hallmark of a professional and an excellent defence against technological obsolescence.

EXPRESSIONS
AND
VALUES

Expressions

Expressions are linguistic objects.

To study them, we must consider:

- *Formulation* of expressions (*syntax*).
- *Evaluation* of expressions (*semantics*).
- *Manipulation* of expressions (*calculus*).

We consider these for **arithmetic expressions**.

Formulation

1. The simplest expressions (for now) comprise just a **numeral**:

   $$0, 1, 2, 3, \ldots, 10.2, 10.34$$

2. More complex expressions comprise **operators** and their **operands** (or **arguments**), where the operands are themselves expressions.

   To build expressions using these, we need to know the **arity** and **fixity** of each operator.

3. Any expression may be enclosed in parentheses and will still be an expression.

4. No other string of symbols is an arithmetic expression.

Formulation: operator arity

The **arity** of an operator is the number of operands it requires.

- Unary (or 1-ary) operators
  - e.g. $\sin, \cos, \tan, !$
  - require one operand, e.g. $\sin(90), 3!$
- Binary (or 2-ary) operators
  - e.g. $+, -, *, /, \uparrow, =, <, \leq, >, \geq, \neq, \not<, \not\leq, \not>, \not\geq$
  - require two operands, e.g. $2 + 3, 2 \uparrow 3$
- Ternary (3-ary) operators, quaternary (4-ary) operators, etc. are also possible

Formulation: operator fixity

The **fixity** of an operator determines the placement of its operands.

- **Prefix** operators, e.g. $\sin, \cos, \tan$
  - precede their operands, e.g. $\sin(90)$
- **Infix** operators
  - e.g. $+, -, *, /, \uparrow, =, <, \leq, >, \geq, \neq, \not<, \not\leq, \not>, \not\geq$
  - come between their operands, e.g. $2 + 3$, $2 * 3$, $2 \uparrow 3$
- **Postfix** operators, e.g. $!$
  - come after their operand(s), e.g. $3!$

Evaluation: simple expressions

In English, the simplest expressions are names; they evaluate to people, dogs, towns, etc.

Sweep

We also use names in maths.

- π the number **3.141592…**
- 5 the number **5**

Evaluation: more complex expressions

Operators are symbols that denote **operations**.

- To evaluate an expression that contains an operator, we work out the values of the operator's operands, we work out what operation the operator denotes, and we apply this operation to the values of the operands; the resulting object is the value of the whole expression.

There are lots of examples in English, e.g.

the_best_friend_of Sooty

A maths example:

$2 + 5$ the number 7

Evaluation: denotations of operators

$+$	Addition of numbers.
$-$	Subtraction of the second number from the first.
$*$	Multiplication of numbers.
$/$	Division of the first number by the second.
$\uparrow$	Exponentiation.
$\mathtt{sin}, \mathtt{cos}, \mathtt{tan}$	Sine, cosine and tangent from Geometry.
$!$	Factorial.

Evaluation: denotations of operators, cont'd.

Exponentiation

$m \uparrow n$ is more commonly written m^n in maths.

The expressions $m \uparrow n$ or m^n are read 'm to the power n' or 'm raised to the power n'.

$$m^n \quad = m \uparrow n \quad = m * m * m * \ldots * m$$
(with m occurring n times)

It is taken that *any* number, m, raised to the power zero, $m \uparrow 0$ or m^0, evaluates to the number 1.

Factorial

The expression $n!$ is read 'n factorial'.

$$n! = n * (n-1) * (n-2) * \ldots * 2 * 1$$

It is taken that $0!$ evaluates to the number 1.

Evaluation: denotations of operations, cont'd.

$=$	Equality.
$\neq$	Not equal.
$<, \not\geq$	The first number is less than the second.
$\leq, \not>$	The first number is less than or equal to the second.
$>, \not\leq$	The first number is greater than the second.
$\geq, \not<$	The first number is greater than or equal to the second.

Evaluation: types

Values fall into classes of similar kinds of objects, these classes being called **types**.

An expression containing an operation can only be evaluated if the operands denote objects of the types required by the operation.

The expression as a whole will return a value of a certain type.

So, in this course,

$$3 * (4 > 2)$$

is syntactically well-formed but cannot be evaluated because of a *type violation*.

Evaluation: denoting the same object

In English, several names or more complex expressions can denote the same object:

Derek
DGB
the_lecturer_of
MIP

Similarly in maths, several expressions (whether simple or complex) can denote the same object:

5	five	
cinq	funf	
V	101	
$\text{\textbar\textbar\textbar\textbar}$	$2+3$	the number 5
$5+0$	$6-1$	
$(2\uparrow 2)+1$	$((2*4)+2)/(3-1)$	

Evaluation: ambiguity

Consider

$$2 + 3 * 4$$

This evaluates either to the number 20 or to the number 14.

Consider also

$$5 - 2 + 3$$

This evaluates either to the number 6 or to the number 0.

Even though this was supposed to be a formal language we have AMBIGUITY!

Evaluation: using parentheses

Parentheses provide one way to disambiguate. They make the evaluation order explicit:

$(2 + 3) * 4$	evaluates to the number 20
$2 + (3 * 4)$	evaluates to the number 14
$(5 - 2) + 3$	evaluates to the number 6
$5 - (2 + 3)$	evaluates to the number 0
$(2 + 3) * (4 \uparrow 2)$	evaluates to the number 80
$(2 + (3 * 4)) \uparrow 2$	evaluates to the number 196
$2 + ((3 * 4) \uparrow 2)$	evaluates to the number 146

Evaluation: precedence and associativity

For arithmetic, the conventional **operator precedences** are:

$\uparrow$	highest precedence
$*, /$	
$+, -$	
$=, <, \leq, >, \geq, \neq, \not<, \not\leq, \not>, \not\geq$	lowest precedence

So

$$2 + 3 * 4$$

is evaluated as

$$2 + (3 * 4)$$

because $*$ has higher precedence than $+$.

Evaluation: precedence and associativity, cont'd.

For arithmetic, the conventional **operator associativities** are:

Associate to the left	Associate to the right
$*, /, +, -,$	$\uparrow$
$=, <, \leq, >, \geq, \neq, \not<, \not\leq, \not>, \not\geq$	

So

$$5 - 2 + 3$$

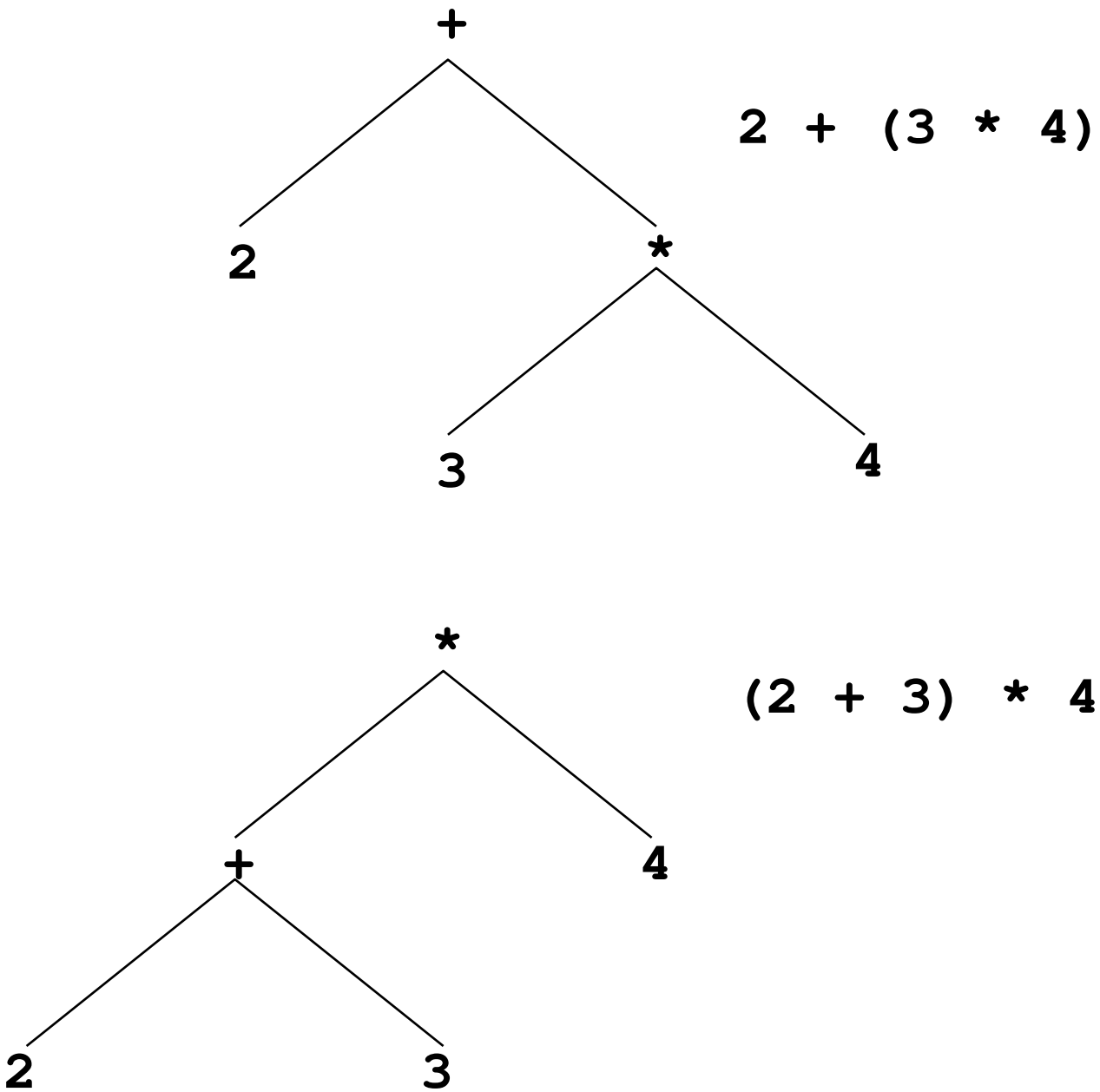
is evaluated as

$$(5 - 2) + 3$$

because $+$ and $-$ associate to the left.

You override the precedence and associativity rules with parentheses.

Evaluation: trees



Definitions: new names

Definitions allow us to associate a **name** with a value

- the new name might be more concise or more mnemonic than using a whole expression

- the definition **binds** the name to the value.

We state the new name and an expression whose value will be associated with the name.

$$
\begin{array}{rcl}
\mathit{No_of_Lectures} & \triangleq & 18 \\
\mathit{Lecture_Length} & \triangleq & 60 - 10 \\
\mathit{Contact_Time} & \triangleq & \mathit{No_of_Lectures} * \mathit{Lecture_Length}
\end{array}
$$

Now we can write expressions that use these new names.

$\mathit{Lecture_Length} + 10$	evaluates to the number 60
$\mathit{Contact_Time} * 12$	evaluates to the number 10800

Definitions: new names, cont'd.

Names can also be associated with operations.

In these definitions, we use **parameters**: (temporary) names that can be taken to stand for an arbitrary value to which the operation is applied.

Examples (x and y are the parameters)

$$
\begin{aligned}
\textit{double}(x) &\triangleq x + x \\
\textit{treble}(y) &\triangleq 3 * y \\
x \heartsuit y &\triangleq y - x \\
\textit{funnysubtract}(x, y) &\triangleq x \heartsuit y \\
x \spadesuit y &\triangleq (x \uparrow 2) + (y \uparrow 2) \\
\textit{sumdoubles}(x, y) &\triangleq \textit{double}(x) + \textit{double}(y) \\
\textit{sixtimes}(x) &\triangleq \textit{double}(\textit{treble}(x))
\end{aligned}
$$

Definitions : new names, cont'd.

Further examples

$$\textit{positive}(x) \triangleq x > 0$$
$$\textit{negative}(x) \triangleq x < 0$$

The new operators can be used in the formulation of expressions; the operations to which they are bound are used in the evaluation of those expressions.

2♡3	evaluates to 1
positive(double(2) + treble(2))	evaluates to true

Formulation (revised)

1. The simplest expressions comprise just a **numeral** or a **name** that we define using $\stackrel{\triangle}{=}$.

2. More complex expressions comprise **operators** and their **operands** (or **arguments**), where the operands are themselves expressions.

   The operators we use are

   $$+, -, *, /, \uparrow, \sin, \cos, \tan, !,$$

   $$=, <, \leq, >, \geq, \neq, \not<, \not\leq, \not>, \not\geq$$

   and any we define using $\stackrel{\triangle}{=}$.

3. Any expression may be enclosed in parentheses and will still be an expression.

4. No other string of symbols is an arithmetic expression.

Definitions: occurrences of names

We have introduced new names in two ways:

- the 'main' name in a definition using $\triangleq$

- the parameters in the definition of new operators

Both of these occurrences of names are said to be **binding occurrences** of the name.

Other occurrences of names are called **applied occurrences**.

Definitions: scope

The textual context of a definition is known as the **scope** of that definition.

- In the case of parameters, the scope is limited to the right hand side of the definition.

- In the case of the main name introduced by a definition, the scope is usually some obvious section of subsequent text.

An applied occurrence of a name is **bound** if the applied occurrence falls within the scope of a binding occurrence for that name.

An applied occurrence of a name is **free** if the applied occurrence does not fall within the scope of a binding occurrence for that name.

We shall usually only write expressions in which every name is bound.

Definitions: scope examples

On this overhead,

$$\begin{aligned} \textit{DayLength} &\triangleq 24 \\ \textit{halve}(x) &\triangleq x/2 \\ \textit{quarter}(x) &\triangleq x * 0.25 \\ \textit{eighth}(x) &\triangleq \textit{halve}(\textit{quarter}(y)) \end{aligned}$$

Here are expressions for evaluation:

$$\textit{halve}(\textit{DayLength}) + 10$$

$$\textit{quarter}(z)$$

$$\textit{even}(32)$$

Manipulation

Manipulation is transforming one expression into another expression that has a significant semantic relationship with the original expression.

Crucially, in manipulation the transformations are done without reference to the meanings (values) of the expressions involved

We will do manipulation with reference to **laws**.

Manipulation: laws

An example of a **law**:

For all arithmetic expressions AE_1 and AE_2,

$$AE_1 + AE_2 = AE_2 + AE_1$$

This law acts as a template for an (infinite) number of true statements of equality. It says all of the following (and more) all in one go:

$$0 + 1 = 1 + 0 \qquad 5 + 4 = 4 + 5$$
$$2 + 3 = 3 + 2 \quad (2 * 3) + 4 = 4 + (2 * 3)$$

This law allows us to swap the order of the operands to an addition operator in any expression or subexpression.

Manipulation: laws of the algebra of arithmetic

For brevity and simplicity, we will only present laws that pertain to addition and multiplication.

For all arithmetic expressions AE_1, AE_2 and AE_3,...

Commutativity of $+$ and $*$

$$AE_1 + AE_2 = AE_2 + AE_1$$

$$AE_1 * AE_2 = AE_2 * AE_1$$

Associativity of $+$ and $*$

$$AE_1 + (AE_2 + AE_3) = (AE_1 + AE_2) + AE_3$$

$$AE_1 * (AE_2 * AE_3) = (AE_1 * AE_2) * AE_3$$

Manipulation: laws of the algebra of arithmetic, cont'd.

For all arithmetic expressions AE, AE_1, AE_2 and AE_3...

Distributivity of $*$ over $+$

$$AE_1 * (AE_2 + AE_3) = (AE_1 * AE_2) + (AE_1 * AE_3)$$

Identities

$$0 + AE = AE$$

$$1 * AE = AE$$

Manipulation: using the laws in proofs

We can use laws to prove that two expressions are equal.

Example: to show that for all arithmetic expressions E_1 and E_2,

$$E_1 * (E_2 + 1) = (E_1 * E_2) + E_1$$

Take the left hand side (LHS) and manipulate it until it is the same as the right hand side (RHS):

$$\begin{aligned} \text{LHS} &= E_1 * (E_2 + 1) \\ &= (E_1 * E_2) + (E_1 * 1) \\ &= (E_1 * E_2) + (1 * E_1) \\ &= (E_1 * E_2) + E_1 \\ &= \text{RHS} \end{aligned}$$

Manipulation: another example

Show that for all arithmetic expressions E_1, E_2, E_3 and E_4,

$$E_1 + (E_2 + (E_3 + E_4)) = \\ E_4 + (E_3 + (E_2 + E_1))$$

Take the left hand side (LHS) and manipulate it until it is the same as the right hand side (RHS):

$$\begin{aligned} \text{LHS} &= E_1 + (E_2 + (E_3 + E_4)) \\ &= (E_1 + E_2) + (E_3 + E_4) \\ &= (E_2 + E_1) + (E_3 + E_4) \\ &= ((E_2 + E_1) + E_3) + E_4 \\ &= (E_3 + (E_2 + E_1)) + E_4 \\ &= E_4 + (E_3 + (E_2 + E_1)) \\ &= \text{RHS} \end{aligned}$$

Manipulation: another example

Show that for all arithmetic expressions E_1, E_2, E_3 and E_4,

$$E_1 * (E_2 + (E_3 + E_4)) = \\ ((E_1 * E_2) + (E_1 * E_3)) + (E_1 * E_4)$$

Take the left hand side (LHS) and manipulate it until it is the same as the right hand side (RHS):

$$\begin{aligned} \text{LHS} &= E_1 * (E_2 + (E_3 + E_4)) \\ &= (E_1 * E_2) + (E_1 * (E_3 + E_4)) \\ &= (E_1 * E_2) + ((E_1 * E_3) + (E_1 * E_4)) \\ &= ((E_1 * E_2) + (E_1 * E_3)) + (E_1 * E_4) \\ &= \text{RHS} \end{aligned}$$

SET THEORY

Sets: what are they?

A **set** is a collection of objects.

- The objects can be any identifiable things of interest, whether physical or abstract.

- The objects in a set are called its **members** or **elements**.

- An object can belong to more than one set.

- The members of a set do not have to be the same kind of thing, though in information processing they usually are.

Simple expressions that denote sets

There are two ways to define a set.

Extensional definition: *enumerate* the members

- The set made up of the four numbers 5, 7, 10 and 103:

$$\{5, 7, 10, 103\}$$

- The vowels:

$$\{a, e, i, o, u\}$$

- Autumn term courses shared by 1st year ITBML and MScIP:

$$\{\textit{IPP, MIP}\}$$

Simple expressions that denote sets

Intensional definition: specify a property possessed by all the members and no other objects, the **defining property**.

- $\{x \mid x$ is a number and $x \geq 0$ and $x < 4\}$

- $\{l \mid l$ is a lecturer in the Comp. Sci. Dept. at the Uni. of York$\}$

- $\{c_1 \mid c_1$ is a city of a country c_2, where c_2 is a country in the set $\{$*France, Germany*$\}\}$

Two fundamental properties of sets

1. Each element appears only once.

2. The elements have no order.

The above two properties emphasise the importance of distinguishing an expression that defines a set from its value (as with expressions of arithmetic).

For example, the following different expressions all denote the same set:

$$\{2, 3, 4\} \qquad \{3, 2, 4\}$$
$$\{2, 2, 3, 4, 4, 4\} \qquad \{n \mid 2 \leq n \text{ and } 2 \leq 4\}$$
$$\{m \mid 2 \leq m \text{ and } 2 \leq 4\} \qquad \{n \mid 1 < n \text{ and } n < 5\}$$

Set theory: formulation

1. The simplest expressions are extensional and intensional set definitions.

2. More complex expressions comprise **operators** and their **operands**, where the operands are themselves expressions.

$$=, \neq, \in, \notin, \subseteq, \subset, \nsubseteq, \not\subset, \supseteq, \supset, \not\supseteq, \not\supset, \cup, \cap, \setminus$$

$$\wp, \mathit{card}$$

$$'$$

3. Any expression may be enclosed in parentheses and will still be an expression.

4. No other string of symbols is an expression of set theory.

Definitions of new names

We can also name sets and new operations using $\triangleq$, and use the new names in expressions.

E.g.

$$\textit{DGB_Courses} \triangleq \{\textit{MIP, DEC, NLP}\}$$

Some names are already in common use:

- $\mathbb{Z}$ the **integers** (i.e. all 'whole' numbers)
- $\mathbb{N}$ the **natural numbers** (i.e. all integers ≥ 0)
- $\mathbb{P}$ the **positive integers** (i.e. all integers > 0)
- $\mathbb{Q}$ the **rationals**
- $\mathbb{R}$ the **reals**

Special sets

The **universal set**

- The set of all elements of interest to your present purpose.
- The universal set is usually written U.
- Sometimes it is defined explicitly; sometimes it is supposed to be ‘obvious’.

The **empty set**

- The empty set has no elements.
- The empty set is usually written $\{\}$ or $\emptyset$.

Set equality and set inequality

$=$ denotes the binary infix operation that returns true if its operands denote the same set and false otherwise.

$\neq$ denotes the binary infix operation that returns true if its operands denote different sets and false otherwise.

E.g.

$$\{2, 3, 4\} = \{n \mid 1 < n \text{ and } n < 5\}$$

$$\{t \mid t \text{ is a U.K. university town}\} \neq$$

$$\{c \mid c \text{ is a U.K. cathedral city}\}$$

Set membership

For any set A and any object a, either a is a member of A or it is not.

- To indicate that a is a member of A, we write

$$a \in A$$

- To indicate that a is not a member of A, we write

$$a \notin A$$

Examples

- $1 \in \{0, 1, 2\}$
- *York* $\notin \{c \mid c$ is a capital city$\}$

Subset

For any two sets A_1 and A_2, A_1 is **included in** A_2, or A_1 is a **subset** of A_2, if and only if for all a, if $a \in A_1$ then $a \in A_2$.

- To indicate that A_1 is a subset of A_2, we write

$$A_1 \subseteq A_2$$

- To indicate that A_1 is not a subset of A_2, we write

$$A_1 \not\subseteq A_2$$

Examples

- $\{0, 2\} \subseteq \{0, 1, 2\}$
- $\{0, 1, 3\} \not\subseteq \{0, 1, 2\}$
- $\{0, 1, 2\} \subseteq \{0, 1, 2\}$

Proper subset

For any two sets A_1 and A_2, A_1 is a **proper subset** of A_2 if and only if $A_1 \subseteq A_2$ and there exists a such that $a \in A_2$ but $a \notin A_1$.

We write $A_1 \subset A_2$ and $A_1 \not\subset A_2$.

Examples

- $\{0, 2\} \subset \{0, 1, 2\}$
- $\{$*Rome, Paris*$\} \subset \{c \mid c$ is a capital city$\}$
- $\{0, 1, 3\} \not\subset \{0, 1, 2\}$
- $\{0, 1, 2\} \not\subset \{0, 1, 2\}$
- $\mathbb{P} \subset \mathbb{N} \subset \mathbb{Z} \subset \mathbb{Q} \subset \mathbb{R}$

Set supersets and proper supersets

Superset

$$A_1 \supseteq A_2 \triangleq A_2 \subseteq A_1$$

Proper superset

$$A_1 \supset A_2 \triangleq A_2 \subset A_1$$

$\not\supseteq$ and $\not\supset$ are defined as you might expect.

Examples

- $\{0, 1, 2\} \supseteq \{0, 1\}$
- $\{0, 1, 2\} \supset \{0, 1\}$

Subsets: special cases

For any set A,

$$A \subseteq A$$

For any set A,

$$\emptyset \subseteq A$$

Set equality revisited

Using $\in$

For any two sets A_1 and A_2,

$A_1 = A_2$ if and only if for all a,

$$a \in A_1 \text{ if and only if } a \in A_2$$

Using $\subseteq$

For any two sets A_1 and A_2,

$A_1 = A_2$ if and only if

$$A_1 \subseteq A_2 \text{ and } A_2 \subseteq A_1$$

Intensional set definitions revisited

It is very common for an intensional set definition to take the following form:

$$\{x \mid x \in A \text{ and } P(x)\}$$

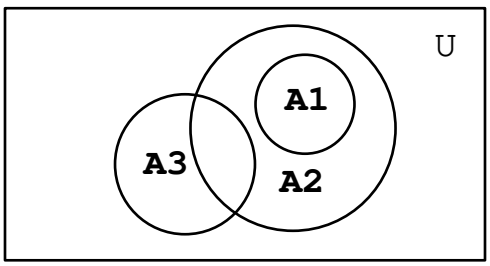
(This set will be that subset of A whose members satisfy property P.)

For this common form of definition, a shorthand is popular:

$$\{x \in A \mid P(x)\}$$

Venn diagrams

In a Venn diagram, a set, A, is represented by a circle, the members of A being the points that lie within the circle. The universal set, U, is portrayed by a rectangle that encloses the circles.



$$\begin{array}{lll} A_1 \subseteq A_2 & A_1 \not\subseteq A_3 & A_1 \subseteq U \\ A_2 \not\subseteq A_1 & A_2 \not\subseteq A_3 & A_2 \subseteq U \\ A_3 \not\subseteq A_1 & A_3 \not\subseteq A_2 & A_3 \subseteq U \end{array}$$

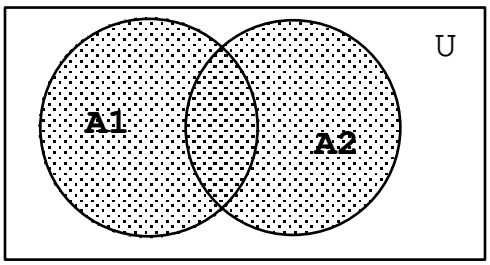
Set union

The **union** of two sets A_1 and A_2 is the set denoted by $A_1 \cup A_2$ and defined by the condition:

> for all a, $a \in A_1 \cup A_2$ if and only if $a \in A_1$ or $a \in A_2$

More formally,

$$A_1 \cup A_2 \stackrel{\triangle}{=} \{a \mid a \in A_1 \text{ or } a \in A_2\}$$



The shaded area is $A_1 \cup A_2$.

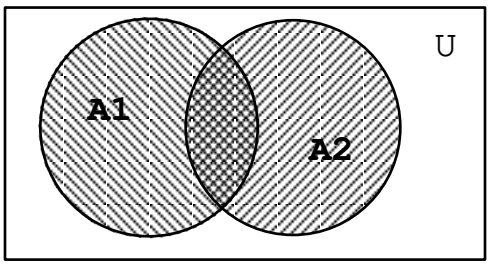
Set intersection

The **intersection** of two sets A_1 and A_2 is the set denoted by $A_1 \cap A_2$ and defined by the condition:

> for all a, $a \in A_1 \cap A_2$ if and only if $a \in A_1$ and $a \in A_2$

More formally,

$$A_1 \cap A_2 \stackrel{\triangle}{=} \{a \mid a \in A_1 \text{ and } a \in A_2\}$$



The dark shaded area in the middle is $A_1 \cap A_2$.

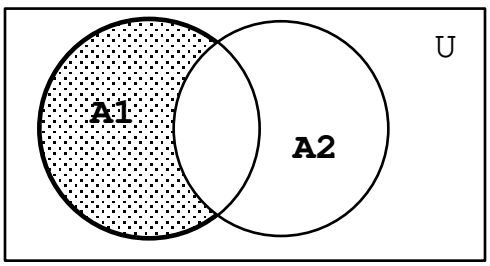
Set difference

The **difference** of two sets A_1 and A_2 is the set denoted by $A_1 \setminus A_2$ and defined by the condition:

> for all a, $a \in A_1 \setminus A_2$ if and only if $a \in A_1$ and $a \notin A_2$

More formally,

$$A_1 \setminus A_2 \stackrel{\triangle}{=} \{a \mid a \in A_1 \text{ and } a \notin A_2\}$$



The shaded area is $A_1 \setminus A_2$.

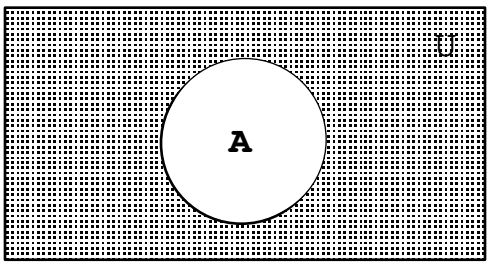
Set complement

The **complement** of a set A is the set denoted by A' and defined by the condition:

> for all a in U, $a \in A'$ if and only if $A \in U$ but $a \notin A$

More formally,

$$A' \stackrel{\triangle}{=} \{a \mid a \in U \text{ and } a \notin A\}$$



The shaded area is A'.

Class exercises

1. $U \triangleq \{n \mid n \in \mathbb{N} \text{ and } n > 4\}$

   Evaluate
   $(\{5, 6, 7\} \setminus \{7, 8\}) \subseteq \{x \mid x \in \mathbb{N} \text{ and } x > 10\}'$

2. $A \triangleq \{s \mid s \text{ was a student at York last year}\}$
   $B \triangleq \{s \mid s \text{ is a student who left York at the end of last year}\}$
   $C \triangleq \{s \mid s \text{ is a newly arrived York student}\}$

Write an expression that, if evaluated, would compute York's current students.

3. $A \triangleq \{s \mid s \text{ is a student who will take ADP}\}$
   $B \triangleq \{s \mid s \text{ is a student who will take BMG}\}$

Why might the timetabling officer evaluate

$$(A \cap B) = \emptyset$$

Sets as members of sets

A set may have other sets as its members.

$$\{1, \{2, 3\}, 3\}$$

$$\{\{1\}, \emptyset, \{1, 3\}\}$$

$$1 \neq \{1\} \neq \{\{1\}\} \neq \{\{\{1\}\}\}$$

$$\emptyset \neq \{\emptyset\} \neq \{\{\emptyset\}\} \neq \{\{\{\emptyset\}\}\}$$

In general, for any object a,

$$a \neq \{a\}$$

$$\{1\} \in \{\{1\}, \emptyset, \{1, 3\}\}$$

$$1 \notin \{\{1\}, \emptyset, \{1, 3\}\}$$

$$\{\{1\}\} \subseteq \{\{1\}, \emptyset, \{1, 3\}\}$$

Partitions

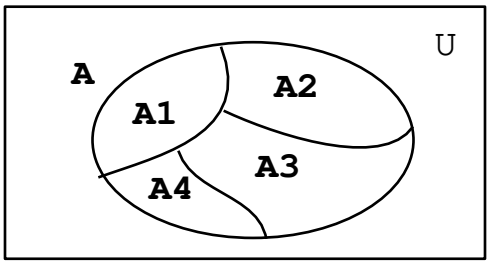
Let A be any non-empty set. A **partition** of A is a sub-division of all the members of A into disjoint, non-empty sets.

More formally, a partition of A is a set of non-empty sets $\{A_1, A_2, \ldots, A_n\}$ such that

(i) for any $a \in A$, $a \in A_i$ for some i

(ii) for all i and j, $i \neq j$, $A_i \cap A_j = \emptyset$

Diagrammatically,



Partitions: examples

Example: let $S \triangleq \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

One partition is:

$$\{\{1, 3, 5\}, \{2, 4, 6, 8\}, \{7, 9\}\}$$

Others are:

$$\{\{1, 5, 7\}, \{2, 4, 8\}, \{3, 6, 9\}\}$$

$$\{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}, \{8\}, \{9\}\}$$

$$\{\{1, 2, 3, 4, 5, 6, 7, 8, 9\}\}$$

Powerset

The **powerset** of a set A is the set denoted by $\wp(A)$ and defined by the condition:

for all a, $a \in \wp(A)$ if and only if $a \subseteq A$

More formally,

$$\wp(A) \stackrel{\triangle}{=} \{a \mid a \subseteq A\}$$

— the set of all subsets of A.

Set cardinality

The number of members of a set A is called the **cardinality** of A.

The cardinality of set A will be denoted *card(*A*)*.

Examples

$$\begin{aligned} \textit{card}(\{0,1,2\}) &= 3 \\ \textit{card}(\{3\}) &= 1 \\ \textit{card}(\emptyset) &= 0 \end{aligned}$$

Computing cardinalities

- For any finite, disjoint sets A_1 and A_2,

$$\mathit{card}(A_1 \cup A_2) = \mathit{card}(A_1) + \mathit{card}(A_2)$$

- For any finite sets A_1 and A_2 (not necessarily disjoint):

$$\mathit{card}(A_1 \cup A_2) =$$

$$(\mathit{card}(A_1) + \mathit{card}(A_2)) - \mathit{card}(A_1 \cap A_2)$$

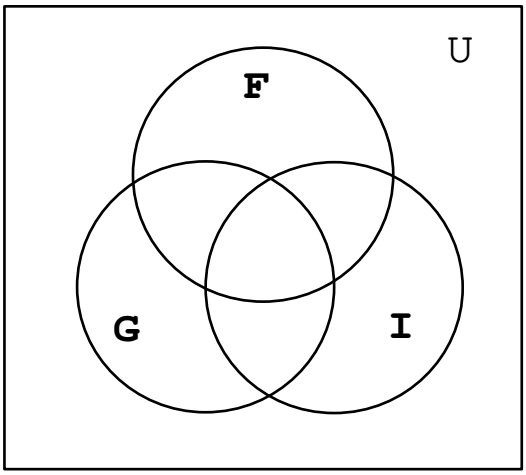
- For any finite set A,

$$\mathit{card}(\wp(A)) = 2 \uparrow \mathit{card}(A)$$

Computing cardinalities, cont'd

For any finite sets A_1, A_2 and A_3,

$$\begin{aligned}
&\mathit{card}((A_1 \cup A_2) \cup A_3) = \\
&\quad \mathit{card}(A_1) + \mathit{card}(A_2) + \mathit{card}(A_3) \\
&\quad -\mathit{card}(A_1 \cap A_2) - \mathit{card}(A_1 \cap A_3) - \mathit{card}(A_2 \cap A_3) \\
&\quad +\mathit{card}((A_1 \cap A_2) \cap A_3)
\end{aligned}$$



Laws of the algebra of sets

For any sets SE_1, SE_2 and SE_3,...

Commutativity of $\cup$ and $\cap$

$$SE_1 \cup SE_2 = SE_2 \cup SE_1$$

$$SE_1 \cap SE_2 = SE_2 \cap SE_1$$

Associativity of $\cup$ and $\cap$

$$(SE_1 \cup SE_2) \cup SE_3 = SE_1 \cup (SE_2 \cup SE_3)$$

$$(SE_1 \cap SE_2) \cap SE_3 = SE_1 \cap (SE_2 \cap SE_3)$$

Laws of the algebra of sets, cont'd.

For any sets SE_1, SE_2 and SE_3,...

Distributivity of $\cup$ over $\cap$, and of $\cap$ over $\cup$

$$SE_1 \cup (SE_2 \cap SE_3) = (SE_1 \cup SE_2) \cap (SE_1 \cup SE_3)$$

$$SE_1 \cap (SE_2 \cup SE_3) = (SE_1 \cap SE_2) \cup (SE_1 \cap SE_3)$$

De Morgan's laws

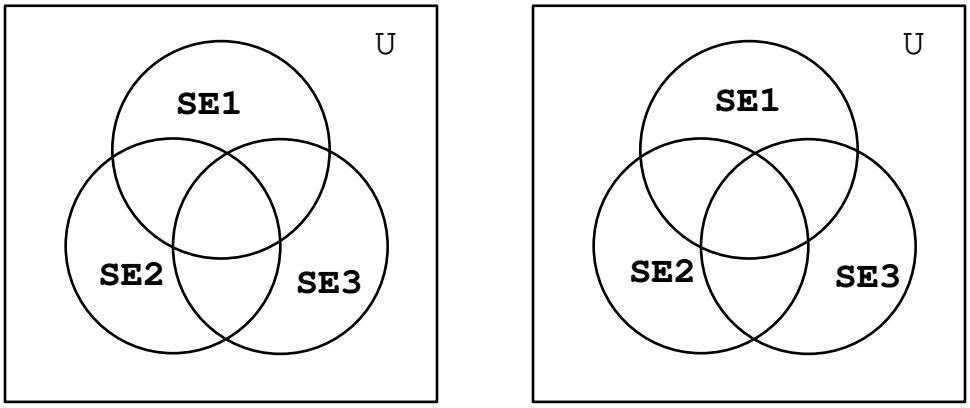
$$(SE_1 \cup SE_2)' = SE_1' \cap SE_2'$$

$$(SE_1 \cap SE_2)' = SE_1' \cup SE_2'$$

Demonstration of two of the laws

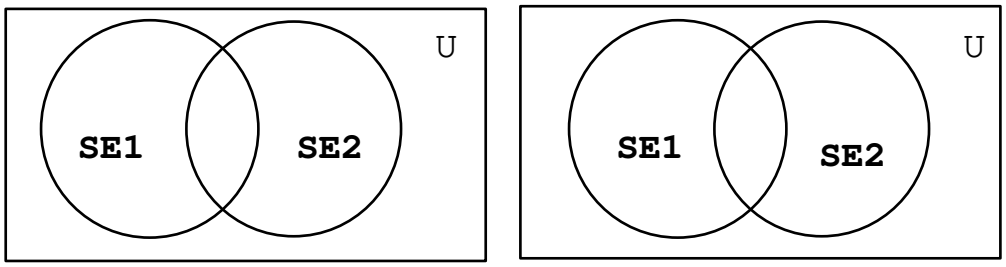
Distributivity of $\cup$ over $\cap$

$$SE_1 \cup (SE_2 \cap SE_3) = (SE_1 \cup SE_2) \cap (SE_1 \cup SE_3)$$



One of De Morgan's laws

$$(SE_1 \cup SE_2)' = SE_1' \cap SE_2'$$



Laws of the algebra of sets, cont'd

For any set SE,...

Idempotence of $\cup$ and $\cap$

$$SE \cup SE = SE$$

$$SE \cap SE = SE$$

Identities

$$\emptyset \cup SE = SE$$

$$U \cup SE = U$$

$$\emptyset \cap SE = \emptyset$$

$$U \cap SE = SE$$

Laws of the algebra of sets, cont'd.

For any set SE,...

Involution

$$(SE')' = SE$$

Complement laws

$$SE \cup SE' = U$$

$$SE \cap SE' = \emptyset$$

$$U' = \emptyset$$

$$\emptyset' = U$$

Using the laws in proofs

Example: prove that for any sets A_1 and A_2,

$$(A_1 \cap A_2) \cup (A_1 \cap U) = A_1$$

$$\begin{aligned}
\text{LHS} &= (A_1 \cap A_2) \cup (A_1 \cap U) \\
&= A_1 \cap (A_2 \cup U) \\
&= A_1 \cap (U \cup A_2) \\
&= A_1 \cap U \\
&= U \cap A_1 \\
&= A_1 \\
&= \text{RHS}
\end{aligned}$$

RELATIONS
AND
FUNCTIONS

Couples

A **couple** or **ordered pair** or **2-tuple** consists of *two* objects *taken in a certain order*.

$\langle 1, 2 \rangle$ $\quad$ $\langle 1, 1 \rangle$

$\langle DGB, MIP \rangle$ $\quad$ $\langle DGB, \{MIP, DEC, NLP\} \rangle$

$\langle DGB, \langle MIP, \langle DEC, NLP \rangle \rangle \rangle$ $\quad$ $\langle DGB, \langle \langle MIP, DEC \rangle, NLP \rangle \rangle$

The **first co-ordinate** and the **second co-ordinate**:

- can be any type of object
- can be different objects or the same object
- and *order matters*

Cartesian product

For any two sets A_1 and A_2, the **Cartesian product** of A_1 and A_2, denoted by $A_1 \times A_2$, is the set of all the couples $\langle a_1, a_2 \rangle$ such that $a_1 \in A_1$ and $a_2 \in A_2$.

More formally,

$$A_1 \times A_2 \stackrel{\triangle}{=} \{\langle a_1, a_2 \rangle \mid a_1 \in A_1 \text{ and } a_2 \in A_2\}$$

Examples: Let $S1 \stackrel{\triangle}{=} \{1, 2, 3\}$ and $S2 \stackrel{\triangle}{=} \{a, b\}$.

$$\begin{aligned} S1 \times S2 &= \{\langle 1, a \rangle, \langle 2, a \rangle, \langle 3, a \rangle, \langle 1, b \rangle, \langle 2, b \rangle, \langle 3, b \rangle\} \\ S2 \times S1 &= \{\langle a, 1 \rangle, \langle a, 2 \rangle, \langle a, 3 \rangle, \langle b, 1 \rangle, \langle b, 2 \rangle, \langle b, 3 \rangle\} \\ S2 \times S2 &= \{\langle a, a \rangle, \langle a, b \rangle, \langle b, a \rangle, \langle b, b \rangle\} \end{aligned}$$

Note that for sets A_1 and A_2,

$$\mathit{card}(A_1 \times A_2) = \mathit{card}(A_1) * \mathit{card}(A_2)$$

n-tuples

An **n-tuple** consists of n objects, where n is any positive integer, taken in a certain order.

Examples

1-tuple	$\langle a_1 \rangle$	(a degenerate case)
2-tuple	$\langle a_1, a_2 \rangle$	(couples, ordered pairs)
3-tuple	$\langle a_1, a_2, a_3 \rangle$	(triples)
4-tuple	$\langle a_1, a_2, a_3, a_4 \rangle$	
$\vdots$	$\vdots$	
n-tuple	$\langle a_1, a_2, \ldots, a_n \rangle$	(tuple)

Give three ways that n-tuples and sets differ.

Cartesian product: generalised

For any sets $A_1, A_2, \ldots, A_n$, the **Cartesian product**, denoted

$$A_1 \times A_2 \times \ldots \times A_n$$

is the set of all n-tuples $\langle a_1, a_2, \ldots, a_n \rangle$ where $a_1 \in A_1, a_2 \in A_2, \ldots, a_n \in A_n$.

Note that for sets $A_1, A_2, \ldots, A_n$

$$\textit{card}(A_1 \times A_2 \times \ldots \times A_n) = \textit{card}(A_1) * \textit{card}(A_2) * \ldots * \textit{card}(A_n)$$

Example

$$\begin{aligned} S1 &\triangleq \{1, 2, 3\} \\ S2 &\triangleq \{a, b\} \\ S3 &\triangleq \{\alpha, \beta\} \end{aligned}$$

$$S1 \times S2 \times S3 \; =$$

Cartesian product, cont'd.

For any set A,

$A \times A \times \ldots \times A$ where there are n occurrences of A

is also written as A^n

For example, $S \stackrel{\triangle}{=} \{1, 2, \}$

$$= S^2 \quad = S \times S$$

$$= S^3 \quad = S \times S \times S$$

What about S^1? $S^1 = \{\langle 1 \rangle, \langle 2 \rangle\}$. So $S^1 \neq S$.

What about S^0? $S^0 = \emptyset$

Binary relations

Relations give a mathematical way to capture 'relationships' that hold between objects.

More formally, a **binary relation** R is an object with three features:

- a set called the **domain** of R and written *Dom*(R)
- a set called the **codomain** of R and written *Cod*(R)
- a subset of the Cartesian product *Dom*(R) $\times$ *Cod*(R) called the **graph** or **extension** of R and denoted simply by R

Binary relations, cont'd.

To name a new relation R, we must specify all three components (domain, codomain and graph/extension).

To specify domain D and codomain C, we write:

$$R : D \leftrightarrow C$$

To specify the graph/extension of R, we write

$$R \stackrel{\triangle}{=} \ldots \text{some specification of a set of couples} \ldots$$

Binary relations, cont'd.

To show that objects a_1 and a_2 are elements of the relation R we write

$$\langle a_1, a_2 \rangle \in R$$

$$a_1 \; R \; a_2$$

$$R(a_1, a_2)$$

$$a_1 \mapsto a_2$$

(these sometimes being called **maplets**).

To show that objects a_1 and a_2 are *not* elements of relation R we write one of

$$\langle a_1, a_2 \rangle \notin R$$

$$a_1 \; \not R \; a_2$$

Examples of binary relations

$$S \triangleq \{\textit{Ann, Bill, Clare}\}$$
$$C \triangleq \{\textit{Goodricke, Vanbrugh, Wentworth}\}$$

We introduce the relation *BelongsTo* as follows:

$$\textit{BelongsTo} : S \leftrightarrow C$$

$$\textit{BelongsTo} \triangleq \{\langle \textit{Ann, Vanbrugh}\rangle, \langle \textit{Bill, Vanbrugh}\rangle, \langle \textit{Clare, Goodricke}\rangle\}$$

So, *Dom(BelongsTo)* $= S$ and *Cod(BelongsTo)* $= C$, and we have the following maplets:

$$\textit{Ann} \mapsto \textit{Vanbrugh}$$

$$\textit{Bill} \mapsto \textit{Vanbrugh}$$

$$\textit{Clare} \mapsto \textit{Goodricke}$$

Further examples of binary relations

$\in$ and $\notin$

$\subseteq, \subset, \not\subseteq, \not\subset$

$<, \leq, >, \geq$

$$\leq : \mathbb{N} \leftrightarrow \mathbb{N}$$

$$\leq \triangleq$$
$$\{\langle x, y\rangle \mid \text{ there exists some } z \text{ such that } z \in \mathbb{N} \text{ and } y - z = x\}$$

(The graph/extension here is infinite.)

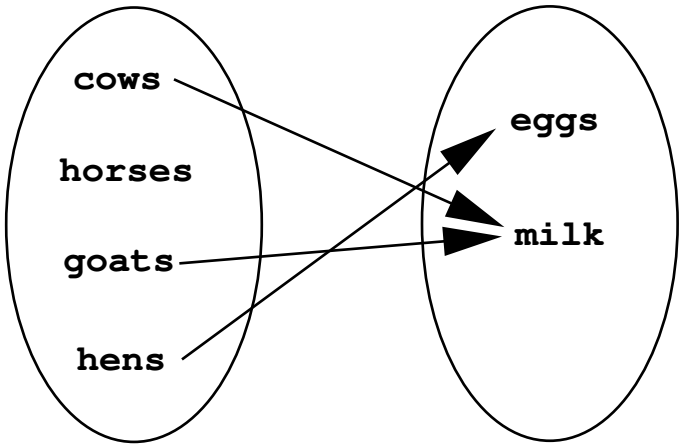
Binary relations: pictorial representations

$$\begin{aligned} \mathit{Livestock} &\triangleq \{\mathit{cows}, \mathit{horses}, \mathit{goats}, \mathit{hens}\} \\ \mathit{Foodstuffs} &\triangleq \{\mathit{eggs}, \mathit{milk}\} \end{aligned}$$

$$\mathit{Produces} : \mathit{Livestock} \leftrightarrow \mathit{Foodstuffs}$$

$$\begin{aligned} \mathit{Produces} \triangleq \quad & \{\langle \mathit{hens}, \mathit{eggs} \rangle, \langle \mathit{cows}, \mathit{milk} \rangle, \\ & \langle \mathit{goats}, \mathit{milk} \rangle\} \end{aligned}$$

Arrow diagrams

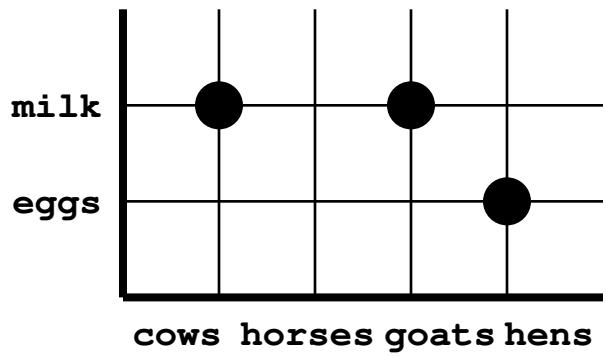


Binary relations: pictorial representations, cont'd.

Matrices

	eggs	*milk*
cows	0	1
horses	0	0
goats	0	1
hens	1	0

Coördinate diagrams

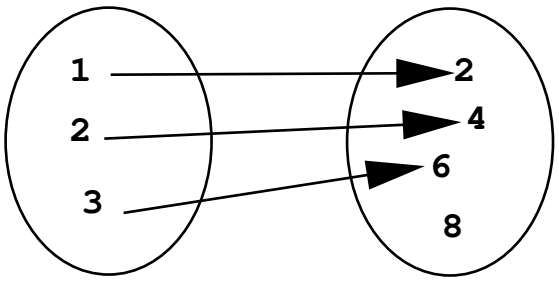


Binary relations: pictorial representations, cont'd.

$$S1 \triangleq \{1, 2, 3\}$$
$$S2 \triangleq \{2, 4, 6, 8\}$$

$$\textit{Double} \triangleq \{\langle x, y\rangle \mid y = 2 * x\}$$

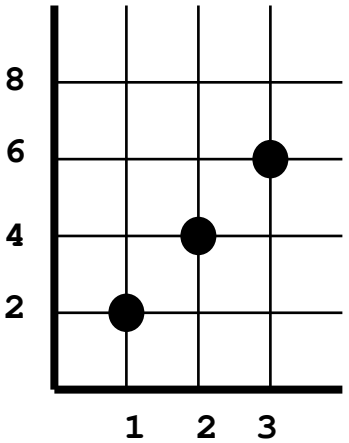
Arrow diagram



Matrix

	2	4	6	8
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0

Coördinate diagram



Binary relations: definition domains and ranges

The **definition domain** of a relation R is the set denoted by *Def*(R) and given by

$$\textit{Def}(R) \stackrel{\triangle}{=} \{a_1 \mid \text{there is some } a_2 \text{ such that } \langle a_1, a_2 \rangle \in R\}$$

The **range** of a relation R is the set denoted by *Ran*(R) and given by

$$\textit{Ran}(R) \stackrel{\triangle}{=} \{a_2 \mid \text{there is some } a_1 \text{ such that } \langle a_1, a_2 \rangle \in R\}$$

Clearly,

$$\textit{Def}(R) \subseteq \textit{Dom}(R)$$

$$\textit{Ran}(R) \subseteq \textit{Cod}(R)$$

Binary relations: definition domains and ranges example

$$
\begin{array}{lcl}
S1 & \triangleq & \{1, 2, 3, 4, 5\} \\
S2 & \triangleq & \{a, b, c\}
\end{array}
$$

$$
\mathit{REL} : S1 \leftrightarrow S2
$$

$$
\mathit{REL} \triangleq \{\langle 1, b\rangle, \langle 1, c\rangle, \langle 3, b\rangle, \langle 5, c\rangle\}
$$

Then

$$
\begin{array}{rl}
\mathit{Def(REL)} & = \{1, 3, 5\} \\
\mathit{Ran(REL)} & = \{b, c\}
\end{array}
$$

What are the definition domains and ranges of the *Produces* and *Double* relations?

Binary relations: total and partial

A relation R is **total** if and only if

$$\mathit{Def}(R) = \mathit{Dom}(R)$$

or, if you prefer,

> for all $a_1 \in \mathit{Dom}(R)$ there is at least one a_2 with $\langle a_1, a_2 \rangle \in R$

i.e. every element of the domain maps to something.

Otherwise, the relation is **partial**. (In this case, $\mathit{Def}(R) \subset \mathit{Dom}(R)$.)

Example using *REL*

$$\mathit{Def(REL)} = \{1, 3, 5\}, \quad \mathit{Dom(REL)} = \{1, 2, 3, 4, 5\}$$

Since $\mathit{Def(REL)} \subset \mathit{Dom(REL)}$, *REL* is partial.

Are *Produces* and *Double* total or partial?

Binary relations: surjective or not

A relation R is **surjective** (or **onto**) if and only if

$$\mathit{Ran}(R) = \mathit{Cod}(R)$$

or, if you prefer,

> for all $a_2 \in \mathit{Cod}(R)$ there is at least one a_1 with $\langle a_1, a_2 \rangle \in R$

i.e. every element of the codomain is mapped to.

Otherwise, the relation is **not surjective** (or **into**). (In this case, $\mathit{Ran}(R) \subset \mathit{Cod}(R)$.)

Example using *REL*

$$\mathit{Ran(REL)} = \{b, c\}, \quad \mathit{Cod(REL)} = \{a, b, c\}$$

Since $\mathit{Ran(REL)} \subset \mathit{Cod(REL)}$, *REL* is not surjective.

Are *Produces* and *Double* surjective or not?

Binary relations: functional or not

A relation R is **functional** if and only if

> for all $a_1 \in$ *Def*(R) there is exactly one a_2 with $\langle a_1, a_2 \rangle \in R$

Otherwise, the relation is **not functional**.

Example using *REL*.

$$1 \ \mathit{REL} \ b$$

$$1 \ \mathit{REL} \ c$$

There *is* an element that maps to more than one thing, so *REL* is not functional.

Are *Produces* and *Double* functional or not?

Binary relations: injective or not

A relation R is **injective** if and only if

> for all $a_2 \in$ *Ran*(R) there is exactly one a_1 with $\langle a_1, a_2 \rangle \in R$

Otherwise, the relation is **not injective**.

Example using *REL*

$$1 \ \textit{REL} \ b$$

$$2 \ \textit{REL} \ b$$

There *is* an element that is mapped to by more than one thing, so *REL* is not injective.

Are *Produces* and *Double* injective or not?

Binary relations: further terminology

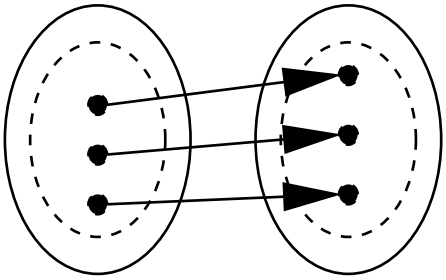
	Functional	Not functional
Injective	One-to-one	One-to-many
Not injective	Many-to-one	Many-to-many

When a relation is one-to-one (i.e. functional and injective) and also total and surjective, it is called a **bijection**.

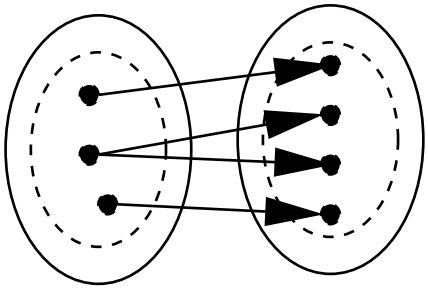
Question: if we know there is a bijection from a set A_1 to a set A_2 what else do we know about A_1 and A_2?

Binary relations: terminology summary

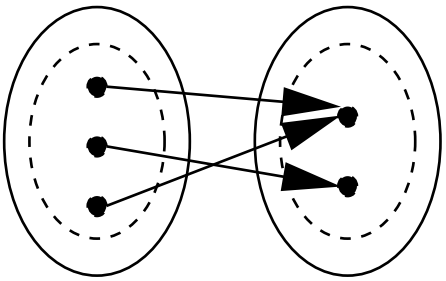
Here are diagrams demonstrating these ideas.



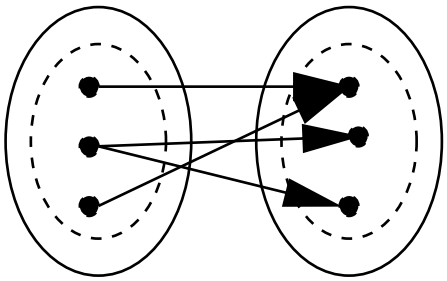
Functional & injective (One-to-one)



Not functional but injective (One-to-many)



Functional but not injective (Many-to-one)



Not functional & not injective (Many-to-many)

Which of these four describes *REL*, *Produces* and *Double*?

Operations on binary relations

For any relation R,

Dom(R)	is the operation that returns R's domain
Cod(R)	is the operation that returns R's codomain
Def(R)	is the operation that returns R's definition domain
Ran(R)	is the operation that returns R's range

Operations on binary relations: relational image

For a relation R from a set A_1 to a set A_2 and for a set A, **the image of** A **under** R is written *Im(*R, A*)*, and is defined as follows:

$$\textit{Im}(R, A) \stackrel{\triangle}{=} \{a_2 \mid \text{ there is some } a_1, \\ a_1 \in A \text{ and } a_1 \ R \ a_2\}$$

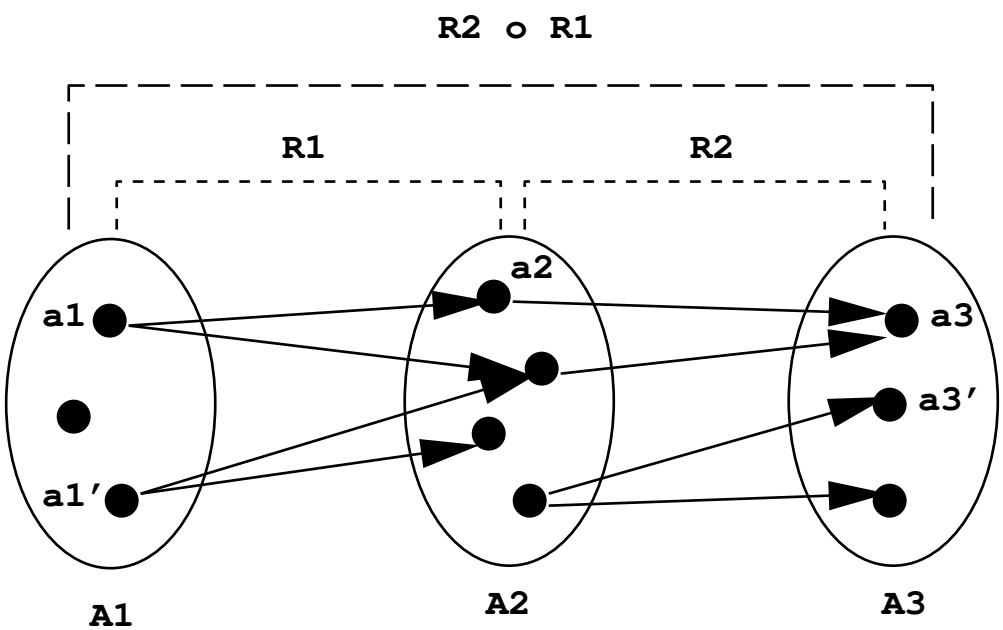
E.g.

$$\begin{aligned} \textit{Im(REL, } \{1\}\textit{)} &= \{b, c\} \\ \textit{Im(REL, } \{2, 4\}\textit{)} &= \emptyset \\ \textit{Im(Double, } \{1, 2\}\textit{)} &= \{2, 4\} \\ \textit{Im(Produces, } \{\textit{cows}\}\textit{)} &= \{\textit{milk}\} \end{aligned}$$

Operations on binary relations: relation composition

Consider two binary relations R_1 and R_2 such that

$$\mathit{Cod}(R_1) = \mathit{Dom}(R_2)$$



Operations on binary relation: relation composition, cont'd.

We can derive a third relation, the **composition** of R_1 and R_2, written $R_2 \circ R_1$ such that

$$
\begin{aligned}
\textit{Dom}(R_2 \circ R_1) &\triangleq \textit{Dom}(R_1) \\
\textit{Cod}(R_2 \circ R_1) &\triangleq \textit{Cod}(R_2) \\
R_2 \circ R_1 &\triangleq \{\langle a_1, a_3 \rangle \mid \; a_1 \in \textit{Dom}(R_1) \text{ and } a_3 \in \textit{Cod}(R_2) \text{ and there exists some } a_2, \text{ such that } a_1 \; R_1 \; a_2 \text{ and } a_2 \; R_2 \; a_3\}
\end{aligned}
$$

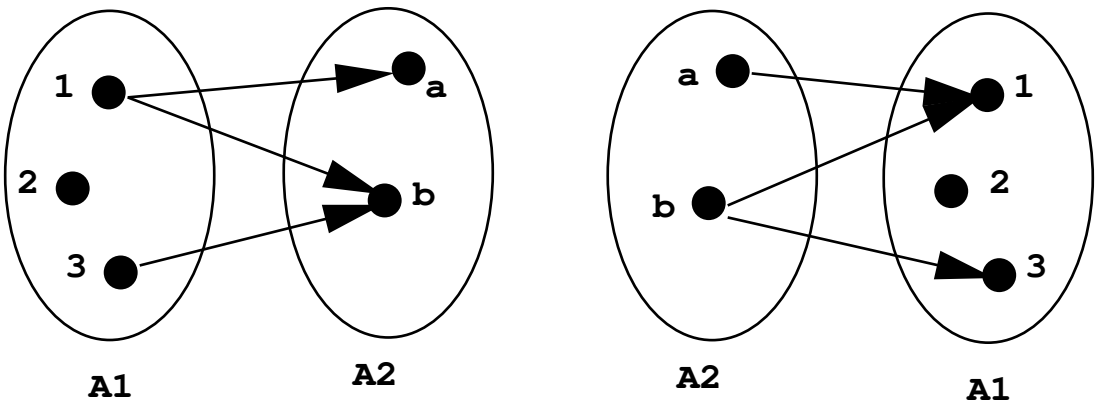
Operations on binary relation: relation composition example

$$
\begin{array}{rcl}
\mathit{REL1} & : & \{2, 4, 6\} \leftrightarrow \{1, 2, 3\} \\
\mathit{REL1} & \stackrel{\triangle}{=} & \{\langle x, y\rangle \mid x = 2 * y\} \cup \{\langle 4, 3\rangle\} \\
 & & \\
\mathit{REL2} & : & \{1, 2, 3\} \leftrightarrow \{1, 4, 9\} \\
\mathit{REL2} & \stackrel{\triangle}{=} & \{\langle x, y\rangle \mid y = x \uparrow 2\} \setminus \{\langle 1, 1\rangle\} \\
 & & \\
\mathit{REL2} \circ \mathit{REL1} & : & \{2, 4, 6\} \leftrightarrow \{1, 4, 9\} \\
\mathit{REL2} \circ \mathit{REL1} & \stackrel{\triangle}{=} & \{\langle 4, 4\rangle, \langle 4, 9\rangle, \langle 6, 9\rangle\}
\end{array}
$$

REL1 $\circ$ *REL2*, on the other hand, is undefined. Why?

Operations on binary relations: inverse

From relation $R : A_1 \leftrightarrow A_2$, we construct the **inverse** of R, denoted by R^{-1}, which is the relation from A_2 to A_1 whose graph/extension consists of those ordered pairs which, when reversed, belong to R.



$$\begin{aligned} \mathit{Dom}(R^{-1}) &\triangleq \mathit{Cod}(R) \\ \mathit{Cod}(R^{-1}) &\triangleq \mathit{Dom}(R) \\ R^{-1} &\triangleq \{\langle a_2, a_1\rangle \mid \langle a_1, a_2\rangle \in R\} \end{aligned}$$

E.g.

$$\begin{aligned} \mathit{Double}^{-1} &: S2 \leftrightarrow S1 \\ \mathit{Double}^{-1} &= \{\langle 2, 1\rangle, \langle 4, 2\rangle, \langle 6, 3\rangle\} \end{aligned}$$

Class exercise: an airline

$$
\begin{aligned}
\mathit{Passengers} &\triangleq \{\ldots, \mathit{Max}, \mathit{Nell}, \ldots\} \\
\mathit{Seats} &\triangleq \{\mathit{1A}, \mathit{1B}, \mathit{1C}, \mathit{2A}, \mathit{2B}, \ldots\} \\
\mathit{WindowSeats} &\triangleq \{\mathit{1A}, \mathit{1C}, \mathit{2A}, \mathit{2C}, \ldots\} \\
\mathit{TravCompanies} &\triangleq \{\mathit{SunFun}, \mathit{HotSpot}, \ldots\}
\end{aligned}
$$

$$
\begin{aligned}
\mathit{Alloc} &: \mathit{Passengers} \leftrightarrow \mathit{Seats} \\
\mathit{Alloc} &\triangleq \{\ldots, \langle \mathit{Max}, \mathit{46A} \rangle, \langle \mathit{Nell}, \mathit{52C} \rangle, \ldots\}
\end{aligned}
$$

$$
\begin{aligned}
\mathit{BookedFlightFor} &: \mathit{TravCcompanies} \leftrightarrow \mathit{Passengers} \\
\mathit{BookedFlightFor} &\triangleq \{\ldots, \langle \mathit{SunFun}, \mathit{Max} \rangle, \\
&\qquad \langle \mathit{HotSpot}, \mathit{Quentin} \rangle, \ldots\}
\end{aligned}
$$

Class exercise

1. What happens if *Alloc* is *not* injective?

2. What can we conclude if *Alloc* is *not* functional?

3. What does the following test for?

$$Ran(Alloc) \subset Cod(Alloc)$$

4. Write an expression that would test whether every passenger has a seat.

Class exercise

5. Max, Nell, Oswald and Penny are a family. What would the following compute for them?

$$\mathit{Im}(\mathit{Alloc}, \{\mathit{Max}, \mathit{Nell}, \mathit{Oswald}, \mathit{Penny}\})$$

6. Write an expression that would tell Quentin where he should sit.

7. Quentin wants to know whether he has a window seat. What's wrong with the following?

$$\mathit{Im}(\mathit{Alloc}, \{\mathit{Quentin}\}) \in \mathit{WindowSeats}$$

Write an expression that would answer Quentin's question.

8. What would the following compute?

$$\mathit{Cod}(\mathit{BookedFlightFor}) \setminus \mathit{Ran}(\mathit{BookedFlightFor})$$

Class exercise

9. What would the following compute?

$$Alloc \circ BookedFlightFor$$

10. I'm the travel rep. for HotSpot. Write an expression that would tell me where my passengers are sitting.

11. What would the following compute?

$$Alloc^{-1}$$

and

$$BookedFlightFor^{-1}$$

12. The passengers in seats 46A, 46B and 45B misbehave during dinner. The steward(ess) wants to complain to their travel company/companies. Write an expression that would determine which companies are involved.

Relations on a set

A binary relation from a set A to the same set is called a binary **relation on** A, i.e. relation R is a relation on A if and only if

$$\mathit{Dom}(R) = \mathit{Cod}(R)$$

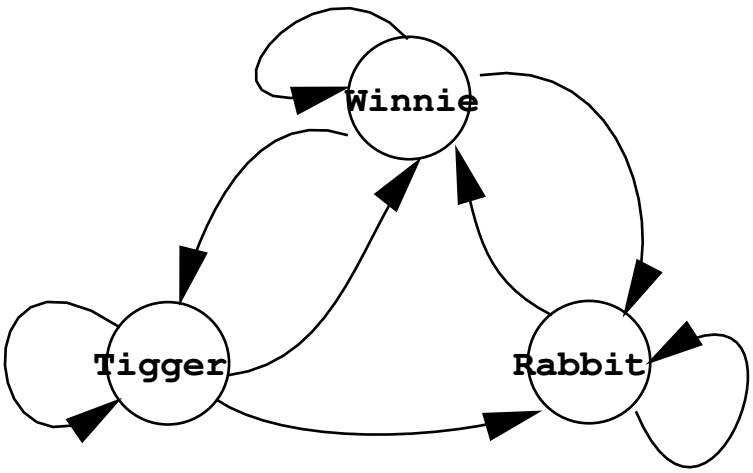
$$R \subseteq A^2$$

Example

$$\begin{array}{rcl} \mathit{Creatures} & \triangleq & \{\mathit{Winnie}, \mathit{Tigger}, \mathit{Rabbit}\} \\ \mathit{Likes} & : & \mathit{Creatures} \leftrightarrow \mathit{Creatures} \\ \mathit{Likes} & \triangleq & (\mathit{Creatures} \times \mathit{Creatures}) \setminus \\ & & \quad \{\langle \mathit{Rabbit}, \mathit{Tigger} \rangle\} \end{array}$$

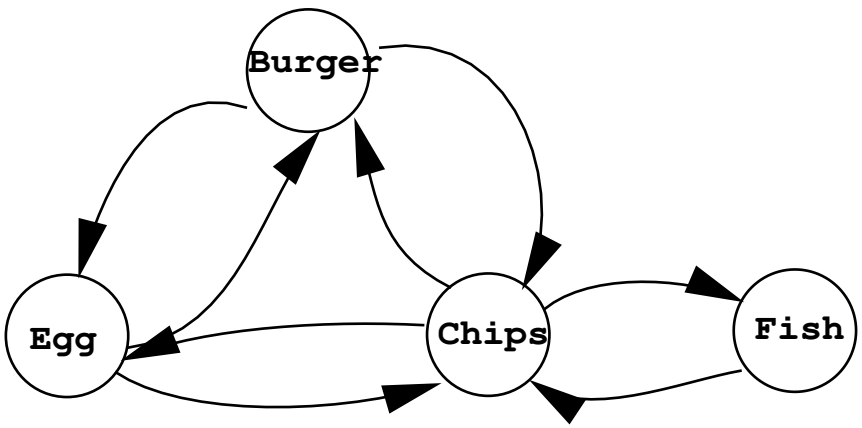
Relations on a set: pictorial representation

Directed graph



Relations on a set: another example

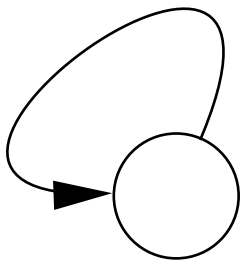
$$
\begin{array}{rcl}
\mathit{Grub} & : & \{\mathit{Burger}, \mathit{Chips}, \mathit{Eggs}, \mathit{Fish}\} \\
\mathit{GoTogether} & : & \mathit{Grub} \leftrightarrow \mathit{Grub} \\
\mathit{GoTogether} & \triangleq & \{\langle \mathit{Burger}, \mathit{Chips} \rangle, \ldots\}
\end{array}
$$



Relations on sets: reflexive

A binary relation R on a set A is **reflexive** if and only if for every element $a \in A$,

$$a \; R \; a$$



A binary relation R on a set A is **non-reflexive** if there's at least one element that is not related to itself.

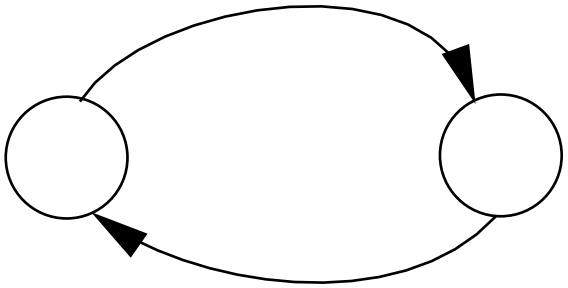
A binary relation R on a set A is **antireflexive** if no element is related to itself.

Were *Likes* and *GoTogether* reflexive, non-reflexive or antireflexive?

Relations on sets: symmetric

A binary relation R on a set A is **symmetric** if and only if for every pair of elements a_1 and a_2,

if $a_1 \ R \ a_2$ then $a_2 \ R \ a_1$



A binary relation R on a set A is **non-symmetric** if there's at least one maplet $a_1 \mapsto a_2$ for which there is no maplet $a_2 \mapsto a_1$.

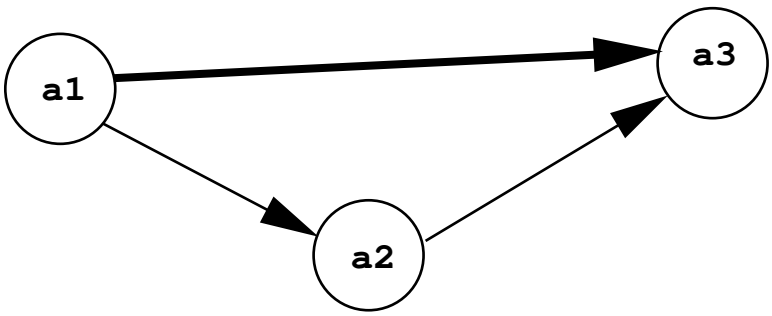
A binary relation R on a set A is **antisymmetric** if none of the maplets $a_1 \mapsto a_2$ where $a_1 \neq a_2$ have corresponding maplets in the other direction ($a_2 \mapsto a_1$).

Were *Likes* and *GoTogether* symmetric, non-symmetric or antisymmetric?

Relations on sets: transitive

A binary relation R on a set A is **transitive** if and only if for every triple of elements a_1, a_2 and a_3,

if $a_1 \ R \ a_2$ and $a_2 \ R \ a_3$ then $a_1 \ R \ a_3$



Were *Likes* and *GoTogether* transitive or not?

Relations on sets: equivalence relations

A relation R on a set A is called an **equivalence relation** if it is *reflexive, symmetric* and *transitive*.

Examples

Consider the relation *IsTheSameSpeciesAs* on the set of animals. Clearly, every animal a is the same species as itself (reflexive); if animal a_1 is the same species as animal a_2 then animal a_2 is the same species as animal a_3 (symmetric); and if animal a_1 is the same species as animal a_2 and animal a_2 is the same species as animal a_3, then animal a_1 is the same species as animal a_3 (transitive). Hence, *IsTheSameSpeciesAs* is an equivalence relation.

Other examples: the relation *IsTheSameHeightAs* on the set of people (say); the relation *IsTheSameAgeAs* on the set of students (say); the relation $=$ on $\mathbb{N}$ (say).

Relations on sets: equivalence relations, cont'd.

An equivalence relation R on set A forms a *partition* of set A. Each set within the partition is called an **equivalence class**. Elements related to each other by R will belong to the same equivalence class.

Example

$$\textit{People} \stackrel{\triangle}{=} \{\textit{Jack, Joan, Jill, Jim, Joe, Judith}\}$$

Consider the relation on *People* called *SAA* (same age as) given that: *Jack* and *Joan* are 19, *Jim, Joe* and *Judith* are 20, and *Jill* is 21.

The partition is

$$\{\{\textit{Jack, Joan}\}, \{\textit{Jim, Joe, Judith}\}, \{\text{Jill}\}\}$$

Relations on sets: equivalence relations, cont'd.

Give a partition of a set A, we can determine an equivalence relation R that would partition A in the way given. Specifically, we will have $a_1 \ R \ a_2$ if and only if a_1 and a_2 are members of the same set within the partition.

Example: from this partition of the set *People*,

$$\{\{\textit{Jack}\}, \{\textit{Jim, Jill}\}, \{\textit{Joan, Joe}\}, \{\textit{Judith}\}\}$$

we can derive the equivalence relation on *People* whose graph/extension is

$$\begin{aligned} \textit{SSL} \triangleq \\ &\{\langle \textit{Jack, Jack}\rangle, \langle \textit{Joan, Joan}\rangle, \langle \textit{Jill, Jill}\rangle, \\ &\ \langle \textit{Jim, Jim}\rangle, \langle \textit{Joe, Joe}\rangle, \langle \textit{Judith, Judith}\rangle, \\ &\ \langle \textit{Jim, Jill}\rangle, \langle \textit{Jill, Jim}\rangle, \langle \textit{Joan, Joe}\rangle, \langle \textit{Joe, Joan}\rangle\} \end{aligned}$$

Relations on sets: orderings

Some relations on a set are **orderings** of the elements of the set.

Examples are: $<$ on $\mathbb{N}$ (say), $\leq$ on $\mathbb{N}$ (say), *IsShorterThan* on the set of people (say), *IsYoungerThan* on the set of buildings (say), *ScoredHigherThan* on a set of maths students (say).

Crucial to ordering is transitivity (if a_1 is less than a_2 and a_2 is less than a_3 then a_1 is less than a_3).

We will require antisymmetry (in no case is a_1 less than a_2 and a_2 also less than a_1 where $a_1 \neq a_2$.)

So, we require **transitivity** and **antisymmetry**.

Relations on sets: orderings example

$$People \triangleq \{Jack, Joan, Jill, Jim, Joe, Judith\}$$

Consider the relation on *People* called *YT* (younger than) given that: *Jack* and *Joan* are 19, *Jim, Joe* and *Judith* are 20, and *Jill* is 21.

Relations on sets: orderings example

$$SomeNumbers \triangleq \{1, 2, 3, 4\}$$

Consider the relations *LT* (less than) and *LTOET* (less than or equal to) on *SomeNumbers*

Relations on sets: strict/non-strict orderings

- **Strict** or **non-strict**
  - If an ordering is *antireflexive* (no object is related to itself), as with $<$ on $\mathbb{N}$, then the ordering is *strict*.
  - If an ordering is *reflexive* (every element is related to itself), as with $\leq$ on $\mathbb{N}$, then the ordering is *non-strict*.

Relations on sets: total and partial orderings

- **Total** versus **partial**
  - If for all elements a_1 and a_2 where $a_1 \neq a_2$, a_1 is related to a_2 or a_2 is related to a_1, then the ordering is *total*.
  - If there is at least one pair of elements a_1 and a_2 where $a_1 \neq a_2$ such that a_1 is not related to a_2 and a_2 is not related to a_1 (i.e. they are incomparable), then the ordering is *partial*.

This gives us four kinds of ordering:

$$\left\{ \begin{array}{c} \text{strict} \\ \text{non-strict} \end{array} \right\} \left\{ \begin{array}{c} \text{total} \\ \text{partial} \end{array} \right\} \text{ordering.}$$

Operations on relations on sets: closures

The **reflexive closure** operation takes in a relation R on a set A and returns another relation denoted *RC(R)* whose graph/extension is the same as the graph/extension of R except that it also contains the minimum additional tuples needed to make the new relation reflexive.

The **transitive closure** operation takes in a relation R on a set A and returns another relation denoted *TC(R)* whose graph/extension is the same as the graph/extension of R except that it also contains the minimum additional tuples needed to make the new relation transitive.

The **reflexive, transitive closure** operation takes in a relation R on a set A and returns another relation denoted *RTC(R)* whose graph/extension is the same as the graph/extension of R except that it also contains the minimum additional tuples needed to make the new relation reflexive and transitive.

Operations on relations on sets: closure examples

$$S \triangleq \{0, 1, 2\}$$

$$\mathit{REL} : S \leftrightarrow S$$

$$\mathit{REL} \triangleq \{\langle 0, 0\rangle, \langle 0, 1\rangle, \langle 1, 2\rangle, \langle 2, 0\rangle\}$$

Then,

$$\mathit{RC(REL)} \triangleq \{\langle 0, 0\rangle, \langle 0, 1\rangle, \langle 1, 2\rangle, \langle 2, 0\rangle, \langle 1, 1\rangle, \langle 2, 2\rangle\}$$

$$\mathit{TC(REL)} \triangleq \{\langle 0, 0\rangle, \langle 0, 1\rangle, \langle 1, 2\rangle, \langle 2, 0\rangle, \langle 0, 2\rangle, \langle 1, 0\rangle, \langle 2, 1\rangle\}$$

$$\mathit{RTC(REL)} \triangleq \{\langle 0, 0\rangle, \langle 0, 1\rangle, \langle 1, 2\rangle, \langle 2, 0\rangle, \langle 1, 1\rangle, \langle 2, 2\rangle, \langle 0, 2\rangle, \langle 1, 0\rangle, \langle 2, 1\rangle\}$$

Operations on relations on sets: closure example

$$A \stackrel{\triangle}{=} \{\textit{Adam, Dan, Christine, Phil, David, Elizabeth, Shula, Kenton}\}$$

Consider the *Parent* relation on *A*.

What does *TC(Parent)* compute?

Functional relations

Functional relations are simply called **functions** for short.

To introduce a function F with domain D and codomain C,

$$F : D \to C$$

Functions

To specify elements of the graph/extension

$$F(a_1) = a_2$$

For any object a in the domain of function F, the object assigned to a, if there is one, is denoted $F(a)$. We say that F is **applied to** a, a is the **argument** of F, $F(a)$ is **the result of applying** F to a, and $F(a)$ is the **value of** F **at** a.

Example: *treble* is a function on $\mathbb{N}$.

$$\mathit{treble} : \mathbb{N} \to \mathbb{N}$$

$$\mathit{treble} \stackrel{\triangle}{=} \{\langle x, y\rangle \mid y = 3 * x\}$$

or,

$$\mathit{treble}(x) \stackrel{\triangle}{=} 3 * x$$

Function: non-numerical examples

The relation *Mother* from people to their mothers is functional; so is *Father*.

Parents and *Children* (from people to their parents/children) are not functions.

Height, *Age*, *Gender* and *Birthday* (from people to their heights/ ages/ genders/ birthdays) are functions (if the domain does not include the Queen and Paddington Bear).

For books, *Title*, *NoOfPages* and *Price* (from books to their titles, lengths and prices) are (probably) functional; *Authors* is not.

Representing (non-functional) relations as functions

$$Royals \triangleq \{Liz,\ Phil,\ Chas,\ Andy,\ Willie\}$$

Consider the following (non-functional) relation:

$$Child : Royals \leftrightarrow Royals$$

$$\begin{aligned} Child \triangleq\ & \{\langle Chas,\ Liz\rangle, \langle Chas,\ Phil\rangle, \\ & \langle Andy,\ Liz\rangle, \langle Andy,\ Phil\rangle, \\ & \langle Willie,\ Chas\rangle\} \end{aligned}$$

Now consider the following function:

Alone

$$ChildFunc : Royals \to \wp(Royals)$$

$$\begin{aligned} ChildFunc(Chas) &= \{Liz,\ Phil\} \\ ChildFunc(Andy) &= \{Liz,\ Phil\} \\ ChildFunc(Willie) &= \{Chas\} \end{aligned}$$

Representing sets as functions

$$U \stackrel{\triangle}{=} \{\textit{Liz, Phil, Chas, Andy, Willie}\}$$

$$\textit{Alone} \stackrel{\triangle}{=} \{\textit{Chas, Andy}\}$$

$$\textit{AloneFunc} : U \rightarrow \textit{Bool}$$

$$\begin{aligned} \textit{AloneFunc(Liz)} &= F \\ \textit{AloneFunc(Phil)} &= F \\ \textit{AloneFunc(Chas)} &= T \\ \textit{AloneFunc(Andy)} &= T \\ \textit{AloneFunc(Willie)} &= F \end{aligned}$$

For any set A drawn from some universal set U, the function from U to *Bool* that maps an element $a \in U$ to T (true) if $a \in A$ and to F (false) if $a \notin A$ is called the **characteristic function of set** A.

Representing tuples as functions

$$\langle \mathit{Phil, Chas, Willie} \rangle$$

Now consider this function:

$$\mathit{TupleFunc} : \mathbb{P} \rightarrow \{\mathit{Phil, Chas, Willie}\}$$

$$\begin{array}{lcl} \mathit{TupleFunc(1)} & = & \mathit{Phil} \\ \mathit{TupleFunc(2)} & = & \mathit{Chas} \\ \mathit{TupleFunc(3)} & = & \mathit{Willie} \end{array}$$

Any n-tuple has a corresponding function from the set of integers from 1 to n to the elements of the tuple.

n-ary relations

An n-**ary relation** is an object R with the following features:

- n sets, called the **domains** of R, each denoted $\mathit{Dom}_i(R)$ for $i = 1$ to n.

- a subset of the Cartesian product of the domains called the **graph/extension** of R

$$R \subseteq \mathit{Dom}_1(R) \times \mathit{Dom}_2(R) \times \ldots \times \mathit{Dom}_n(R)$$

To introduce a new n-ary relation,

$$R : D_1 \leftrightarrow D_2 \leftrightarrow \ldots \leftrightarrow D_n$$

$R \triangleq$ …some specification of a set of n-tuples…

n-ary relations, cont'd.

To show that objects $a_1, a_2, \ldots, a_n$ are elements of an n-ary relation R, we write

$$\langle a_1, a_2, \ldots, a_n \rangle \in R$$

or

$$R(a_1, a_2, \ldots, a_n)$$

Example of a 3-ary relation

The relation *Employees* has the following domains:

$$
\begin{array}{rcl}
\mathit{Names} & \triangleq & \{\mathit{Ann},\ \mathit{Bill},\ \mathit{Clare},\ldots,\ \mathit{Zoë}\} \\
\mathit{Genders} & \triangleq & \{\mathit{Male},\ \mathit{Female}\} \\
\mathit{MarStatus} & \triangleq & \{\mathit{Single},\ \mathit{Married}\}
\end{array}
$$

$$\mathit{Employees} : \mathit{Names} \leftrightarrow \mathit{Genders} \leftrightarrow \mathit{MarStatus}$$

$$
\begin{array}{l}
\mathit{Employees} \triangleq \\
\quad \{\langle \mathit{Ann},\ \mathit{Female},\ \mathit{Single},\rangle, \\
\quad\ \langle \mathit{Bill},\ \mathit{Male},\ \mathit{Single},\rangle, \\
\quad\ \langle \mathit{Clare},\ \mathit{Female},\ \mathit{Married}\rangle, \ldots\}
\end{array}
$$

n-ary relations, cont'd.

Tabular representation

EMPLOYEES		
EmpNames	EmpGenders	EmpMarStatus
Ann	Female	Single
Bill	Male	Single
Clare	Female	Married
⋮	⋮	⋮

n-ary functions

An n-ary function would be introduced as follows:

$$F : A_1 \times A_2 \times \ldots \times A_n \rightarrow A_{n+1}$$

i.e.

$$\textit{Dom}(F) = A_1 \times A_2 \times \ldots \times A_n$$

$$\textit{Cod}(F) = A_{n+1}$$

And, in the graph/extension of F, each tuple $\langle a_1, a_2, \ldots, a_n \rangle$ from *Dom(F)* will map to no more than one object in *Cod(F)*.

Note that an n-ary function is a $(n + 1)$-ary relation. So, e.g., a unary function is a binary relation; a binary function is a ternary relation; a ternary function is a 4-ary relation; and so on.

Examples of n-ary functions

Addition, subtraction, multiplication and division on $\mathbb{N}$ (say) binary functions (3-ary functional relations):

$$+ : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$$

$$* : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$$

A timetable is a 4-ary relation, and, if there are no double bookings, then it is a 3-ary function:

$$\mathit{Timetable} : \mathit{Day} \times \mathit{Time} \times \mathit{Room} \rightarrow \mathit{Course}$$

Using our newer way of showing maplets, we would write:

$$\mathit{Timetable(Tue,\ 12.15\text{-}13.15,\ B/B006)} = \mathit{MIP}$$

PROPOSITIONAL LOGIC

Logic

Logic is the study of reasoning.

1. Every elephant is grey.

2. Clyde is an elephant.

3. Therefore, Clyde is grey.

This a valid argument.

The validity of an argument rests on its *form*, not the truth of the sentences that comprise it.

The uses of logic in Computer Science

- Circuit design
- Boolean data types
- Program specification
- Program verification
- Logic programming
- Automated theorem-proving
- Artificial Intelligence

Symbolic logic

For most of the uses of logic, a well-defined, precise, unambiguous notation is needed, leading us to what is referred to as **symbolic logic**.

We look at

- Propositional logic

- First-order predicate logic

Statements

The two logics we look at give analyses of sentences that make **statements**. E.g.:

- It is sunny
- $4 > 5$
- Roses are red and violets are blue

A statement will be true or false. True (T) and false (F) are referred to as **truth values**.

The structure of a statement

We form **compound statements** from **atomic statements**, by joining them using **connectives**; we form atomic statements from expressions that refer to objects, their properties and their relationships. E.g.

> ‘Clyde is cold and Clyde is wet.’

The truth value of an *atomic* statement depends on the objects, properties and relationships denoted by its subexpressions. The truth value of a *compound* statement depends on the truth values of its atomic statements.

Truth-functional connectives

The meaning (truth value) of a compound statement depends *only* on the meanings (truth values) of the statements of which it is composed.

We must therefore restrict our attention to connectives that are **truth-functional**.

> ‘Clyde is cold and Clyde is wet.’

> ‘Clyde is cold because Clyde is wet.’

So, ‘and’ is a truth-functional connective but ‘because’ is not.

Propositional logic

Given the limited amount of decomposition of statements that we are allowing for the present an atomic proposition may just as well be represented by a name such as p, p_1 or q.

These names are called **propositional symbols**.

If desired, they can be related to English statements, as follows:

$$p \stackrel{\triangle}{=} \text{Clyde is cold}$$

$$q \stackrel{\triangle}{=} \text{Clyde is wet}$$

The syntax of propositional logic

The expressions of this language are called **well-formed formulas** or **wffs**.

1. Every **propositional symbol** is a wff.

2. For any wff W, the following is also a wff:

$$\neg W$$

3. For any two wffs W_1 and W_2, the following are also wffs:

$$W_1 \wedge W_2$$

$$W_1 \vee W_2$$

$$W_1 \Rightarrow W_2$$

$$W_1 \Leftrightarrow W_2$$

The syntax of propositional logic, cont'd.

4. Any wff may be enclosed in parentheses and will still be a wff.

5. No other string of symbols is a wff.

The symbols $\neg, \wedge, \vee, \Rightarrow$ and $\Leftrightarrow$ are the **connectives**.

Here are examples of wffs:

$$p \qquad q$$
$$(\neg p) \qquad (\neg q)$$
$$(p \wedge p) \qquad (p \vee q)$$
$$(p \Leftrightarrow (q_1 \vee q_2)) \qquad ((p_1 \wedge p_2) \vee (q_1 \wedge q_2))$$

Precedence and associativity

- Parentheses determine evaluation order.

- Precedence rules can be given:

  Highest $\neg \vee \wedge \Rightarrow \Leftrightarrow$ Lowest

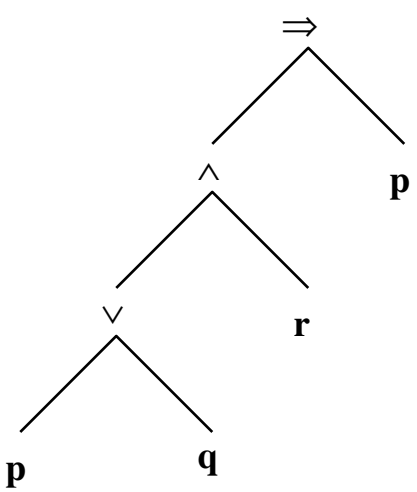
  e.g. $p \vee q \wedge r \Rightarrow p$ evaluates as $((p \vee q) \wedge r) \Rightarrow p$

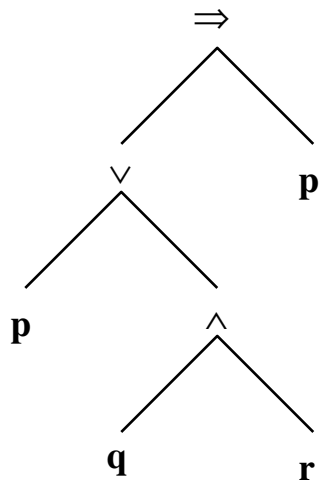
- Associativity rules can be given:

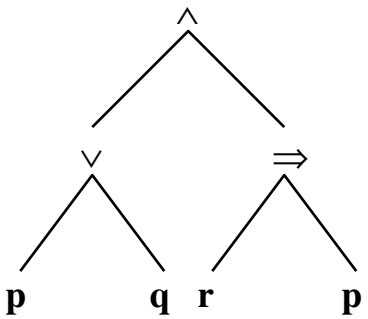
  Associate to the left: $\wedge \vee \Leftrightarrow$
  Associate to the right: $\Rightarrow$

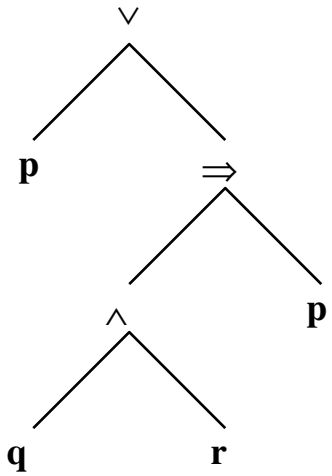
  e.g. $p \wedge q \wedge r \Rightarrow p \Rightarrow r$ evaluates as $((p \wedge q) \wedge r) \Rightarrow (p \Rightarrow r)$

Trees









The semantics of propositional logic

Wffs represent propositions, and these are either true or false.

Atomic wffs

The truth values of atomic wffs are *stipulated*.

A stipulation of the truth values of atomic wffs is called an **interpretation** of the atomic wffs.

Example

$$\mathcal{I}(p) \text{ is } T; \quad \mathcal{I}(q) \text{ is } T; \quad \mathcal{I}(r) \text{ is } F$$

Compound wffs

The truth value of a compound wff is determined by the truth values of its component wffs.

Negation

Given any wff W, another wff, called the **negation** of W, can be formed, and is denoted

$$\neg W$$

This is read as 'not W'.

For any wff W,

$$\mathcal{I}(\neg W) \stackrel{\triangle}{=} \begin{array}{l}\text{if } \mathcal{I}(W) \text{ is } T \\ \quad \text{then } F \\ \quad \text{else } T\end{array}$$

W	$\neg W$
T	F
F	T

Negation, cont'd.

$$
\begin{aligned}
p_1 &\stackrel{\triangle}{=} \text{ Paris is in France} \\
p_2 &\stackrel{\triangle}{=} 2+2=5 \\
p_3 &\stackrel{\triangle}{=} \text{ Clyde is intelligent}
\end{aligned}
$$

Consider the interpretation

$$\mathcal{I}(p_1) \text{ is } T; \quad \mathcal{I}(p_2) \text{ is } F; \quad \mathcal{I}(p_3) \text{ is } T$$

then

$$\mathcal{I}(\neg p_1) \text{ is } F; \quad \mathcal{I}(\neg p_2) \text{ is } T; \quad \mathcal{I}(\neg p_3) \text{ is } F$$

Negation, cont'd.

$\neg p_1$ 'It is not the case that Paris in in France'
$\neg p_2$ 'It's not the case that $2 + 2 = 5$'

$\neg p_1$ 'Paris is not in France'
$\neg p_2$ $2 + 2 \neq 5$

Negation: (easy) class exercise

$$q_1 \triangleq \text{Clyde likes Flopsy}$$
$$q_2 \triangleq \text{Clyde is happy}$$

Paraphrase into English

$$\neg q_1$$

Translate into propositional logic

It's not the case that Clyde is happy.
Clyde isn't happy.
Clyde is unhappy.
Clyde is sad.

Conjunction

Any two wffs W_1 and W_2 can be combined to form a compound wff, called the **conjunction** of W_1 and W_2, and denoted

$$W_1 \wedge W_2$$

W_1 and W_2 are called the **conjuncts**.

For any wffs W_1 and W_2,

$$\mathcal{I}(W_1 \wedge W_2) \stackrel{\triangle}{=} \begin{array}{l}\text{if } \mathcal{I}(W_1) \text{ is } T \text{ and } \mathcal{I}(W_2) \text{ is } T \\ \quad \text{then } T \\ \quad \text{else } F\end{array}$$

W_1	W_2	$W_1 \wedge W_2$
T	T	T
T	F	F
F	T	F
F	F	F

Conjunction, cont'd.

$$
\begin{aligned}
p_1 &\stackrel{\triangle}{=} \text{Paris is in France} \\
p_2 &\stackrel{\triangle}{=} \text{York is in England} \\
p_3 &\stackrel{\triangle}{=} 2+2=5
\end{aligned}
$$

Consider

$$\mathcal{I}(p_1) \text{ is } T; \quad \mathcal{I}(p_2) \text{ is } T; \quad \mathcal{I}(p_3) \text{ is } F$$

then, e.g.,

$\mathcal{I}(p_1 \wedge p_2)$ is T; $\mathcal{I}(p_1 \wedge p_3)$ is F; $\mathcal{I}(p_3 \wedge p_3)$ is F

$p_1 \wedge p_3$ 'Paris is in France and $2+2=5$'
'Both, Paris is in France and $2+2=5$'

Conjunction: (easy) class exercise

$$q_1 \stackrel{\triangle}{=} \text{Clyde likes Flopsy}$$
$$q_2 \stackrel{\triangle}{=} \text{Clyde is happy}$$

Paraphrase into English

$$q_1 \wedge \neg q_2$$

Translate into propositional logic

> It's not the case that both Clyde likes Flopsy and Clyde is happy.

Conjunction, cont'd.

$p_1 \wedge p_2$ 'Paris is in France but York is in England'
'Paris in in France whereas York is in England'
'Although Paris is in France, York is in England'

'Batman pulled off his socks and jumped into bed.'

'Batman jumped into bed and pulled off his socks.'

'Batman and Robin are superheroes.'

'Batman is a superhero and Robin is a superhero.'

'Batman and Robin are fun at parties.'

Disjunction

Any two wffs W_1 and W_2 can be combined to form a compound wff, called the **disjunction** of W_1 and W_2, and denoted

$$W_1 \vee W_2$$

This is read as 'W_1 or W_2' or 'W_1 or W_2 or both' or 'W_1 inclusive-or W_2'. W_1 and W_2 are called the **disjuncts**.

For any wffs W_1 and W_2,

$$\mathcal{I}(W_1 \vee W_2) \stackrel{\triangle}{=} \begin{array}{l}\text{if } \mathcal{I}(W_1) \text{ is } T \text{ or } \mathcal{I}(W_2) \text{ is } T \text{ (or both)} \\ \quad \text{then } T \\ \quad \text{else } F\end{array}$$

W_1	W_2	$W_1 \vee W_2$
T	T	T
T	F	T
F	T	T
F	F	F

Disjunction, cont'd.

$$
\begin{aligned}
p_1 &\stackrel{\triangle}{=} \text{Paris is in France} \\
p_2 &\stackrel{\triangle}{=} \text{York is in England} \\
p_3 &\stackrel{\triangle}{=} 2+2=5
\end{aligned}
$$

Consider

$$\mathcal{I}(p_1) \text{ is } T; \quad \mathcal{I}(p_2) \text{ is } F; \quad \mathcal{I}(p_3) \text{ is } F$$

then, e.g.

$$\mathcal{I}(p_1 \vee p_2) \text{ is } T; \quad \mathcal{I}(p_2 \vee p_3) \text{ is } F; \quad \mathcal{I}(p_1 \vee p_1) \text{ is } T$$

$p_1 \vee p_3$ 'Paris is in France or $2+2=5$ (or both)'

Disjunction, cont'd.

'W_1 or W_2 or both'	**inclusive-or**
'W_1 or W_2 but not both'	**exclusive-or**

$p_1 \vee p_2$	'Paris is in France or York is in England (or both)'
$(p_1 \vee p_2) \wedge \neg(p_1 \wedge p_2)$	'Paris is in France or York is in England (but not both)' '*Either* Paris is in France or York is in England' 'Paris is in France unless York is in England'

Conditional

Any two wffs W_1 and W_2 can be combined to form a compound wff denoted

$$W_1 \Rightarrow W_2$$

called a **conditional**. This can be read as 'if W_1 then W_2'. W_1 is called the **antecedent** and W_2 is called the **consequent**.

For any wffs W_1 and W_2,

$$\mathcal{I}(W_1 \Rightarrow W_2) \stackrel{\triangle}{=} \begin{array}{l}\text{if } \mathcal{I}(W_1) \text{ is } F \text{ or } (W_2) \text{ is } T \\ \quad \text{then } T \\ \quad \text{else } F\end{array}$$

W_1	W_2	$W_1 \Rightarrow W_2$
T	T	T
T	F	F
F	T	T
F	F	T

Conditional, cont'd.

Consider

$$p \stackrel{\triangle}{=} \text{I win the election}$$
$$q \stackrel{\triangle}{=} \text{Taxes will fall}$$

- Suppose s/he wins and taxes fall:

  $\mathcal{I}(p)$ is T; $\mathcal{I}(q)$ is T; so $\mathcal{I}(p \Rightarrow q)$ is T

- Suppose s/he wins and taxes rise or stay the same:

  $\mathcal{I}(p)$ is T; $\mathcal{I}(q)$ is F; so $\mathcal{I}(p \Rightarrow q)$ is F

- Suppose s/he loses and taxes fall:

  $\mathcal{I}(p)$ is F; $\mathcal{I}(q)$ is T; so $\mathcal{I}(p \Rightarrow q)$ is T

- Suppose s/he loses and taxes rise or stay the same:

  $\mathcal{I}(p)$ is F; $\mathcal{I}(q)$ is F; so $\mathcal{I}(p \Rightarrow q)$ is T

Conditional, cont'd.

$$p_1 \stackrel{\triangle}{=} \text{Paris is in France}$$
$$p_2 \stackrel{\triangle}{=} \text{York is in England}$$
$$p_3 \stackrel{\triangle}{=} 2 + 2 = 5$$

$p_1 \Rightarrow p_2$ 'If Paris is in France then York is in England'
'Paris is in France implies York is in England'
'Paris is in France only if York is in England'
'York is in England if Paris is in France'
'Paris being in France is a sufficient condition for York to be in England'
'York being in England is a necessary condition for Paris to be in France'

Biconditional

Any two wffs W_1 and W_2 can be combined to form a compound wff, denoted

$$W_1 \Leftrightarrow W_2$$

which is called a **biconditional**. This is read as 'W_1 if and only if W_2'.

For any wffs W_1 and W_2,

$$\mathcal{I}(W_1 \Leftrightarrow W_2) \stackrel{\triangle}{=} \begin{array}{l}\text{if } \mathcal{I}(W_1) \text{ is the same as } \mathcal{I}(W_2) \\ \quad \text{then } T \\ \quad \text{else } F\end{array}$$

W_1	W_2	$W_1 \Leftrightarrow W_2$
T	T	T
T	F	F
F	T	F
F	F	T

Biconditional, cont'd.

$$
\begin{aligned}
p_1 &\stackrel{\triangle}{=} \text{Paris is in France} \\
p_2 &\stackrel{\triangle}{=} \text{York is in England} \\
p_3 &\stackrel{\triangle}{=} 2 + 2 = 5
\end{aligned}
$$

Consider

$$\mathcal{I}(p_1) \text{ is } T; \quad \mathcal{I}(p_2) \text{ is } T; \quad \mathcal{I}(p_3) \text{ is } F$$

then, e.g.,

$\mathcal{I}(p_1 \Leftrightarrow p_2)$ is T; $\quad \mathcal{I}(p_1 \Leftrightarrow p_3)$ is F; $\quad \mathcal{I}(p_3 \Leftrightarrow p_3)$ is T

Biconditional, cont'd.

$p_1 \Leftrightarrow p_2$	'Paris is in France if and only if York is in England'
	'Paris being in France is a necessary and sufficient condition for York to be in England'
	'Paris is in France provided York is in England'
	'Paris is in France exactly if York is in England'
	Paris is in France just if York is in England'

Translations of more complex sentences

$$
\begin{aligned}
p_1 &\stackrel{\triangle}{=} \text{Paris is in France} \\
p_2 &\stackrel{\triangle}{=} \text{York is in England} \\
p_3 &\stackrel{\triangle}{=} 2 + 2 = 5
\end{aligned}
$$

$(p_1 \wedge p_2) \Rightarrow \neg p_3$ ‘If Paris is in France and York is in England, then $2 + 2 \neq 5$

But, beware of ambiguity in English compound statements.

$\neg((p_1 \vee p_2) \Rightarrow p_3)$
$(p_1 \vee p_2) \Rightarrow \neg p_3$
$\neg(p_1 \Rightarrow p_2) \vee p_2$
$(p_1 \Rightarrow \neg p_3) \vee p_2$
$\neg((p_1 \Rightarrow p_3) \vee p_2)$

‘It is not the case that $2 + 2 = 5$ if Paris is in France or York is in England’

The semantics of (more complex) compound wffs

We will determine $\mathcal{I}(p_1 \wedge (p_2 \Rightarrow p_3))$ given

$$\mathcal{I}(p_1) \text{ is } T; \quad \mathcal{I}(p_2) \text{ is } F; \quad \mathcal{I}(p_3) \text{ is } T$$

$$\mathcal{I}(p_2 \Rightarrow p_3) \text{ is } T$$

$$\mathcal{I}(p_1 \wedge (p_2 \Rightarrow p_3)) \text{ is } T$$

So the whole wff is true (*for this interpretation of the atomic wffs*): this wff is true *under* this interpretation.

Given a wff, W, and an interpretation of the wff's atomic wffs, $\mathcal{I}$, we say that $\mathcal{I}$ **satisfies** W if and only if $\mathcal{I}(W)$ is T

Hence, $\mathcal{I}$ from above satisfies $p_1 \wedge (p_2 \Rightarrow p_3)$.

The semantics of (more complex) compound wffs, cont'd.

$\mathcal{I}(p_1)$ is T; $\mathcal{I}(p_2)$ is T; $\mathcal{I}(p_3)$ is F

p_1	p_2	p_3	$(p_2 \Rightarrow p_3)$	p_1	$\wedge$	$(p_2 \Rightarrow p_3)$
T	T	F	F		F	
					$\triangle$	

So the whole wff is false under this interpretation, i.e. this interpretation does not satisfy $p_1 \wedge (p_2 \Rightarrow p_3)$.

The semantics of (more complex) compound propositions, cont'd.

$\mathcal{I}(p_1)$ is F; $\mathcal{I}(p_2)$ is F; $\mathcal{I}(p_3)$ is F

p_1	p_2	p_3	p_1	$\wedge$	$(p_2$	$\Rightarrow$	$p_3)$
F	F	F	F	F	F	T	F
				$\triangle$			

So, the whole wff is false under this interpretation, i.e. this interpretation does not satisfy $p_1 \wedge (p_2 \Rightarrow p_3)$.

Truth tables

If there are n different proposition symbols in a compound wff, then there are $2 \uparrow n$ different interpretations of those atomic wffs.

p_1	p_2	p_3	p_1	$\wedge$	$(p_2 \Rightarrow p_3)$
T	T	T		T	
T	T	F		F	
T	F	T		T	
T	F	F		T	
F	T	T		F	
F	T	F		F	
F	F	T		F	
F	F	F		F	

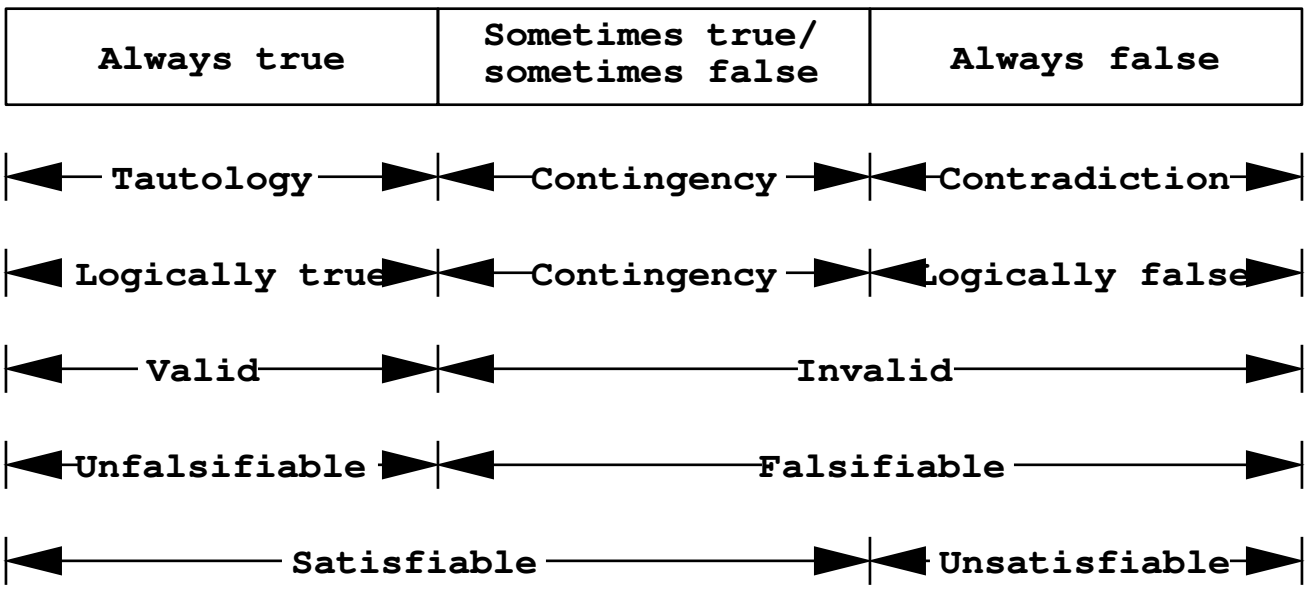
Truth tables: examples

p_1	p_2	$(p_1 \wedge p_2) \Leftrightarrow (p_1 \vee p_2)$
T	T	
T	F	
F	T	
F	F	

p	$p \vee \neg p$
T	
F	

p	$p \wedge \neg p$
T	
F	

Valid, invalid, satisfiable and unsatisfiable



Valid, invalid, satisfiable and unsatisfiable

To determine validity, invalidity, satisfiability, unsatisfiability, draw up a truth table.

E.g.

Is $(p_1 \wedge p_2) \Leftrightarrow (p_1 \vee p_2)$ valid or invalid? Is it satisfiable or unsatisfiable?

Is $p \vee \neg p$ valid or invalid? Is it satisfiable or unsatisfiable?

Is $p \wedge \neg p$ valid or invalid? Is it satisfiable or unsatisfiable?

Question: do you always need to draw up all $2 \uparrow n$ rows of the truth table to answer questions such as those given above?

Logical consequence

A wff W (the conclusion) is a **logical consequence** of a set of wffs A (the premisses) if and only if for *every* interpretation in which *all* the premisses are true, the conclusion is also true.

In other words, whenever all the wffs in A are true, then W must also be true.

Logical consequence, cont'd.

Example: q is a logical consequence of $\{p \Rightarrow q, p\}$.

p	q	$p \Rightarrow q$	p	q
T	T	T	T	T
T	F	F	T	F
F	T	T	F	T
F	F	T	F	F

Notation:

$$A \models W$$

E.g.

$$\{p \Rightarrow q, p\} \models q$$

Logical consequence: examples

Show that $\{\neg p, \neg q\} \models \neg(p \vee q)$

p	q	$\neg p$	$\neg q$	$\neg(p \vee q)$
T	T			
T	F			
F	T			
F	F			

Show that $\{p \vee q, q\} \models (\neg p \wedge q) \vee p$

p	q	$p \vee q$	q	$(\neg p \wedge q) \vee p$
T	T			
T	F			
F	T			
F	F			

Show that $\{p \Rightarrow q, q\} \not\models p$

p	q	$p \Rightarrow q$	q	p
T	T			
T	F			
F	T			
F	F			

Logical equivalence

Two wffs W_1 and W_2 are **logically equivalent** (or simply **equivalent**) if and only if $\{W_1\} \models W_2$ and $\{W_2\} \models W_1$.

$$W_1 \equiv W_2$$

To show $W_1 \equiv W_2$:

- draw up truth tables for W_1 and for W_2 and confirm that $\mathcal{I}(W_1)$ is the same as $\mathcal{I}(W_2)$ for each $\mathcal{I}$

Logical equivalence: examples

Show that $\neg(p \wedge q) \equiv \neg p \vee \neg q$

p	q	$\neg(p \wedge q)$	$\neg p \vee \neg q$
T	T		
T	F		
F	T		
F	F		

Show that $(p \Rightarrow q) \equiv \neg(p \wedge \neg q)$

p	q	$(p \Rightarrow q)$	$\neg(p \wedge \neg q)$
T	T		
T	F		
F	T		
F	F		

Laws of the algebra of propositions

For any propositions W_1, W_2 and W_3,...

Commutativity of $\wedge$ and $\vee$

$$W_1 \wedge W_2 \equiv W_2 \wedge W_1$$

$$W_1 \vee W_2 \equiv W_2 \vee W_1$$

Associativity of $\wedge$ and $\vee$

$$(W_1 \wedge W_2) \wedge W_3 \equiv W_1 \wedge (W_2 \wedge W_3)$$

$$(W_1 \vee W_2) \vee W_3 \equiv W_1 \vee (W_2 \vee W_3)$$

Laws of the algebra of propositions, cont'd.

For any propositions W_1 and W_2,...

Distributivity of $\wedge$ over $\vee$, and of $\vee$ over $\wedge$

$$W_1 \wedge (W_2 \vee W_3) \equiv (W_1 \wedge W_2) \vee (W_1 \wedge W_3)$$

$$W_1 \vee (W_2 \wedge W_3) \equiv (W_1 \vee W_2) \wedge (W_1 \vee W_3)$$

De Morgan's laws

$$\neg(W_1 \wedge W_2) \equiv \neg W_1 \vee \neg W_2$$

$$\neg(W_1 \vee W_2) \equiv \neg W_1 \wedge \neg W_2$$

Laws of the algebra of propositions, cont'd.

For any proposition W,...

Idempotence of $\wedge$ and $\vee$

$$W \wedge W \equiv W$$

$$W \vee W \equiv W$$

Identities

$$t \wedge W \equiv W$$

$$f \wedge W \equiv f$$

$$t \vee W \equiv t$$

$$f \vee W \equiv W$$

Question: what is the difference between T and t, and between F and f?

Laws of the algebra of propositions, cont'd.

For any proposition W,...

Involution

$$\neg\neg W \equiv W$$

Complement laws

$$W \wedge \neg W \equiv f$$

$$W \vee \neg W \equiv t$$

$$\neg t \equiv f$$

$$\neg f \equiv t$$

Laws for propositions: biconditionals and conditionals

Definition of biconditional

For any propositions W_1 and W_2,

$$W_1 \Leftrightarrow W_2 \equiv (W_1 \Rightarrow W_2) \wedge (W_2 \Rightarrow W_1)$$

Definition of conditional

For any propositions W_1 and W_2,

$$W_1 \Rightarrow W_2 \equiv \neg W_1 \vee W_2$$

Laws for propositions: conditionals

W_1	W_2	Conditional $W_1 \Rightarrow W_2$	Converse $W_2 \Rightarrow W_1$
T	T	T	T
T	F	F	T
F	T	T	F
F	F	T	T

W_1	W_2	Inverse $\neg W_1 \Rightarrow \neg W_2$	Contrapositive $\neg W_2 \Rightarrow \neg W_1$
T	T	T	T
T	F	T	F
F	T	F	T
F	F	T	T

Contrapositive law

For any propositions W_1 and W_1,

$$W_1 \Rightarrow W_2 \equiv \neg W_2 \Rightarrow \neg W_1$$

The laws of the algebra of propositions: their use

The laws give another 'method' for showing $W_1 \equiv W_2$

- use algebraic manipulation.

E.g. to show $p \vee q \equiv \neg(\neg p \wedge \neg q)$ algebraically.

$$
\begin{array}{lll}
\text{RHS} & = \neg(\neg p \wedge \neg q) & \\
& = \neg\neg(p \vee q) & \text{(by De Morgan's Law)} \\
& = p \vee q & \text{(by Involution)} \\
& = \text{LHS} &
\end{array}
$$

Deduction systems

Checking all interpretations is manageable for wffs that have 2 or 3 proposition symbols, but becomes increasingly difficult when there are more.

This motivates the development of apparatus that enables reasoning to take place at the purely syntactic level, before interpretations are considered.

The apparatus is called a **(formal) deduction system**, a **proof theory**, an **inference system** or a **logical calculus**.

Deduction systems, cont'd

A **deduction system** enables us to derive conclusions from a set of wffs by *syntactic operations* alone.

A deduction system comprises a finite set of **inference rules** and a set of wffs called **logical axioms**.

Inference rules

An **inference rule** comprises a set of patterns called the **conditions**, and another pattern called the **conclusion**.

E.g.

$\Rightarrow$-ELIMINATION

$$\frac{W_1, W_1 \Rightarrow W_2}{W_2}$$

1.	p	premiss
2.	$p \Rightarrow q$	premiss
3.	q	from 1,2 by $\Rightarrow$-ELIMINATION

Derivations

We allow *sequences* of inferences that might make use of the same inference rule repeatedly on the newly generated wffs:

1.	p	premiss
2.	$p \Rightarrow q$	premiss
3.	$q \Rightarrow r$	premiss
4.	q	from 1,2 by $\Rightarrow$-ELIMINATION
5.	r	from 3,4 by $\Rightarrow$-ELIMINATION

Such a sequence is called a **deduction** or a **derivation**.

If A is the set of premisses and W is the *final* conclusion, then we write

$$A \vdash W$$

We say that W is **deducible** or **derivable** from A.

The inference rules

$\wedge$-INTRODUCTION

$$\frac{W_1, W_2}{W_1 \wedge W_2}$$

$\wedge$-ELIMINATION-LEFT

$$\frac{W_1 \wedge W_2}{W_1}$$

$\wedge$-ELIMINATION-RIGHT

$$\frac{W_1 \wedge W_2}{W_2}$$

Derivation: example

E.g. show that $\{p \wedge q\} \vdash q \wedge p$

1. $p \wedge q$ premiss
2. p 1, $\wedge$-E-L
3. q 1, $\wedge$-E-R
4. $q \wedge p$ 2,3, $\wedge$-I

E.g. show that $\{p \wedge (q \wedge r)\} \vdash (p \wedge q) \wedge r$

1.
2.
3.
4.
5.
6.
7.

Inference rules, cont'd.

$\Rightarrow$-ELIMINATION

$$\frac{W_1, W_1 \Rightarrow W_2}{W_2}$$

$\vee$-INTRODUCTION-RIGHT

$$\frac{W_1}{W_1 \vee W_2}$$

$\vee$-INTRODUCTION-LEFT

$$\frac{W_2}{W_1 \vee W_2}$$

Inference rules, cont'd.

$\Leftrightarrow$-ELIMINATION-LEFT

$$\frac{W_1 \Leftrightarrow W_2, W_1}{W_2}$$

$\Leftrightarrow$-ELIMINATION-RIGHT

$$\frac{W_1 \Leftrightarrow W_2, W_2}{W_1}$$

$\neg$-ELIMINATION

$$\frac{\neg\neg W}{W}$$

Derivation: examples

Show that $\{p \wedge q\} \vdash p \vee q$

1.	$p \wedge q$	premiss
2.	p	1, $\wedge$-E-L
3.	$p \vee q$	2, $\vee$-I-R

Show that $\{p, q, (p \wedge q) \Rightarrow r\} \vdash r$

1.	p	premiss
2.	q	premiss
3.	$(p \wedge q) \Rightarrow r$	premiss
4.	$p \wedge q$	1,2, $\wedge$-I
5.	r	3,4, $\Rightarrow$-E

Subderivations

$\Rightarrow$-INTRODUCTION

$$\frac{\begin{array}{c}W_1\\ \hline W_2\end{array}}{W_1 \Rightarrow W_2}$$

Example: we show that $\{p \Rightarrow q, q \Rightarrow r\} \vdash p \Rightarrow r$

1. $p \Rightarrow q$ premiss
2. $q \Rightarrow r$ premiss
3. Subderivation:
   - 3.1 p assumption
   - 3.2 q 1, 3.1, $\Rightarrow$-E
   - 3.3 r 2, 3.2, $\Rightarrow$-E
4. $p \Rightarrow r$ 3, $\Rightarrow$-I

Subderivations: example

Show that $\{p \Rightarrow q, p \Rightarrow r\} \vdash p \Rightarrow (q \wedge r)$

1.
2.
3.
    3.1
    3.2
    3.3
    3.4
4.

Inference rules, cont'd.

$\vee$-ELIMINATION

$$\frac{W_1 \vee W_2, \frac{W_1}{W_3}, \frac{W_2}{W_3}}{W_3}$$

$\Leftrightarrow$-INTRODUCTION

$$\frac{\frac{W_1}{W_2}, \frac{W_2}{W_1}}{W_1 \Leftrightarrow W_2}$$

$\neg$-INTRODUCTION

$$\frac{\frac{W_1}{W_2 \wedge \neg W_2}}{\neg W_1}$$

One final rule, which is not really essential, but can make derivations easier to read is:

REPETITION

$$\frac{W}{W}$$

Proofs

A proof is a derivation from no premisses.

E.g. prove that $\vdash p \Rightarrow p$

1. Subderivation:
   1.1 p assumption
   1.2 p 1.1, repetition
2. $p \Rightarrow p$ 1, $\Rightarrow$-I

Soundness and completeness: soundness

At least two things may go wrong.

An inference rule might license an inference that is not a logical consequence. For example,

$\Rightarrow$-DUFF

$$\frac{W_1 \Rightarrow W_2, W_2}{W_1}$$

An inference rule of this kind is said to be **unsound**. A deduction system that contains such a rule is **unsound**.

An inference rule is **sound** if the conclusions one can infer from any set of wffs using the rule are logical consequences of the set of wffs.

A deduction system is sound if it contains only sound inference rules.

Soundness and completeness: completeness

Another thing that may go wrong is that there may be logical consequences of a set of wffs that the deduction system fails to derive.

For example, we know that $\{p \vee q, \neg p\} \models q$. But if our deduction system has insufficient inference rules or insufficiently 'powerful' inference rules, it may be that we cannot show that $\{p \vee q, \neg p\} \vdash q$. In this case, the deduction system is said to be **incomplete**.

A deduction system is **complete** if every logical consequence of any set of wffs can also be derived from the set of wffs.

Soundness and completeness: summary

A deduction system is **sound** so long as for any set of wffs A and any wff W,

$$\text{if } A \vdash W \text{ then } A \models W$$

A deduction system is **complete** so long as for any set of wffs A and any wff W,

$$\text{if } A \models W \text{ then } A \vdash W$$

Soundness is essential, but completeness might be sacrificed in Computer Science uses of deduction systems, where the efficiency of the automated deduction system is important.

Soundness and completeness: summary

If a deduction system is sound and complete, then for any wff W,

W is valid if and only if $\vdash W$

And, for any set of wffs A and wff W,

there will be no interpretation that satisfies all the wffs in A
if and only if $A \vdash W \wedge \neg W$

(When a wff of the form $W \wedge \neg W$ can be derived from a set of wffs A, we say that A is **inconsistent**.)

FIRST-ORDER PREDICATE LOGIC

Limitations of propositional logic

1. Every elephant is grey.

2. Clyde is an elephant.

3. Therefore, Clyde is grey.

In propositional logic, e.g.,

$$\{p, q\} \not\models r$$

In predicate logic, there will also be constant symbols function symbols and predicate symbols There will also be variables to represent unspecified objects from the universe of discourse, and quantifiers, which will determine the number of objects these variables range over.

An informal introduction: constant and predicate symbols

‘Clyde is an elephant.’

‘Clyde likes Martie.’

‘Clyde likes Martie and Martie likes Clyde.’

We use **constant symbols** to denote entities.

We use **predicate symbols** to denote relations.

$$elephant(c)$$

$$likes(c, m)$$

$$likes(c, m) \wedge likes(m, c)$$

An informal introduction: function symbols

> ‘Clyde’s father is an elephant.’
>
> ‘Clyde and Martie’s mutual friend likes Clyde and Martie.’

We use **function symbols** to denote functions from tuples of entities to other entities. Functions cannot map a tuple of entities to more than one object. This is what makes the use of a function symbol suitable in translation of the phrase ‘Clyde’s father’ (Clyde will have only one father). In this regard, the ‘mutual friend’ example is, perhaps, a poor one, as some people can have several mutual friends (though use of a function symbol here means that Clyde and Martie do not).

An informal introduction: function symbols, cont'd.

elephant(fatherOf(c))

likes(mf(c, m), c) $\wedge$ *likes(mf(c, m), m)*

Question: The following two wffs both say that c's father is an elephant. Explain the different ways in which they say this.

father(c, g) $\wedge$ *elephant(g)*

elephant(fatherOf(c))

An informal introduction: variables and quantification

'Everybody is happy!' (Hoorah!)

$$\textit{happy(c)} \wedge \textit{happy(m)} \wedge \textit{happy(f)} \wedge \textit{happy(g)} \wedge$$

happy(fatherOf(g)) $\wedge$ happy(fatherOf(fatherOf(g))) $\wedge \ldots$

Such a wff is unwieldy for a large universe of discourse, and impossible for an infinite one.

$\forall$ is the symbol used for **universal quantification**: the statement holds for *all* objects in the universe of discourse.

$$\forall x \textit{happy}(x)$$

i.e. for all x, x is happy. Everybody is happy! Hoorah!

An informal introduction: variables and quantification, cont'd.

'Someone is wealthy.'

wealthy(c) $\vee$ *wealthy(m)* $\vee$ *wealthy(f)* $\vee$ *wealthy(g)* $\vee$

wealthy(fatherOf(g)) $\vee$ wealthy(fatherOf(fatherOf(g))) $\vee \ldots$

$\exists$ is the symbol used for **existential quantification**: the statement holds for *at least one* object in the universe of discourse.

$$\exists x \textit{wealthy}(x)$$

i.e. there is at least one x such that x is wealthy.

The syntax of first-order predicate logic: terms

We explain how to construct well-formed formulae (wffs) of first-order predicate logic in two steps: first, we show how to construct **terms**; then we show how to construct **wffs**.

1. Every constant symbol is a term.

2. Every variable is a term.

3. For every function symbol F of arity n and any n terms $T_1, T_2, \ldots, T_n$ then

   $$F(T_1, T_2, \ldots, T_n)$$

   is a term.

4. Nothing else is a term.

The syntax of first-order predicate logic: wffs

Now we give the rules for wffs.

1. For any predicate symbol P of arity n and any n terms $T_1, T_2, \ldots T_n$ then

   $$P(T_1, T_2, \ldots T_n)$$

   is a wff. (Such wffs are called **atomic wffs**, or simply **atoms**.)

2. For any variable X and any wff W

   $$\forall X W$$

   is a wff, and

   $$\exists X W$$

   is a wff. (Such wffs are called **quantified wffs**.)

The syntax of first-order predicate logic: wffs, cont'd.

3. For any wffs W, W_1 and W_2, the following are also wffs:

$$\neg W$$

$$W_1 \wedge W_2 \qquad W_1 \vee W_2$$

$$W_1 \Rightarrow W_2 \qquad W_1 \Leftrightarrow W_2$$

4. Any wff may be enclosed in parentheses and will still be a wff.

5. No other string of symbols is a wff.

The syntax of first-order predicate logic: example wffs

$$\mathit{likes(c,\ m)}$$

$$\mathit{likes(c,\ fatherOf(m))} \Rightarrow \mathit{elephant(c)}$$

$$\mathit{likes(mf(fatherOf(m),\ fatherOf(c)),\ c)}$$

$$\forall x\ \mathit{likes}(x, x)$$

$$\forall x(\mathit{elephant}(x) \Rightarrow \neg \mathit{likes}(x, m))$$

$$\forall x(\mathit{elephant}(x) \Rightarrow \exists y(\mathit{mouse}(y) \wedge \neg \mathit{likes}(x, y)))$$

$$\mathit{mouse}(x)$$

$$\forall x\ \mathit{likes}(y, y)$$

$$\mathit{elephant}(x) \wedge \exists x \mathit{mouse}(x)$$

$$(\forall x \mathit{elephant}(x)) \Rightarrow \exists y(\mathit{mouse}(y) \wedge \neg \mathit{likes}(x, y))$$

$$\forall y(\mathit{elephant}(y) \Rightarrow \exists y(\mathit{mouse}(y) \wedge \neg \mathit{likes}(y, y)))$$

Precedence and associativity

- Parentheses determine evaluation order.
- Precedence rules are:

  Highest $\quad \forall \exists \neg \vee \wedge \Rightarrow \Leftrightarrow \quad$ Lowest

- Associativity rules are as before.

E.g. by the precedence rules,

$$\forall x \ \textit{elephant}(x) \Rightarrow \exists y \ \textit{mouse}(y)$$

evaluates as

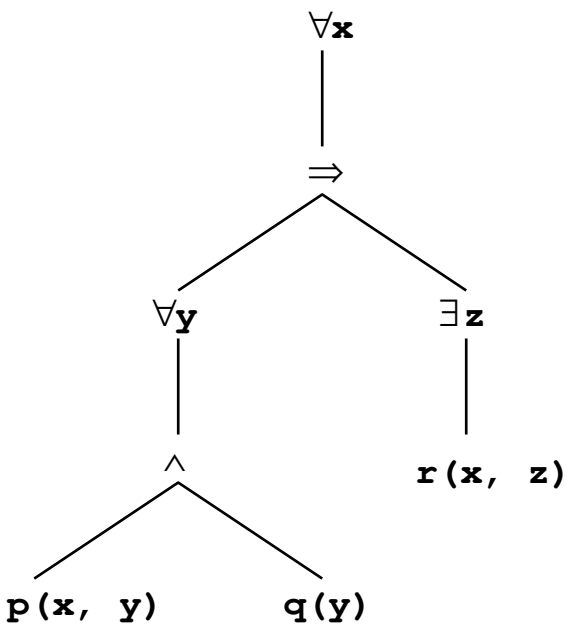
$$(\forall x \ \textit{elephant}(x)) \Rightarrow (\exists y \ \textit{mouse}(y))$$

Scope of a quantifier

If a wff contains a subwff of the form $\forall XW$ or $\exists XW$, then W is called the **scope** of this particular quantifier.

Example

$$\underline{\forall x(\underline{\forall y(p(x,y) \wedge q(y))} \Rightarrow \underline{\exists z r(x,z)})}$$



Free and bound variables

The occurrence of a variable adjacent to a quantifier is its **binding occurrence**.

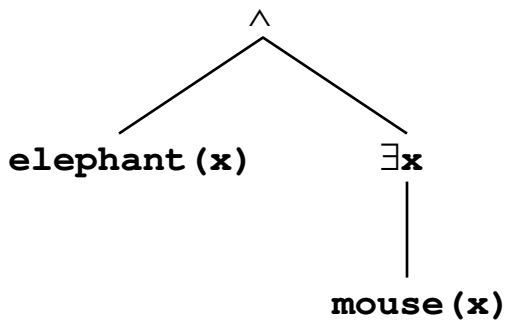
All other occurrences are **applied occurrences**.

If a wff contains a subwff of the form $\forall X W$ or $\exists X W$, then applied occurrences of X in W are said to be **bound**.

Other applied occurrences of variables are said to be **free**.

Scope, and free and bound variables: example

$$\textit{elephant}(x) \wedge \exists x \textit{mouse}(x)$$



A wff with no free variables is called a **sentence**.

Scope, and free and bound variables: examples

$$\textit{likes(c, m)}$$

$$\textit{likes(c, fatherOf(m))} \Rightarrow \textit{elephant(c)}$$

$$\textit{likes(mf(fatherOf(m), fatherOf(c)), c)}$$

$$\forall x\ \textit{likes}(x, x)$$

$$\forall x(\textit{elephant}(x) \Rightarrow \neg \textit{likes}(x, m))$$

$$\forall x(\textit{elephant}(x) \Rightarrow \exists y(\textit{mouse}(y) \wedge \neg \textit{likes}(x, y)))$$

$$\textit{mouse}(x)$$

$$\forall x\ \textit{likes}(y, y)$$

$$(\forall x \textit{elephant}(x)) \Rightarrow \exists y(\textit{mouse}(y) \wedge \neg \textit{likes}(x, y))$$

$$\forall y(\textit{elephant}(y) \Rightarrow \exists y(\textit{mouse}(y) \wedge \neg \textit{likes}(y, y)))$$

The semantics of first-order predicate logic

Since this logic is still concerned with the analysis of English *statements*, it is still the case that a wff will be either true or false.

In predicate logic, the meaning of the lowest level expressions (constant symbols, function symbols and predicate symbols) will be *stipulated* by an *interpretation*. Then, the meaning of a wff will be determined by the meanings of its subexpressions (in this case, its terms and subwffs).

We assume that there is some set of entities (possibly an infinite set) which we refer to as the **universe of discourse**, and which we denote as $\mathcal{U}$.

It is obviously possible to define a large number of relations *on* $\mathcal{U}$.

Universe of discourse: example

The Books World

- Entities: Here we have five entities (excluding the table): Down&Out, Zen, Cuckoo, Jude and Spire.

- Functions: Here we have such functions as the unary function that maps from a book to the book directly on top of it.

- Relations: Here we have such relations as the binary relation that tells us whether a pair of books is such that the second is *somewhere* above the first, and the unary relation that tells us which books have a clear top.

Interpretations

In predicate logic, an interpretation $\mathcal{I}$ makes the following mappings:

1. For every constant symbol C, $\mathcal{I}(C)$ is an entity from $\mathcal{U}$.

2. For every n-ary function symbol F, $\mathcal{I}(F)$ is an n-ary function on $\mathcal{U}$, i.e. one that maps n-tuples to an entity.

3. For every n-ary predicate symbol P, $\mathcal{I}(P)$ is an n-ary relation on $\mathcal{U}$, i.e. one that holds of n-tuples.

Interpretation: example

For constant symbols a, b, c, d, e, a unary function symbol f and the binary predicate symbols p and q, we give an interpretation $\mathcal{I}_1$:

$\mathcal{I}_1(a)$ is (Down&Out)

$\mathcal{I}_1(b)$ is (Zen)

$\mathcal{I}_1(c)$ is (Cuckoo)

$\mathcal{I}_1(d)$ is (Jude)

$\mathcal{I}_1(e)$ is (Spire)

Interpretation: example, cont'd.

$\mathcal{I}_1(f)$ is	the unary function that maps a book to the book *directly on top* of it
$\mathcal{I}_1(p)$ is	the binary relation whose graph/extension contains pairs of books where the second book of the pair is *directly on top* of the first book of the pair
$\mathcal{I}_1(q)$ is	the binary relation whose graph/extension contains pairs of books where the second book of the pair is *somewhere above* the first book of the pair

Interpretation: example

Here is another example interpretation, $\mathcal{I}_2$, for the same universe of discourse and the same symbols. $\mathcal{I}_2$ is identical to $\mathcal{I}_1$ except:

$\mathcal{I}_2(a)$ is (Zen)

$\mathcal{I}_2(b)$ is (Down&Out)

$\mathcal{I}_2(p)$ is the binary relation whose graph/extension contains pairs of books where the second book of the pair is *directly below* the first book of the pair

$\mathcal{I}_2(q)$ is the binary relation whose graph/extension contains pairs of books where the second book of the pair is *directly on top* of the first book of the pair

The semantics of first-order predicate logic, cont'd.

The semantics of terms

- For any constant C, $\mathcal{I}(C)$ is stipulated.

- For any variable X, the semantics will be dealt with below, when we consider quantified wffs.

- For any n-ary function symbol F with arguments $T_1, T_2, \ldots, T_n$, then

  $\mathcal{I}(F(T_1, T_2, \ldots, T_n))$ is
  $\mathcal{I}(F)(\mathcal{I}(T_1), \mathcal{I}(T_2), \ldots, \mathcal{I}(T_n))$

  i.e. you get the function that corresponds to F (as stipulated by $\mathcal{I}$), you work out the objects that correspond to $T_1, T_2, \ldots, T_n$, and then you apply the function to these objects. This gets you an object, and this object is the meaning of the term $F(T_1, T_2, \ldots, T_n)$.

The semantics of first-order predicate logic, cont'd.

The semantics of atomic wffs

- For any atomic wff comprising an n-ary predicate symbol P with arguments $T_1, T_2, \ldots, T_n$ then

  $\mathcal{I}(P(T_1, T_2, \ldots, T_n))$ is true if and only if
  $\langle \mathcal{I}(T_1), \mathcal{I}(T_2), \ldots, \mathcal{I}(T_n) \rangle \in \mathcal{I}(P)$.
  Otherwise, it is false.

  I.e. you get the relation that corresponds to P (as stipulated by $\mathcal{I}$), you work out the objects that correspond to $T_1, T_2, \ldots, T_n$, and you see whether this n-tuple of objects is an element of the graph/extension of the relation.

The semantics of first-order predicate logic, cont'd

The semantics of quantified wffs

- For any quantified wff $\forall XW$, $\mathcal{I}(\forall XW)$ is true if and only if $\mathcal{I}(W)$ is true for all assignments of entities in $\mathcal{U}$ to X. Otherwise, it is false.

- For any quantified wff $\exists XW$, $\mathcal{I}(\exists XW)$ is true if and only if $\mathcal{I}(W)$ is true for at least one assignment of an entity in $\mathcal{U}$ to X. Otherwise, it is false.

The semantics of compound wffs

- For any compound wff $\neg W, W_1 \wedge W_2, W_1 \vee W_2$, $W_1 \Rightarrow W_2$ and $W_1 \Leftrightarrow W_2$, determine $\mathcal{I}(W), \mathcal{I}(W_1)$ and $\mathcal{I}(W_2)$ and use these truth values as before (as in propositional logic) to determine the truth value of the compound wff.

Semantics: examples on the 'Books World'

We will determine the truth value of the following wffs under interpretation $\mathcal{I}_1$.

- $p(a, b)$

- $p(a, b) \vee q(c, f(b))$

Semantics: more examples on the 'Books World'

- $\exists x\, p(c, x)$
  - We know that $\mathcal{I}(c)$ is Cuckoo.
  - Take $\mathcal{I}_1(x)$ to be Down&Out.
    - Is $\langle$ Cuckoo, Down&Out $\rangle \in \mathcal{I}_1(p)$?
    - No.
  - Take $\mathcal{I}_1(x)$ to be Zen.
    - Is $\langle$ Zen, Cuckoo $\rangle \in \mathcal{I}_1(p)$?
    - Yes.
  - We have found an assignment of an entity to x that makes $p(c, x)$ true. Existential quantification only requires at least one such assignment, so the wff as a whole is true.

Semantics: more examples on the 'Books World'

- $\forall x\, p(c, x)$
  - Take $\mathcal{I}_1(x)$ to be Down&Out.
    - Is $\langle$ Cuckoo, Down&Out $\rangle \in \mathcal{I}_1(p)$?
    - No.
  - Universal quantification requires that every assignment make $p(c, x)$ true. Since we have found an assignment that makes this false, there is no point in checking other assignments. The wff as a whole is false.

Semantics: another example on the ‘Books World’

We will use interpretation $\mathcal{I}_2$ and evaluate

$\mathcal{I}_2(\forall x \forall y(q(x, y) \Rightarrow p(y, x))$

From English statements to predicate logic

The English word ‘every’ generally signals use of universal quantification, and the English word ‘some’ generally signals use of existential quantification.

‘Every elephant is happy’

$$\forall x(\textit{elephant}(x) \Rightarrow \textit{happy}(x))$$

i.e. for every x, if x is a elephant, then x is happy.

‘Some elephant is happy’

$$\exists x(\textit{elephant}(x) \wedge \textit{happy}(x))$$

i.e. there exists an x such that x is a elephant and x is happy.

The question we want to explore is: why does the universal example use $\Rightarrow$ and the existential example use $\wedge$?

From English statements to predicate logic, cont'd.

Suppose you translated 'Every elephant is happy' as $\forall x(\textit{elephant}(x) \wedge \textit{happy}(x))$.

This translation is too strong.

Consider the universe $\{$Clyde, Flopsy$\}$. Let $\mathcal{I}(\textit{elephant})$ be the unary relation true of Clyde but not of Flopsy and let $\mathcal{I}(\textit{happy})$ be the unary relation true of Clyde (and possibly of Flopsy: it doesn't matter).

Now, intuitively, you'd agree that in the world just described every elephant is happy: the English statement is true.

But $\forall x(\textit{elephant}(x) \wedge \textit{happy}(x))$ will be false in this world.

From English statements to predicate logic, cont'd

Suppose you translated 'Some elephant is happy' as $\exists x(\textit{elephant}(x) \Rightarrow \textit{happy}(x))$

This translation is too weak.

Consider the universe $\{$Clyde, Flopsy$\}$. Let $\mathcal{I}$(*elephant*) be the unary relation true of Clyde but not of Flopsy, and let $\mathcal{I}$(*happy*) be the unary relation which might be true of Flopsy (it doesn't matter) but is *not* true of Clyde.

Now, intuitively, you'd agree that in the world just described there isn't a happy elephant: the English statement is false.

But $\exists x(\textit{elephant}(x) \Rightarrow \textit{happy}(x))$ will be true in this world.

Vacuous quantification

Suppose we have the following universe of discourse $\{$Clyde, Dumbo$\}$. Let the unary predicate *elephant* be the relation true of both Clyde and Dumbo, the unary predicate *mouse* be true of neither, the unary predicate *happy* be true of neither and the unary predicate *canFly* be true of Dumbo but not of Clyde.

Consider 'Every happy elephant can fly'.

$$\forall x((\textit{happy}(x) \wedge \textit{elephant}(x)) \Rightarrow \textit{canFly}(x))$$

Is this true or false?

Since the conditional is true for all assignments of objects to x, the wff as a whole is true. Does this tally with your intuitions?

Vacuous quantification, cont'd.

Our intuitions are that

- if there are no happy elephants, then the statement that every happy elephant can fly is false;

- if there are no mice, then the statement that every mouse can fly is false.

But our translations are wffs that are true in these cases.

From English statements to predicate logic, cont'd

We have said that the English word 'every' is generally translated by $\forall$. Similarly, the words 'all' and 'each' are also translated by $\forall$:

'All creatures are mortal.'

'Bill has attended every practical.'

'Each house has a roof.

But be careful, some uses of 'all' and, more rarely, some uses of 'every' do not translate as $\forall$:

'All the children play together'

'Every house has a different colour.'

From English statements to predicate logic, cont'd

We have said that 'some' is generally translated by $\exists$. Similarly, the words and phrases 'at least one', 'a', 'an', 'there is', 'there are' and, perhaps more controversially, 'a few', 'one or more' and 'a number of' are also translated by $\exists$:

'Some Computer Scientist is a poet.'

'Some Computer Scientists are poets.'

But, be careful because some uses of 'a' (or 'an') signal universal quantification:

'A country with no monarch is a republic.'

'A student who fails is offered a resit.'

From English statements to predicate logic, cont'd.

'Any' in unnegated sentences usually signals universal quantification:

> 'Anyone who cheats will be thrown out.'

'Any' in negated sentences usually gives a choice:

> 'Clyde didn't eat any buns.'

Consider plural nouns.

> 'Students who fail will be excluded from the University.'

> 'Clyde attends practicals (so he'll be safe in the exam).'

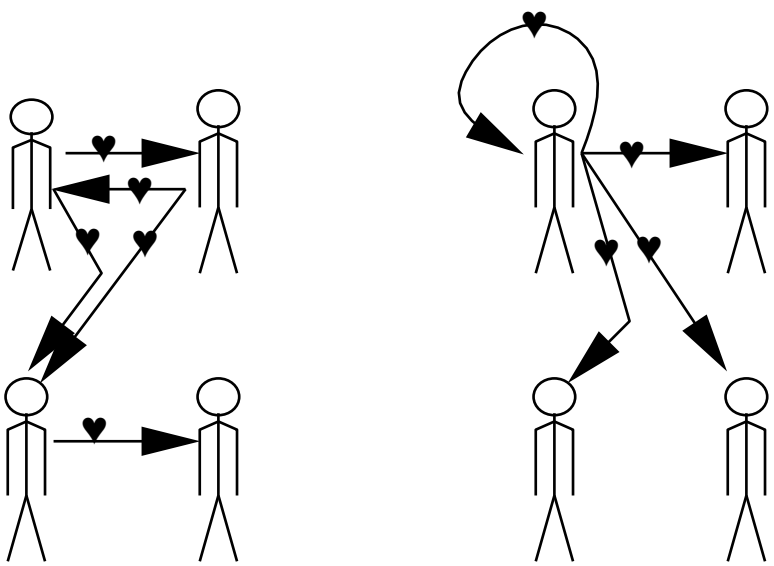
Scope ambiguities

‘Everyone is loved by someone.’

$\forall x \exists y$ *loves(y, x)*

$\exists y \forall x$ *loves(y, x)*

The first translation is true in the left-hand situation; both are true in the right-hand situation:



Validity and satisfiability

- An interpretation $\mathcal{I}$ satisfies a wff W if $\mathcal{I}(W)$ is T.

- A wff is satisfiable if there is at least one interpretation that satisfies it. Otherwise it is **unsatisfiable**.

- A wff is **valid** if all interpretations satisfy it. Otherwise it is **invalid**.

Logical consequence and logical equivalence

- A wff W is a **logical consequence** of a set of wffs A if and only if every interpretation that satisfies all the wffs in A also satisfies W.

- Two wffs W_1 and W_2 are **logically equivalent** (or simply **equivalent**) if and only if $\{W_1\} \models W_2$ and $\{W_2\} \models W_1$.

Example of a logical consequence:

$\{\forall x(\textit{elephant}(x) \Rightarrow \textit{grey}(x)), \textit{elephant(clyde)}\} \models$
grey(clyde)

Laws for quantifiers

We can define $\exists$ in terms of $\forall$ and vice versa.

Definition of existential

$$\exists X W \equiv \neg \forall X \neg W$$

Definition of universal

$$\forall X W \equiv \neg \exists X \neg W$$

Substitution of variables

Given a wff W_1 containing a variable X_1, the wff W_2 which is just like W_1 except that X_2 has been substituted for some or all occurrences of X_1, then $W_1 \equiv W_2$ provided the substitution changes no bindings

Logical equivalences: example

'No duck likes Clyde.'

$$\neg\exists x(\mathit{duck}(x) \wedge \mathit{likes}(x, c)) \equiv \forall x(\mathit{duck}(x) \Rightarrow \neg\mathit{likes}(x, c))$$

$$\begin{aligned}
\text{LHS} &\equiv \neg\exists x(\mathit{duck}(x) \wedge \mathit{likes}(x, c)) \\
&\equiv \neg\neg\forall x\neg(\mathit{duck}(x) \wedge \mathit{likes}(x, c)) \\
&\equiv \forall x\neg(\mathit{duck}(x) \wedge \mathit{likes}(x, c)) \\
&\equiv \forall x(\neg\mathit{duck}(x) \vee \neg\mathit{likes}(x, c)) \\
&\equiv \forall x(\mathit{duck}(x) \Rightarrow \neg\mathit{likes}(x, c)) \\
&\equiv \text{RHS}
\end{aligned}$$

A problem

The problem with the definitions of satisfiability and particularly of unsatisfiability, validity, logical consequence and logical equivalence is that they require us to test *every* interpretation.

For predicate logic, there is an infinite number of interpretations, some with infinite domains.

So, in predicate logic, the need for a deduction system, which works at a syntactic level, is now very strong.

Deduction system

The definition of inference rule, derivation, proof, soundness and completeness, etc. are all as before.

∃-INTRODUCTION

$$\frac{W(t)}{\exists x W(x)}$$

∃-ELIMINATION

$$\frac{\exists x W_1(x), \frac{W_1(a)}{W_2}}{W_2}$$

(so long as a does not occur in $W_1(x)$ or in W_2 or in a premiss, or in any assumption that is current other than $W_1(a)$)

Deduction system, cont'd.

$\forall$-INTRODUCTION

$$\frac{W(a)}{\forall x W(x)}$$

(so long as a does not occur in $W(x)$, in a premiss or in any assumption that is current)

$\forall$-ELIMINATION

$$\frac{\forall x W(x)}{W(t)}$$