# Discrete Mathematics

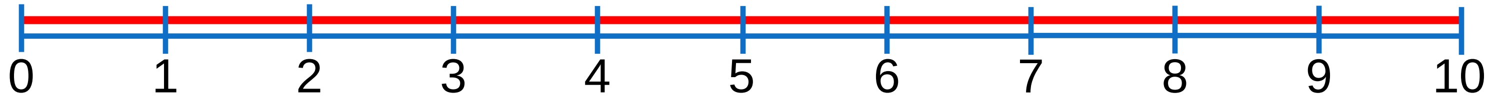
- What is discrete mathematics?

  "Discrete mathematics describes processes that consist of a sequence of individual steps.

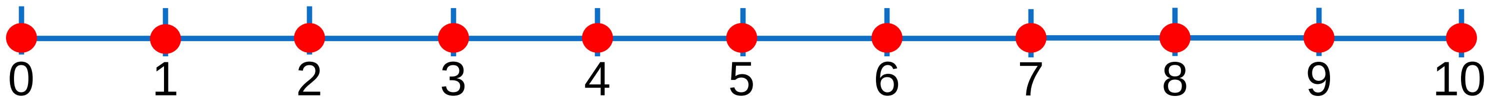
  This contrasts with calculus, which describes processes that change in a continuous fashion."

  -- Susanna S. Epp, "Discrete Mathematics with Applications", 4th edition

Continuous




Discrete



# Discrete Mathematics

- What are covered in discrete mathematics
  - Logic and proof
  - Induction and Recursion
  - Set Theory
  - Functions
  - Relations
  - Counting and Probability
  - Graphs and Trees
  - Analysis of Algorithm Efficiency

# Discrete Mathematics

- Why is it important in computer science?
  - Computers use discrete structures to represent and manipulate data. Discrete math is the mathematics of computing.

# Discrete Mathematics

- Why is it important in computer science?
  - Computer Science is not (only) programming. Computer Science is about problem solving!
  - Mathematics is at the heart of problem solving.
    - To define the problem
    - To derive a solution
    - To analyse an algorithm to justify its correctness or efficiency
  - Example:

“A new hotel has 100 rooms (001 to 100). The decorator needs to put room numbers on all the doors. How many 1’s does he need to buy?”

| 1 | 0 | 0 |
|---|---|---|

1

| 0 | x | x |
|---|---|---|

Case 1: both are 1’s, ‘**011**’, 2

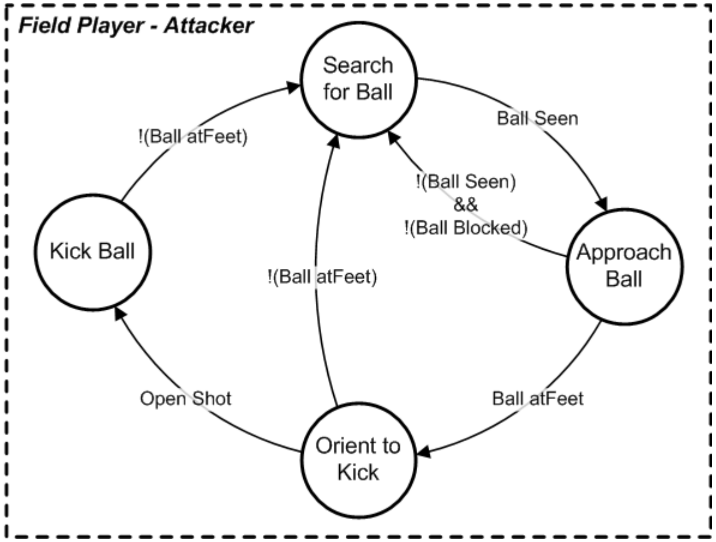
Case 2: only one 1, C(2,1) x C(9,1) = 2 x 9 = 18

1 + 2 + 18 = 21 ← The Sum Rule

↑ The Product Rule

# Discrete Mathematics

- Why is it important in computer science?
  - The concepts in discrete mathematics are useful in all branches of computer science: databases, cyber security, computer organisation, compilers, programming languages, artificial intelligence, …
  - Robot soccer



# Discrete Mathematics

- Why is it important in computer science?
  - Abstract thinking

  "…It makes you use your brain in ways no other classes do. It is a logical thinking class…"

  "…It brings rational clarity to your solutions and a formal way of analysing a problem…"

  https://softwareengineering.stackexchange.com/questions/163168/how-important-is-discrete-mathematics-for-a-computer-scientist

Discrete Mathematics is fun!

CM1103 PROBLEM SOLVING WITH PYTHON

# LOGIC

## AN INTRODUCTION

Dr Jing Wu

School of Computer Science & Informatics

Cardiff University

# Propositional logic

| Definition |
|---|
| A proposition is a sentence declaring a fact that is either ***True*** or ***False***. |

| Example | |
|---|---|
| Stuart Allen is the module leader of CM1103. | |
| Cardiff is the capital of France. | |
| $8 + 2 = 10$ | |
| $8 + 2 = 15$ | |
| Is it Tuesday? | |
| Read this. | |
| $n < 5$ | |
| If $n$ is 4, then $n < 5$ | |

# Combining propositions

**Definition**

Compound propositions are formed from other propositions using logical operators.

**Logical operators**

| | |
|---|---|
| Negation | $\neg$ |
| Conjunction | $\wedge$ |
| Disjunction | $\vee$ |
| Exclusive or | $\oplus$ |
| Conditional / Implication | $\Rightarrow$ |
| Biconditional | $\Leftrightarrow$ |

# Negation $\neg$

**Definition**

Let $p$ be a proposition. The *negation* of $p$, denoted $\neg p$, is the statement "*It is not the case that* $p$".

**Example**

| $p$ | $\neg p$ |
|---|---|
| Cardiff is the capital of France. | |
| 8 + 2 = 10 | |
| My hard drive has at least 500 GB. | |
| | |

# Conjunction $\wedge$ and disjunction $\vee$

**Definition**

Let $p$ and $q$ be propositions. The *conjunction* of $p$ and $q$, denoted $p \wedge q$, is the proposition "$p$ *and* $q$". It is true if both $p$ and $q$ are true, and false otherwise.

**Definition**

Let $p$ and $q$ be propositions. The *disjunction* of $p$ and $q$, denoted $p \vee q$, is the proposition "$p$ *or* $q$". It is true if either $p$ or $q$ (or both) are true, and false otherwise.

**Example**

Let $p$ be the proposition "*Cardiff is the capital of France*", and let $q$ be the proposition 2 + 2 = 4.

| $p$ | $q$ | $p \wedge q$ | $p \vee q$ | $\neg p \wedge q$ |
|---|---|---|---|---|
| | | | | |

# Truth tables

A truth table gives the truth value of a compound proposition for all possible combinations of truth-values of the individual propositions.

Each row shows one possible combination.

| × | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 2 | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 | 18 | 20 | 22 | 24 |
| 3 | 3 | 6 | 9 | 12 | 15 | 18 | 21 | 24 | 27 | 30 | 33 | 36 |
| 4 | 4 | 8 | 12 | 16 | 20 | 24 | 28 | 32 | 36 | 40 | 44 | 48 |
| 5 | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 |
| 6 | 6 | 12 | 18 | 24 | 30 | 36 | 42 | 48 | 54 | 60 | 66 | 72 |
| 7 | 7 | 14 | 21 | 28 | 35 | 42 | 49 | 56 | 63 | 70 | 77 | 84 |
| 8 | 8 | 16 | 24 | 32 | 40 | 48 | 56 | 64 | 72 | 80 | 88 | 96 |
| 9 | 9 | 18 | 27 | 36 | 45 | 54 | 63 | 72 | 81 | 90 | 99 | 108 |
| 10 | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 | 110 | 120 |
| 11 | 11 | 22 | 33 | 44 | 55 | 66 | 77 | 88 | 99 | 110 | 121 | 132 |
| 12 | 12 | 24 | 36 | 48 | 60 | 72 | 84 | 96 | 108 | 120 | 132 | 144 |

**VS**

$p$

| ∧ | T | F |
|---|---|---|
| T | T | F |
| F | F | F |

$q$

| $p$ | $q$ | $p \land q$ |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | F |
| F | F | F |

# Truth tables

| $p$ | $\neg p$ |
|---|---|
| T | F |
| F | T |

| $p$ | $q$ | $p \land q$ |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | F |
| F | F | F |

| $p$ | $q$ | $p \lor q$ |
|---|---|---|
| T | T | T |
| T | F | T |
| F | T | T |
| F | F | F |

# Truth tables

Add operands;

Compare different compound propositions.

| $p$ | $q$ | $r$ | $p \land q$ | $(p \land q) \land r$ | $(p \land q) \lor r$ |
|---|---|---|---|---|---|
| T | T | T | T | T | T |
| T | T | F | T | F | T |
| T | F | T | F | F | T |
| T | F | F | F | F | F |
| F | T | T | F | F | T |
| F | T | F | F | F | F |
| F | F | T | F | F | T |
| F | F | F | F | F | F |

# Truth tables

Construct truth tables for complex propositions, e.g.,

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

# Short circuit evaluation

For some Boolean operators in Python (and in some other programming languages as well), the second argument is executed or evaluated only if the first argument does not suffice to determine the value of the expression:

- When the first argument of the AND function evaluates to *false*, the overall value must be *false*;
- When the first argument of the OR function evaluates to *true*, the overall value must be *true*.

| $p$ | $q$ | $p \land q$ |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | F |
| F | F | F |

| $p$ | $q$ | $p \lor q$ |
|---|---|---|
| T | T | T |
| T | F | T |
| F | T | T |
| F | F | F |

# Exclusive or $\oplus$

**Definition**

Let $p$ and $q$ be propositions. The *exclusive or* of $p$ and $q$, denoted $p \oplus q$, is the proposition that is true if exactly one of $p$ and $q$ are true, and false otherwise.

| $p$ | $q$ | $p \oplus q$ |
|---|---|---|
| T | T | F |
| T | F | T |
| F | T | T |
| F | F | F |

**Inclusive vs. Exclusive Or**

| | |
|---|---|
| I'd like some tea ***or*** coffee. | |
| I'd like some milk ***or*** sugar with my tea. | |

# Translate English into Logic

Example:

Let $p$ be: Today is hot.

Let $q$ be: Today is sunny.

Translate:

Today it is hot, but not sunny.

Today it is neither hot nor sunny.

## Knights & Knaves example

A very special island is inhabited only by knights and knaves.
Knights always tell the truth, and knaves always lie.

You meet two inhabitants: Zoey and Mel. Zoey tells you that Mel is a knave.
Mel says, “Neither Zoey nor I are knaves.”

Can you determine who is a knight and who is a knave?

More Knights and Knaves puzzles

http://philosophy.hku.hk/think/logic/knights.php

The key is to translate their statements into logic.

Logic operators may not be obvious.

Always ask yourself what's the statement *'in other words'*

# Implication

**Definition**

Let $p$ and $q$ be propositions. The *implication* or *conditional statement* $p \Rightarrow q$, is the proposition "if $p$ then $q$". We call $p$ the hypothesis, $q$ the conclusion. It is false if $p$ is true and $q$ is false, and true otherwise. It has the truth table:

| $p$ | $q$ | $p \Rightarrow q$ |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | T |
| F | F | T |

Think it as a promise.

When the promise is broken.

**Example**

| | |
|---|---|
| If you tidy your room, you will get a candy. | $p \Rightarrow$ q |

| Examples: Are these implications true or false? | |
|---|---|
| If $4 \times 4 = 16$, then $(-1) \times (-1) = 1$. | |
| If $2 + 2 = 4$, then all dogs are black. | |
| If $2 + 2 = 5$, then all dogs are black. | |
| If it rains today, then 2 is a prime number. | |

# Contrapositive

**Definition (Contrapositive)**

The *contrapositive* of the implication $p \Rightarrow q$ is $\neg q \Rightarrow \neg p$.

**Example**

| $p \Rightarrow q$ | $\neg q \Rightarrow \neg p$ |
|---|---|
| If you tidy your room, you will get a candy. | |
| If you get full marks on CM1103 exam, you will pass the module. | |

| $p$ | $q$ | $p \Rightarrow q$ | $\neg q$ | $\neg p$ | $\neg q \Rightarrow \neg p$ |
|---|---|---|---|---|---|
| T | T | T | | | |
| T | F | F | | | |
| F | T | T | | | |
| F | F | T | | | |

# Converse

**Definition (Converse)**

The *converse* of the implication $p \Rightarrow q$ is $q \Rightarrow p$.

**Example**

| $p \Rightarrow q$ | $q \Rightarrow p$ |
|---|---|
| If you get full marks on CM1103 exam, you will pass the module. | |

| $p$ | $q$ | $p \Rightarrow q$ | $q \Rightarrow p$ |
|---|---|---|---|
| T | T | T | |
| T | F | F | |
| F | T | T | |
| F | F | T | |

# Biconditional

**Definition**

Let $p$ and $q$ be propositions. The *biconditional of* $p$ *and* $q$ is the proposition "$p$ if, and only if, $q$" and is denoted $p \Leftrightarrow q$. It is true if both $p$ and $q$ have the same truth values and is false if $p$ and $q$ have opposite truth values.

| $p$ | $q$ | $p \Leftrightarrow q$ | $p \Rightarrow q$ | $q \Rightarrow p$ | $(p \Rightarrow q) \land (q \Rightarrow p)$ |
|---|---|---|---|---|---|
| T | T | T | | | |
| T | F | F | | | |
| F | T | F | | | |
| F | F | T | | | |

# Tautologies and contradictions

**Definition (Tautology)**

A *tautology* is a compound proposition that is always true.

**Definition (Contradiction)**

A *contradiction* is a compound proposition that is always false.

**Example**

| Tautology | Contradiction |
|---|---|
| | |
| | |
| | |

# Logical equivalence $\equiv$

**Definition**

Compound propositions $p$ and $q$ are *logically equivalent* if, and only if, $p$ and $q$ have the same truth table. The logical equivalence of $p$ and $q$ is denoted by $p \equiv q$.

## Logical equivalence: example

Prove that propositions $p \Rightarrow q$ and $\neg p \lor q$ are logically equivalent.

# De Morgan's laws

**De Morgan's laws**

Let $p$ and $q$ be propositions. Then:

$\neg(p \vee q)$ is logically equivalent to $\neg p \wedge \neg q$

$\neg(p \wedge q)$ is logically equivalent to $\neg p \vee \neg q$

**Example**

Given a day $d$ and month $m$, when is it not April 23$^{\text{rd}}$?

# Logical equivalences

| | | |
|---|---|---|
| **Commutative** | $p \wedge q \equiv q \wedge p$ | $p \vee q \equiv q \vee p$ |
| **Associative** | $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$ | $(p \vee q) \vee r \equiv p \vee (q \vee r)$ |
| **Distributive** | $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$ | $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$ |
| **Identity** | $p \wedge T \equiv p$ | $p \vee F \equiv p$ |
| **Negation** | $p \vee \neg p \equiv T$ | $p \wedge \neg p \equiv F$ |
| **Idempotent** | $p \wedge p \equiv p$ | $p \vee p \equiv p$ |
| **De Morgan** | $\neg(p \vee q) \equiv \neg p \wedge \neg q$ | $\neg(p \wedge q) \equiv \neg p \vee \neg q$ |
| **Double negation** | $\neg(\neg p) \equiv p$ | |

# Using equivalences

**Example**

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

**Example**

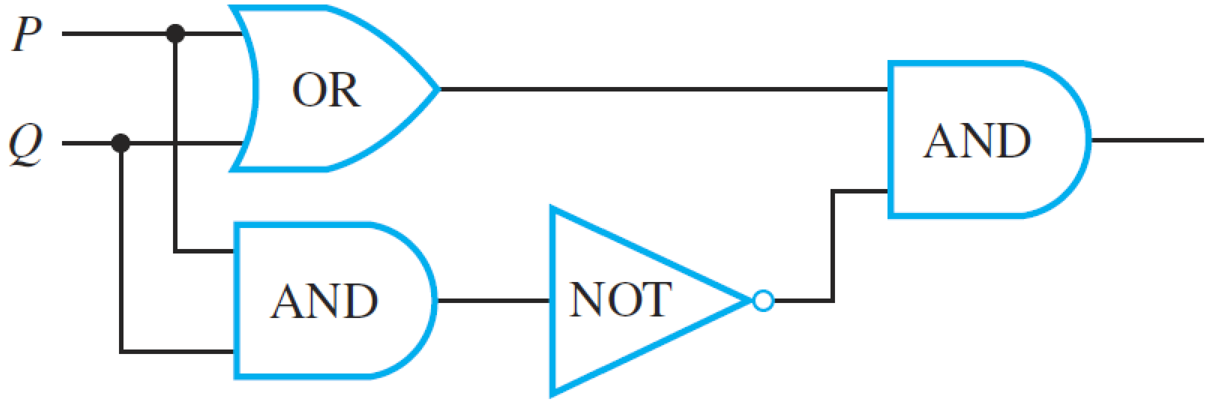
Show $\neg(p \Rightarrow q)$ and $p \land \neg q$ are logically equivalent.

**Example**

Simplify proposition $\neg(p \land q) \lor q$.

# Application: Digital Logic Circuits

Finding a Boolean expression for a Circuit




Constructing Circuits for Boolean Expressions

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

# Summary

You should be able to:

- Describe basics of propositional logic
- Combine propositions to express English sentences
- Derive truth tables of compound propositions
- Show logical equivalence using truth tables
- Show logical equivalence using known logical equivalence

# Definitions covered

- Proposition and compound proposition
- $\neg$, $\wedge$, $\vee$, $\oplus$, $\Rightarrow$, $\Leftrightarrow$ with terms (negation, conjunction, etc.)
- Contrapositive, converse, tautology, and contradiction
- Truth table
- Logical equivalence