CE141 Mathematics for Computing

Dr Nigel Newton (email: `njn`)

School of Computer Science and Electronic Engineering
University of Essex, Colchester

October 1, 2014

University of Essex

Basics of CE141

Timetabled Sessions

- One *Lecture* per fortnight, attended by all students (1 hour).
- Two or three *Exercise Classes* per week. Each student attends one class only, as indicated on their timetable (1 hour).
- One voluntary *Drop-In Help Session* per fortnight (1 hour).
- One *Progress Test* every fourth week.

Numeracy Support Module

You come from a wide variety of mathematical backgrounds, and some of you may need additional mathematical support. The University provides this in a module taught by the Skills Centre: SK080-4-AU. You are **required** to take a diagnostic test **this week** to determine whether or not you would benefit from taking this additional module. To do this follow the link "Instructions for Numeracy Diagnostic Test" on the CE141 Moodle page.

Teaching and learning methods

How to study

- The lectures and classes are of equal importance: the tests and examination may be based on material from either.
- You will learn most effectively by doing the exercises and seeking to *understand* them.
- You will not learn by simply *memorising* information.
- Add your own annotations to the printed lecture notes. You will need these notes for revision later.
- Help each other. Discuss points in pairs or in groups.
- In case of problems with the module, your first action should be to attend the Drop-In Help Session, but I will also see students individually, if this is insufficient. You can also email me.

Overview

Syllabus

Part I:	Propositional Logic
Part II:	Permutations and Combinations
Part III:	Sets
Part IV:	Probability
Part V:	Vectors and Matrices

Assessment

- There are five *Progress Tests*, one for each Part.
- The tests are in weeks 5, 9, 17, 21 and 25.
- Each test is worth 8% of the module mark.
- The remaining 60% of the assessment for the module is in the *June Examination*.

Part I

Propositional Logic

Index

Logic

What is Logic?

- Any system of *valid reasoning*.
- Modern logic is *formal*; it uses *symbols* put together according to a *syntax*, and applies *rules of inference*.
- The aim of this approach is to avoid any un-intended prejudices entering into the reasoning process.
- Logic ultimately comes down to an attempt to prove the truth or falseness of *propositions*.

What is a Proposition?

- A binary variable that can take only two values: `true` or `false`.
- Usually expressed in a natural language such as English.
- For example, "all CE141 lectures take place in LTB7" is a proposition.

Logic

Why Study Logic?

Valid reasoning is important in many fields of human endeavour, particularly in Science and Engineering.

For example, when writing a program, a Computer Scientist has to reason what his/her code will do in all likely scenarios.

Logic can also be used in Computer Science to:

- establish that a program will terminate in finite time;
- establish that two (apparently quite different) programs perform the same function;
- establish that a "machine" (hardware or virtual) is capable of being programmed to perform a usefully large set of tasks.
- ...

You will meet some of these applications in other modules.

Logic

Who else studies Logic?

Logic is also taught:

- In Philosophy, where it is concerned with formal reasoning.
- In Mathematics, where it is used for proving *theorems*.
- In Electronics, where it is used in the design of of digital circuits.

You probably won't be concerned with Philosophy, but you may encounter logic in one of these other contexts as part of your studies.

A different notation is used in Electronics (Boolean algebra with '0's and '1's). You will encounter this notation in CE161. The aims are somewhat different there.

Propositions

A proposition is a binary variable that can take only two possible values: `true` or `false`.

Are these propositions?

7 is a prime number.	Yes, this is a proposition. Value: `true`?
3 is an even number.	Yes, this is a proposition. Value: `false`?
Saturday is a weekday.	Yes, this is a proposition. Value: `false`?
Is there life on Mars?	No, this is a *question*.
Red is the nicest colour.	No, this is a matter of taste.
100 is a large number.	No, unless “large” is first made precise.

Truth values are not always so “obvious”

There is no largest prime number. Yes, this is a proposition, but its truth value is not immediately obvious. It can be proven to be `true` by the use of logic. (Euclid *circa* 300–260 BC.)

Propositions

A proposition is a binary variable that can take only two possible values: true or false.

Are these propositions?
7 is a prime number. Yes, this is a proposition. Value: true?
3 is an even number. Yes, this is a proposition. Value: false?
Saturday is a weekday. Yes, this is a proposition. Value: false?
Is there life on Mars? No, this is a question.
Red is the nicest colour. No, this is a matter of taste.
100 is a large number. No, unless “large” is first made precise.

Truth values are not always so “obvious”
There is no largest prime number. Yes, this is a proposition, but its truth value is not immediately obvious. It can be proven to be true by the use of logic. (Euclid circa 300–260 BC.)

For interest only (not part of the examinable content of CE141):
The theorem that there is no largest prime number, attributed to Euclid, can be found on the Wikipedia under ‘Euclid’s Theorem’:

`http://en.wikipedia.org/wiki/Euclid’s_theorem`

This is a classic example of something that seems intuitively incorrect, since it is easy to think that the larger a number is the more likely it is that it can be divided by some smaller number. Euclid’s Theorem shows by a proof that is not difficult to understand, that intuition can be wrong.

Propositions

"Variable" propositions

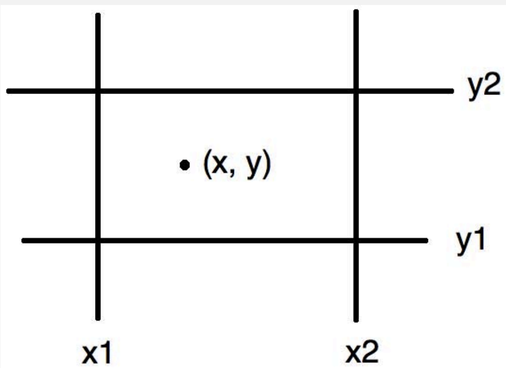
- All the propositions mentioned so far are essentially invariant (or constant): they take one and only one value.
- Propositions can also depend on *variables* or *parameters*.
- For example, the proposition "a bus is scheduled to arrive at 7.00 am", may be `true` on weekdays but `false` on Sundays; i.e. its value may depend on the day of the week.
- The simplest variables on which a proposition can depend are other propositions. Propositions exhibiting this type dependency are called *compound* propositions.

Propositional Logic

This is the simplest system of formal logic. It can be used to prove the truth or falsehood of compound propositions that depend on other propositions whose truth or falsehood is known.

Compound Propositions

In all applications of Logic, there is a need to combine *multiple* propositions to form new *compound* propositions.



In computer graphics, for example, determining whether a point (*e.g.* the position of a cursor) lies inside a rectangular box specified by two X and two Y coordinates requires four propositions to be tested:

$$x > x1, \quad x < x2,$$
$$y > y1, \quad y < y2.$$

If all four are true then the point (x, y) lies inside the box.

Logical Operators

Natural language connectives

In natural languages, such as English, we can combine statements:

- I will go to the meeting if it is held on Friday **and** it is held in London.
- If the price of gas doubles **or** the price of solar installations halves I will install solar water heating.

(Statements underlined, highlighted words *connect* them.)

Logical connectives

In Logic, we use *logical connectives* to combine logic values and propositions, much as we use arithmetic operators in ordinary algebra to combine numbers and ordinary algebraic variables. The three most common logic operators are **and**, **or** and **not**.

Conjunction Operator: AND

Mathematical symbol and properties

- Combines *two* (operand) propositions into a compound proposition.
- cf. multiplying two ordinary algebraic variables, x and y, together to obtain the new algebraic variable xy.
- AND is denoted by the 'wedge' symbol $\wedge$
- The compound proposition is `true` only when *both* operand propositions are `true`.
- If p and q are propositions,

$$p \wedge q \text{ is } \texttt{true} \quad \text{if and only if} \quad p \text{ is } \texttt{true} \quad \text{and} \quad q \text{ is } \texttt{true}.$$

- In the graphics example, if p is "$x > x1$" and q is "$x < x2$", then

$$p \wedge q \text{ is "the } x\text{-coordinate of the cursor is in range"}$$

Disjunction Operator: OR

Mathematical symbol and properties

- This also combines *two* operand propositions.
- OR is denoted by the 'vee' symbol $\vee$
- The compound proposition is `true` when either operand is `true` (and when *both* are true).
- If p and q are propositions,

$$p \vee q \text{ is } \texttt{true} \quad \text{if either} \quad p \text{ is } \texttt{true} \quad \text{or} \quad q \text{ is } \texttt{true} \quad \text{or both.}$$

- In the graphics example, if r is "$x \leq x1$" and s is "$x \geq x2$", then

$$r \vee s \text{ is "the } x\text{-coordinate of the cursor is out of range"}$$

Logical Inversion Operator: NOT

Mathematical symbol and properties

- This has only *one* operand proposition.
- It is denoted by the symbol $\neg$
- The compound proposition is `true` when the operand is `false` and `false` when the operand is `true`.
- If p is a proposition,

$$\neg p \text{ is } \begin{cases} \texttt{true} & \text{if } p \text{ is } \texttt{false} \\ \texttt{false} & \text{if } p \text{ is } \texttt{true} \end{cases}$$

- In the graphics example, if p is "$x > x1$", q is "$x < x2$", r is "$x \leq x1$" and s is "$x \geq x2$", then

$$r \text{ is } \neg p, \quad s \text{ is } \neg q \quad \text{and } r \vee s \text{ is } \neg(p \wedge q)$$

Logical Inversion Operator: NOT

Mathematical symbol and properties

- This has only *one* operand proposition.
- It is denoted by the symbol $\neg$
- The compound proposition is `true` when the operand is `false` and `false` when the operand is `true`.
- If p is a proposition,

$$\neg p \text{ is } \begin{cases} \texttt{true} & \text{if } p \text{ is } \texttt{false} \\ \texttt{false} & \text{if } p \text{ is } \texttt{true} \end{cases}$$

- In the graphics example, if p is "$x > x1$", q is "$x < x2$", r is "$x \leq x1$" and s is "$x \geq x2$", then

$$r \text{ is } \neg p, \quad s \text{ is } \neg q \quad \text{and } r \vee s \text{ is } \neg(p \wedge q)$$

Note that the 'wedge', 'vee', and 'not' symbols are special mathematical characters. The 'vee' in particular is *not* the Roman letter 'v' or 'V'. When you undertake assessed work for this module you are expected to use the correct symbols.

Combining Operators

Rules of precedence

- NOT is always taken to be of higher precedence than the other two.

$$\neg p \wedge q \quad \text{means} \quad (\neg p) \wedge q \;\; \text{not} \;\; \neg(p \wedge q)$$
$$\neg p \vee q \quad \text{means} \quad (\neg p) \vee q \;\; \text{not} \;\; \neg(p \vee q)$$

- To make things clear, we will use *parentheses* in expressions involving AND and OR. So we will not write something like: $p \vee q \wedge r \vee s$ but will instead insert parentheses like this:

$$p \vee (q \wedge r) \vee s \quad \text{or} \quad (p \vee q) \wedge (r \vee s)$$

Different compound propositions!

Truth Tables

A logical expression combining one or more operators and one or more propositions (to form a compound proposition) can be exhaustively tabulated since the number of combinations of input values is finite. A tabulation of all possible operand values and the corresponding compound proposition values is called a *truth table*.

Truth tables for the NOT, AND and OR operators:

p	$\neg p$
F	T
T	F

p	q	$p \wedge q$
F	F	F
F	T	F
T	F	F
T	T	T

p	q	$p \vee q$
F	F	F
F	T	T
T	F	T
T	T	T

(`F` and `T` here mean the same as `false` and `true`.)
You must *learn* these three truth tables and the symbols $\neg$, $\wedge$ and $\vee$.

Number of rows in a Truth Table

How many rows are there in a truth table?

This clearly depends on the number of operands in the expression represented by the table. Let's call that number N. In the following expression, $N = 3$:

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

Answer

- The general answer is 2^N (*i.e.* N copies of the number 2 multiplied together). So, in the example above, $2^3 = 8$.
- We won't discuss why this is so here, think about it!
- All Computing and Electronics students should learn the powers of 2 up to about $N = 10$. They are: 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024.

Truth table for: $f(p, q, r) := (p \wedge \neg q) \vee (q \wedge \neg r)$

We need 8 rows; three columns for p, q and r; two for $\neg q$ and $\neg r$; a column for each bracketed expression; and a column for the result.

p	q	r	$\neg q$	$\neg r$	$p \wedge \neg q$	$q \wedge \neg r$	$f(p, q, r)$
F	F	F	T	T	F	F	F
F	F	T	T	F	F	F	F
F	T	F	F	T	F	T	T
F	T	T	F	F	F	F	F
T	F	F	T	T	T	F	T
T	F	T	T	F	T	F	T
T	T	F	F	T	F	T	T
T	T	T	F	F	F	F	F

Truth table for: $f(p, q, r) := (p \wedge \neg q) \vee (q \wedge \neg r)$

We need 8 rows; three columns for p, q and r; two for $\neg q$ and $\neg r$; a column for each bracketed expression; and a column for the result.

p	q	r	$\neg q$	$\neg r$	$p \wedge \neg q$	$q \wedge \neg r$	$f(p, q, r)$
F	F	F	T	T	F	F	F
F	F	T	T	F	F	F	F
F	T	F	F	T	F	T	T
F	T	T	F	F	F	F	F
T	F	F	T	T	T	F	T
T	F	T	T	F	T	F	T
T	T	F	F	T	F	T	T
T	T	T	F	F	F	F	F

You should write out the 2^N rows of a truth table in a natural binary sequence, starting with either `FF...F` or `TT...T`. The natural binary sequence used for the 3-variable truth table in the slides is ascending: $000, 001, 010, 011, 100, 101, 110, 111$, with `F` corresponding to 0, and `T` corresponding to 1. The idea of natural binary will be covered in more detail in CE161. You can also find material about it on the Internet, for example on Wikipedia:

`http://en.wikipedia.org/wiki/Binary_numeral_system`.

Experience shows that, if students don't construct truth tables with natural binary sequences they omit or duplicate rows, particularly when the number of rows is large.

It also makes it difficult for others to read, check and understand the truth table.

De Morgan's Laws

These two laws show how to express:

- AND in terms of OR and NOT
- OR in terms of AND and NOT

The usual form of the laws is:

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

The validity of the laws can be verified by truth tables; for example

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

De Morgan's Laws
These two laws show how to express:
- AND in terms of OR and NOT
- OR in terms of AND and NOT
The usual form of the laws is:
$\neg(p \vee q) \equiv \neg p \wedge \neg q$
$\neg(p \wedge q) \equiv \neg p \vee \neg q$
The validity of the laws can be verified by truth tables; for example
p	q	$p \vee q$	$\neg(p \vee q)$	$\neg p$	$\neg q$	$\neg p \wedge \neg q$
F	F	F	T	T	T	T
F	T	T	F	T	F	F
T	F	T	F	F	T	F
T	T	T	F	F	F	F

Be sure you understand why the truth table in the slide proves the validity of the (first) law: the third column (corresponding to the left-hand side, or LHS, of the law) has the same entries as the last column (corresponding to the right-hand side, or RHS). This proves the law to be correct, provided the truth table *shows all the possibilities and all the entries match*. Of course, a truth table can also be used to show the contrary: if you are given a putative identity that you believe to be INCORRECT, for example "$p \wedge q \equiv \neg(p \vee q)$" you can prove it to be incorrect by drawing up a truth table and showing that AT LEAST ONE entry in the column corresponding to the LHS is different to the entry in the column corresponding to the RHS. In fact you don't have to draw up the whole table: you can stop as soon as you find one combination of p and q for which the LHS and the RHS give different results.

Algebraic Rules

Truth Tables

- We saw above how to use truth tables to verify a logical identity such as $\neg(p \vee q) \equiv \neg p \wedge \neg q$ (one of De Morgan's Laws).
- This can be done by showing that the left-hand and right-hand sides of the identity have the same truth table (*i.e.* both sides have the same `true`/`false` values for every combination of values of the propositions p and q).

Algebra

- An alternative method to verify equalities is to use *algebra*, that is a sequence of steps based on *known* algebraic identities, to convert one side of the putative identity into the other, and thus show that they are equivalent.
- To do this we need a list of standard algebraic identities ...

A selection of Algebraic Identities

Negative Identity

$$\neg\neg p \equiv p$$

Commutative Identities

$$p \vee q \equiv q \vee p, \qquad p \wedge q \equiv q \wedge p$$

Associative Identities

$$p \vee (q \vee r) \equiv (p \vee q) \vee r, \qquad p \wedge (q \wedge r) \equiv (p \wedge q) \wedge r$$

Distributive Identities

$$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r), \quad p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$$

cf. ordinary algebra, in which

$$\begin{array}{rrcl} & x(y+x) & \equiv & xy + xz \\ \text{but} & x + (yz) & \neq & (x+y)(x+z). \end{array}$$

More Algebraic Identities

Excluded Middle

$$p \lor \neg p \equiv \mathtt{T}$$

Contradiction

$$p \land \neg p \equiv \mathtt{F}$$

Simplifications: OR

$$p \lor p \equiv p, \qquad p \lor \mathtt{T} \equiv \mathtt{T}, \qquad p \lor \mathtt{F} \equiv p, \qquad p \lor (p \land q) \equiv p$$

Simplifications: AND

$$p \land p \equiv p, \qquad p \land \mathtt{T} \equiv p, \qquad p \land \mathtt{F} \equiv \mathtt{F}, \qquad p \land (p \lor q) \equiv p$$

More Algebraic Identities
Excluded Middle
$p \lor \neg p \equiv \mathtt{T}$
Contradiction
$p \land \neg p \equiv \mathtt{F}$
Simplifications: OR
$p \lor p \equiv p, \quad p \lor \mathtt{T} \equiv \mathtt{T}, \quad p \lor \mathtt{F} \equiv p, \quad p \lor (p \land q) \equiv p$
Simplifications: AND
$p \land p \equiv p, \quad p \land \mathtt{T} \equiv p, \quad p \land \mathtt{F} \equiv \mathtt{F}, \quad p \land (p \lor q) \equiv p$

- The list of algebraic identities was taken from a slightly longer list in John E. Munro, *Discrete Mathematics for Computing*, Chapman and Hall, 1992, on page 80.
- The omitted identities include the conditional and biconditional which are covered in later slides. De Morgan's Laws are also omitted as they are stated on an earlier slide.
- Some of the identities have rather elaborate names for such simple concepts. For example, *commutativity* simply means that the order of the operands is not important (as in normal addition and multiplication); *associativity* simply means that the order in which you compute an expression involving two or more identical operators (*e.g.* $p \land q \land r$) does not matter. (So you can omit parentheses without ambiguity.)
- Unlike in ordinary algebra, the distributive property works both ways: AND distributes over OR, and OR distributes over AND.

An example of Algebraic Manipulation

The problem

Show that $p \vee (p \wedge q) \equiv p$ and $p \wedge (p \vee q) \equiv p$ using *other* algebraic identities from the list above.

Proof of first identity

Using $p \wedge \mathtt{T} \equiv p$, replace the first p on the left-hand side to get:

$$p \vee (p \wedge q) \equiv (p \wedge \mathtt{T}) \vee (p \wedge q) \tag{1}$$

Use the first Distributive Law, then two of the Simplification Laws:

$$\begin{aligned} (p \wedge \mathtt{T}) \vee (p \wedge q) &\equiv p \wedge (\mathtt{T} \vee q) \\ &\equiv p \wedge \mathtt{T} \\ &\equiv p. \end{aligned}$$

An example of Algebraic Manipulation
The problem
Show that $p \vee (p \wedge q) \equiv p$ and $p \wedge (p \vee q) \equiv p$ using *other* algebraic identities from the list above.
Proof of first identity
Using $p \wedge \mathtt{T} \equiv p$, replace the first p on the left-hand side to get:
$$p \vee (p \wedge q) \equiv (p \wedge \mathtt{T}) \vee (p \wedge q) \tag{1}$$
Use the first Distributive Law, then two of the Simplification Laws:
$$\begin{aligned} (p \wedge \mathtt{T}) \vee (p \wedge q) &\equiv p \wedge (\mathtt{T} \vee q) \\ &\equiv p \wedge \mathtt{T} \\ &\equiv p. \end{aligned}$$

If your skills at algebra are rusty or you never learned algebra, there are several points in the previous slide which you could find difficult.

- The first is probably where p is replaced by $p \wedge \mathtt{T}$. This may appear crazy because it seems to complicate the expression. It uses one of the Simplification Laws *in the reverse sense*.
- Secondly, why don't we replace the other copy of p also? We could, but it is *not helpful* in what follows. Working out what to do in a given situation comes with experience. The moral is: if you can't immediately see what to do, try out a few of the identites to see where they lead.
- The second step relies on matching expression (1) with the Distributive Law. Once again, this is the reverse of the Distributive Law as stated in the list.

An example continued: the Dual Expression

The problem again

Show that $p \vee (p \wedge q) \equiv p$ and $p \wedge (p \vee q) \equiv p$ using *other* algebraic identities from the list.

Duality

The two equivalences above are **duals** of each other. Notice that $\vee$ in one equivalence is replaced by $\wedge$ in the other.

Exercise: prove the second equivalence

Show that the second equivalence is true, using the same ideas as on the previous slide, but replacing $\vee$ with $\wedge$ and *vice versa* plus any other changes that you find necessary guided by the idea of duality.

Conditional

There are two more common operators in propositional logic, the *conditional* and the *biconditional*.

The *conditional* proposition "if p then q" is denoted $p \longrightarrow q$. As an example of its use in reasoning, we could substitute for p the proposition "I live in Colchester" and for q the proposition "I live in England". The conditional then expresses the compound proposition "If I live in Colchester, then I live in England".

Truth table and an equivalent

p	q	$p \longrightarrow q$
F	F	T
F	T	T
T	F	F
T	T	T

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

Biconditional

The *biconditional* proposition "p if and only if q" is denoted by $p \longleftrightarrow q$. It is equivalent to the compound proposition $(p \longrightarrow q) \wedge (q \longrightarrow p)$.

Truth table

p	q	$p \longleftrightarrow q$
F	F	T
F	T	F
T	F	F
T	T	T

Note that the biconditional is `true` only when p and q take the same value.

Yet more Algebraic Identities

Two identities were stated on the two previous slides, but we didn't name them. Here they are again, just to complete our list:

Law of implication

$$p \longrightarrow q \equiv \neg p \vee q$$

Law of equivalence

$$p \longleftrightarrow q \equiv (p \longrightarrow q) \wedge (q \longrightarrow p)$$

Tautology and Contradiction

Tautology

- A compound proposition that is identically `true` (*i.e.* it takes the value `true` no matter what values its constituent propositions take) is called a **tautology**.
- Equivalently, a compound proposition whose truth table final column is `true` in every row is a tautology.

Contradiction

- A compound proposition that is identically `false` (*i.e.* it takes the value `false` no matter what values its constituent propositions take) is called a **contradiction**.
- Equivalently, a compound proposition whose truth table final column is `false` in every row is a contradiction.

An example

A proposition, expressed in English

Either Alice is English, or Bernard is German;
or Alice is not English and Bernard is not German

Is this a tautology? Can you tell directly?

Here it is expressed more formally, then in symbols

(Alice is English OR Bernard is German) OR
(NOT Alice is English AND NOT Bernard is German)

Let p be the proposition "Alice is English" and q be the proposition "Bernard is German". Then we can write the compound proposition as:

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

and we can use a truth table (exercise!) or algebra (next slide) on it.

Showing that $(p \lor q) \lor (\neg p \land \neg q)$ is a Tautology

Here is a sequence of steps to show that the proposition is `true` for all values of p and q, and is therefore a tautology:

$$\begin{array}{rcll} (p \lor q) \lor (\neg p \land \neg q) & \equiv & (p \lor q) \lor \neg(p \lor q) & \text{by De Morgan} \\ & \equiv & r \lor \neg r & \text{where } r := (p \lor q) \\ & \equiv & \mathtt{T} & \text{excluded middle} \end{array}$$

Here is a different and longer sequence of steps:

$$\begin{array}{rcll} (p \lor q) \lor (\neg p \land \neg q) & \equiv & s \lor (\neg p \land \neg q) & \text{where } s := (p \lor q) \\ & \equiv & (s \lor \neg p) \land (s \lor \neg q) & \text{distributive rule} \\ & \equiv & (p \lor q \lor \neg p) \land (p \lor q \lor \neg q) & \\ & \equiv & (q \lor \mathtt{T}) \land (p \lor \mathtt{T}) & \text{excluded middle} \\ & \equiv & \mathtt{T} \land \mathtt{T} \equiv \mathtt{T} & \text{simplification-OR} \end{array}$$

The end result is `true`, regardless of whether Alice is actually English or Bernard is actually German.

Further Reading

There is no recommended printed book for this part of CE141 because the material is usually covered in books on Discrete Mathematics, and most of the content of such books is not needed. However, Wikibooks has an on-line book on *Discrete Mathematics* with a section on logic:

`http://en.wikibooks.org/wiki/Discrete_Mathematics/Logic`

(This Wikibook uses a double arrow $\Longrightarrow$ for the conditional rather than the single arrow used in these notes and many books.)

You should try some exercises from this book, either on your own, or working with another student. However, not all of the exercises or material are relevant. (The book covers more than CE141.) Stop after Logic Exercises 4 on Page 2.

If you are interested in exploring the wider background to logic, you could start with Wikipedia. Some pages may be incomplete and/or badly written, but on the whole, the material is useful, and particularly so when you want to explore a topic to find out what it is about. Often the articles will contain references to other sources including books that you can locate easily using Wikipedia articles. Further aspects of logic are discussed in the following notes.

Taking Things Further (not examinable)

This section of CE141 has introduced propositional logic. At the level of this module the subject can seem a bit "obvious" or "pointless". Remember, however, that the aim of the early logicians was to create a formal system of reasoning that was immune from pre-conceived notions or prejudices. The first step in this process was the replacement of natural language propositions (about which we may have culturally inherited opinions) with symbols (about which we should be neutral).

The modern subject of propositional logic starts by defining precisely what symbols can be used. As we have seen, they are:

- letters (representing elementary propositions): p, q, r, etc.
- the special symbols: $\wedge$, $\vee$, $\neg$, (, and).

These can be assembled into *strings*, for example "$p\wedge) \vee \neg$". Only certain strings are valid logical expressions. (The example above is not!)

Like a programming language (or even a natural language such as English), what constitutes a valid string is specified by a *syntax*. The syntax of propositional logic is fairly obvious: $\wedge$ and $\vee$ must always be flanked by propositions (either elementary or compound), $\neg$ must always be followed by a proposition (either elementary or compound), brackets must balance, etc. Valid logical expressions themselves represent compound propositions.

Theorems of propositional logic are valid expressions that are tautologies. Such expressions represent compound propositions that are `true` regardless of the truth values of their constituent elementary propositions and are, therefore, fundamental facts about the formal system. For example, De Morgan's first law is a theorem of propositional logic since it can be expressed it terms of the tautolgy $\neg(p \vee q) \longleftrightarrow \neg p \wedge \neg q$. Note that "$\equiv$" was not included in the list of symbols above, and so $\neg(p \vee p) \equiv \neg p \wedge \neg q$ is not a valid expression in the above sense. (It is shorthand for saying that the valid expression $\neg(p \vee q) \longleftrightarrow \neg p \wedge \neg q$ is a tautology.)

We have seen how to manipulate algebraic identities to obtain other algebraic identities in this section of CE141. This process is actually one of proving new theorems from pre-existing theorems. The fact that *all* theorems (valid expressions that are tautologies—a property that can always be tested by a truth table) can be proved in this way starting from a few "axioms" (accepted theorems, which can again be checked by truth tables) is called the *completeness* of propositional logic. "Higher order" systems of logic such as *predicate logic* do not all share this property.

Predicate logic introduces *quantifiers* such as "every" in the proposition "every integer can be expressed as the sum of two perfect squares". The "imperfections" of predicate logic open a chink of doubt, even in proofs as "transparent" as Euclid's maximum prime theorem.