Introduction to Logic

Christopher M. Bourke

Computer Science & Engineering 235
Introduction to Discrete Mathematics

cbourke@cse.unl.edu

Introduction I

Propositional calculus (or logic) is the study of the logical relationship between objects called propositions and forms the basis of all mathematical reasoning.

Definition

A *proposition* is a statement that is either *true* or *false*, but not both (we usually denote a proposition by letters; $p, q, r, s, \ldots$).

Introduction II

Definition

The value of a proposition is called its *truth value*; denoted by T or 1 if it is true and F or 0 if it is false.

Opinions, interrogative and imperative sentences are not propositions.

Examples I

Example (Propositions)

- $2 + 2 = 4$
- The derivative of $\sin x$ is $\cos x$.
- 6 has 2 factors

Example (Not Propositions)

- `C++` is the best language.
- When is the pretest?
- Do your homework.

Examples II

Example (Propositions?)

- $2 + 2 = 5$
- Every integer is divisible by 12.
- Microsoft is an excellent company.

Logical Connectives

Connectives are used to create a *compound* proposition from two or more other propositions.

- Negation (denoted $\neg$ or !)
- And (denoted $\wedge$) or Logical Conjunction
- Or (denoted $\vee$) or Logical Disjunction
- Exclusive Or (XOR, denoted $\oplus$)
- Implication (denoted $\rightarrow$)
- Biconditional; "if and only if" (denoted $\leftrightarrow$)

Negation

A proposition can be negated. This is also a proposition. We usually denote the negation of a proposition p by $\neg p$.

Example (Negated Propositions)

- Today is *not* Monday.
- *It is not the case that* the derivative of $\sin x$ is $\cos x$.
- There exists an even number that has less than two factors.

Marge: Homer, you're not licking toads, are you?
Homer: I'm not NOT licking toads.

Logical And

The logical connective $\textsc{And}$ is true only if *both* of the propositions are true. It is also referred to as a *conjunction*.

Example (Logical Connective: $\textsc{And}$)

- It is raining and it is warm.
- $(2+3=5) \wedge (\sqrt{2} < 2)$
- Schrödinger's cat is dead and Schrödinger's cat is not dead.

Logical Or

The logical disjunction (or logical $\textsc{Or}$) is true if one or both of the propositions are true.

Example (Logical Connective: $\textsc{Or}$)

- It is raining or it is the second day of lecture.
- $(2+2=5) \vee (\sqrt{2} < 2)$
- You may have cake or ice cream.[1]

[1] Can I have both?

Logic Gate Notation

- Alternative notations in other fields
- Negation (overline): $\overline{p}$
- Or (gate): $p+q$
- And (gate): $p \cdot q$ or pq
- Motivated by natural algebraic operations

Exclusive Or

The exclusive or of two propositions is true when exactly *one* of its propositions is true and the other one is false.

Example (Logical Connective: Exclusive Or)

- The circuit is either is on or off.
- Let $ab < 0$, then either $a < 0$ or $b < 0$ but both cannot be the case.
- You may have cake or ice cream, but not both.

May be more convenient to use the following:

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

Implications I

Definition

Let p and q be propositions. The implication

$$p \to q$$

is the proposition that is false when p is true and q is false and true otherwise.

Here, p is called the "hypothesis" (or "antecedent" or "premise") and q is called the "conclusion" or "consequence".

Implications II

The implication $p \to q$ can be equivalently read as

- if p then q
- p implies q
- p only if q
- q if p
- p is a sufficient condition for q
- q is a necessary condition for p
- q follows from p

Examples

Example

- If $2 + 2 = 5$ then all unicorns are pink.
- If a matrix A has determinant 1 then A is invertible.
- If you do your homework, you may have cake or ice cream.[1]
- If Rizzo plays, the Cubs will win. Rizzo didn't play, did we win?
- If we go east, we will get to Memorial Stadium. We got to Memorial Stadium; did we necessarily go east?

[1] Again, am I allowed both?

Material Conditional versus Entailment I

- May seem odd that unrelated propositions can be used in implications
- May seem odd that if premise doesn't hold, conclusion is irrelevant
- Natural languages (English) are *idiomatic* – we speak in metaphors with meanings understood by context and experience, "piece of cake" vs "asa meshi mae"
- English use of implication connotes a *physical causation*
  "If it is sunny, then we will go to the beach"
- Philisophical Implication (Entailment, Connexive logic) involves some relation or relevance
  "If A implies B, then it cannot imply not-B"

Material Conditional versus Entailment II

- Connexive logic (antiquity) denies the possibility of "not-B implies B":
  1. $(A \to B)$
  2. $(\neg A \to B)$
  3. $(\neg B \to \neg A)$ (from 1, contrapositive)
  4. $(\neg B \to B)$ (from 3, 2, transitivity)
- Example: does the following make sense?
  "If it is not raining, then it is raining."
- Boolean (Material Conditional or Material Implication):
  - Since 19th Century
  - Does not involve such reasoning: no relevance is required
  - More general, mathematically consistent
  - Simply "If the conclusion holds, then the premise must"

Exercise

Which of the following implications is true?

- If -1 is a positive number, then $2 + 2 = 5$.
  true: the hypothesis is obviously false, thus no matter what the conclusion, the implication holds.
- If -1 is a positive number, then $2 + 2 = 4$.
  true: for the same reason as above
- If $\sin x = 0$ then $x = 0$.
  false: x can be any multiple of π; i.e. if we let $x = 2\pi$ then clearly $\sin x = 0$, but $x \neq 0$. The implication "if $\sin x = 0$ then $x = k\pi$ for some integer k" is true.

Biconditional

Definition

The *biconditional*,

$$p \leftrightarrow q$$

is the proposition that is true when p and q have the same truth values and is false otherwise.

It can be equivalently read as

- p if and only if q
- p is necessary and sufficient for q
- if p then q, and conversely
- p iff q

Note, that it is equivalent to

$$(p \to q) \wedge (q \to p)$$

Examples

Example

- $x > 0$ if and only if x^2 is positive.
- A matrix A has a non-zero determinant if and only if A is invertible.
- You may have pudding if and only if you eat your meat.[1]

[1] How can you have any pudding if you don't eat your meat?

Exercise

Which of the following biconditionals is true?

- $x^2 + y^2 = 0$ if and only if $x = 0$ and $y = 0$
  true: both implications hold.
- $2 + 2 = 4$ if and only if $\sqrt{2} < 2$
  true: for the same reason above.
- $x^2 \geq 0$ if and only if $x \geq 0$.
  false: The converse holds. That is, "if $x \geq 0$ then $x^2 \geq 0$". However, the implication is false; consider $x = -1$. Then the hypothesis is true, $1^2 \geq 0$ but the conclusion fails.

Converse, Contrapositive, Inverse

The proposition, $q \to p$ is called the converse of the proposition $p \to q$.

The contrapositive of the proposition $p \to q$ is $\neg q \to \neg p$

The inverse of the proposition $p \to q$ is $\neg p \to \neg q$

Truth Tables I

Truth Tables are used to show the relationship between the truth values of individual propositions and the compound propositions based on them.

p	q	$p \wedge q$	$p \vee q$	$\oplus$	$p \to q$	$p \leftrightarrow q$
0	0	0	0	0	1	1
0	1	0	1	1	1	0
1	0	0	1	1	0	0
1	1	1	1	0	1	1

Table : Truth Table for Logical Conjunction, Disjunction, Exclusive Or, and Implication

Constructing Truth Tables

Construct the Truth Table for the following compound proposition.

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

p	q	$p \wedge q$	$\neg q$	$((p \wedge q) \vee \neg q)$
0	0	0	1	1
0	1	0	0	0
1	0	0	1	1
1	1	1	0	1

Precedence of Logical Operators

Just as in arithmetic, an ordering must be imposed on the use of logical operators in compound propositions.

Of course, parentheses can be used to make operators disambiguous:

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

But to avoid using unnecessary parentheses, we define the following precedences:

1. $(\neg)$ Negation
2. $(\wedge)$ Conjunction
3. $(\vee)$ Disjunction
4. $(\to)$ Implication
5. $(\leftrightarrow)$ Biconditional

Bitwise Operations

Logical connectives can be applied to bit strings (of equal length). To do this, we simply apply the connective rules to each bit of the string:

Example

0110	1010	1101	
0101	0010	1111	
0111	1010	1111	bitwise OR
0100	0010	1101	bitwise AND
0011	1000	0010	bitwise XOR

Logic in Programming

One needs logic in programming if for no other reason than to avoid ending up as one of the "Daily WTFs" (`http://thedailywtf.com`).

Example:
`http://thedailywtf.com/forums/39325/ShowPost.aspx`

Another:
`http://thedailywtf.com/forums/31835/ShowPost.aspx`

Logic in Programming I

Example: Subnet Network Addressing

In the IP (Internet Protocol) scheme, network administrators are allowed to divide large network IP ranges into smaller, more efficient subnets. However, when communications come in from outside the network, routers need to know *which* subnet to send packets to. To find the subnet number, the router uses a *subnet mask*. The logical AND operator is performed on the IP address and the subnet mask to recover the subnet number.

Logic in Programming II

Example: Subnet Network Addressing

Example

Say the subnet mask is

$$255.255.255.128 = 11111111.11111111.11111111.10000000$$

and the IP address is

$$150.100.12.176 = 10010110.01100100.00001100.10110000$$

Then the subnet number is

11111111.11111111.11111111.10000000	
10010110.01100100.00001100.10110000	
10010110.01100100.00001100.10000000	= 150.100.12.128

Logic in Programming

Programming Example I

Say you need to define a conditional statement as follows: "Increment x if all of the following conditions hold: $x > 0$, $x < 10$ and $x = 10$."

You may try:

```
if(0<x<10 OR x=10) x++;
```

But is not valid in C++ or Java. How can you modify this statement by using a logical equivalence?

Answer:

```
if(x>0 AND x<=10) x++;
```

Logic In Programming

Programming Example II

Say we have the following loop:

```
while((i<size AND A[i]>10) OR
      (i<size  AND A[i]<0) OR
      NOT (A[i]!= 0 AND NOT (A[i]>= 10)))
```

Is this good code? Keep in mind:

- Readability.
- Extraneous code is inefficient and poor style.
- Complicated code is more prone to errors and difficult to debug.

Solution?

Propositional Equivalences I
Introduction

Definition

Two propositions are *equivalent* and we write

$$p \equiv q$$

if, for all truth values of p, q, they have the same truth value.

Informally, two propositions are equivalent if they have the same truth tables. Examples:

Propositional Equivalences II
Introduction

- Implication Law

$$p \to q \equiv \neg p \vee q$$

- Equivalence Law (one of many)

$$p \leftrightarrow q \equiv (p \to q) \wedge (q \to q)$$

- Equivalence Law

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

Propositional Equivalences III
Introduction

- Associative Laws

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

- Commutative Laws

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

- Distributive Laws

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

Propositional Equivalences IV
Introduction

Lemma (De Morgan's Law)

A negation can be distributed over compound propositions according to the following rules.

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

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

These are known as De Morgan's Laws

Terminology
Tautologies, Contradictions, Contingencies

Definition

A compound proposition that is always true, no matter what the truth values of the propositions that occur in it is called a *tautology*. A compound proposition that is always false is called a *contradiction*. Finally, a proposition that is neither a tautology nor a contradiction is called a *contingency*.

Example

A simple tautology is

$$p \vee \neg p$$

and a simple contradiction is

$$p \wedge \neg p$$

Logical Equivalences
Definition

Definition

Propositions p and q are *logically equivalent* if $p \leftrightarrow q$ is a tautology. We use the notation $p \equiv q$ ("p is equivalent to q"). Alternatively, $p \iff q$ is used $p \Leftrightarrow q$.

Example

Are and $p \to q$ and $\neg p \vee q$ logically equivalent?

To find out, we construct the truth tables for each:

p	q	$p \to q$	$\neg p$	$\neg p \vee q$
0	0	1	1	1
0	1	1	1	1
1	0	0	0	0
1	1	1	0	1

The two columns in the truth table are identical, thus we conclude that

$$p \to q \equiv \neg p \vee q$$

Another Example

(Exercise 23 from Rosen): Show that

$$(p \to r) \vee (q \to r) \equiv (p \wedge q) \to r$$

p	q	r	$p \to r$	$(q \to r)$	$(p \to r) \vee (q \to r)$
0	0	0	1	1	1
0	0	1	1	1	1
0	1	0	1	0	1
0	1	1	1	1	1
1	0	0	0	1	1
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	1	1	1

Another Example
Continued

Now let's do it for $(p \wedge q) \to r$:

p	q	r	$p \wedge q$	$(p \wedge q) \to r$
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	1
1	0	1	0	1
1	1	0	1	0
1	1	1	1	1

The truth values are identical so we conclude that the logical equivalence holds.

Logical Equivalences
Cheat Sheet

Tables of logical equivalences available in Rosen

Cheat sheet available on Blackboard

Using Logical Equivalences
Example 1

- Logical equivalences can be established using truth tables
- May also be established using *other* logical equivalences

Example: Show that $(p \wedge q) \to q$ is a tautology

$$\begin{array}{llll}
((p \wedge q) \to q) & \iff & \neg(p \wedge q) \vee q & \text{Implication Law} \\
 & \iff & (\neg p \vee \neg q) \vee q & \text{De Morgan's Law (1st)} \\
 & \iff & \neg p \vee (\neg q \vee q) & \text{Associative Law} \\
 & \iff & \neg p \vee 1 & \text{Negation Law} \\
 & \iff & 1 & \text{Domination Law}
\end{array}$$

Using Logical Equivalences
Example 2

Show that

$$\neg(p \leftrightarrow q) \iff (p \leftrightarrow \neg q)$$

Sometimes it helps to start out with the second proposition. $(p \leftrightarrow \neg q)$

$$\begin{array}{lll}
\iff & (p \to \neg q) \wedge (\neg q \to p) & \text{Equivalence Law} \\
\iff & (\neg p \vee \neg q) \wedge (q \vee p) & \text{Implication Law} \\
\iff & [(\neg p \vee \neg q) \wedge q] \vee [(\neg p \vee \neg q) \wedge p] & \text{Distribution} \\
\iff & [(\neg p \wedge q) \vee (\neg q \wedge q)] \vee [(\neg p \wedge p) \vee (\neg q \wedge p)] & \text{Distribution} \\
\iff & (\neg p \wedge q) \vee (\neg q \wedge p) & \text{Simplification} \\
\iff & \neg [(p \vee \neg q) \wedge (q \vee \neg p)] & \text{De Morgan} \\
\iff & \neg [(q \to p) \wedge (p \to q)] & \text{Implication Law} \\
\iff & \neg(p \leftrightarrow q) & \text{Implication Law}
\end{array}$$

Using Logical Equivalences

Example 3

Show that

$$\neg(q \to p) \lor (p \land q) \iff q$$

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

$\iff$	$(\neg(\neg q \lor p)) \lor (p \land q)$	Implication Law
$\iff$	$(q \land \neg p) \lor (p \land q)$	De Morgan's & Double Negation
$\iff$	$(q \land \neg p) \lor (q \land p)$	Commutative Law
$\iff$	$q \land (\neg p \lor p)$	Distributive Law
$\iff$	$q \land 1$	Cancelation Law
$\iff$	q	Identity Law

This is a good example of using a law "in reverse": *factoring* out a common proposition

Logic In Programming

Programming Example II Revisited

Recall the loop:

```
while((i<size AND A[i]>10) OR
      (i<size AND A[i]<0) OR
      NOT (A[i]!= 0 AND NOT (A[i]>= 10)))
```

Now, using logical equivalences, simplify it.

Logic In Programming

Programming Example II Revisited

Answer: Use De Morgan's Law and Distributivity.

```
while((i<size) AND
      (A[i]>10 OR A[i]<0) OR
      (A[i]==0 OR A[i]>=10))
```

We should also add parentheses to make the conjunction's precedence explicit and we can simplify the final condition:

```
while( ( (i<size) AND
         (A[i]>10 OR A[i]<0) ) OR
       (A[i]>=10))
```

Programming Pitfall Note

In C, C++ and Java, applying the commutative law is not such a good idea. These languages (compiler dependent) sometimes use "short-circuiting" for efficiency (at the machine level). For example, consider accessing an integer array A of size n.

```
if(i<n && A[i]==0) i++;
```

is not equivalent to

```
if(A[i]==0 && i<n) i++;
```

In fact, you will probably get a segmentation fault. Why?