Proofs

What is a Proof?

A proof, as used by programmers and applications specialists, is a means for verifying the logical validity of a series of conditional statements. This is particularly useful when attempting to identify and correct 'dead code[1]'. It can also be used to verify 'short-cuts'. By 'short-cut', I mean the recoding of a program so that it behaves the same way, but with fewer lines of code, and thus, fewer opportunities for error.

Proofs are valuable in that you may be able to simplify an expression, thus simplifying program code, or significantly reducing database search times and resources. In the case of programming, it may be necessary to document that a desired result or operation will never be executed.

These sections will examine some of the components that go into establishing proofs and the mathematical tools to critically examine and modify them.

Tautology

A tautology is a statement which is always true. Let's consider the truth table below:

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

Notice that the result of the expression is true regardless of the truth value of the arguments.

Contradiction

A contradiction is a statement which is always false. Consider the truth table below:

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

Notice that the result of the expression is false regardless of the truth value of the arguments.

[1] 'Dead code' is program code which, through lapses in logic, will never be executed.

Contingency

A contingency is a statement which is not a tautology, nor a contradiction. Thus, it isn't possible to know in advance the truth value of a particular statement. Most propositions fall into this category.

Useful Equivalences

Through time and experience programmers and mathematicians have identified many useful expressions that are logically equivalent. In most cases a name has been attached to them, but they may be referenced by different names. Below are listed some of the more common.

Complement Laws	$\neg T \equiv F$ $\neg F \equiv T$
Excluded Middle Law	$p \vee \neg p \equiv T$
Contradiction Law	$p \wedge \neg p \equiv F$
Identity Laws	$p \vee F \equiv p$ $p \wedge T \equiv p$
Domination Laws	$p \vee T \equiv T$ $p \wedge F \equiv F$
Idempotent Laws	$p \vee p \equiv p$ $p \wedge p \equiv p$
Double Negation Law	$\neg \neg p \equiv p$
Commutative Laws	$p \vee q \equiv q \vee p$ $p \wedge q \equiv q \wedge p$
Associative Laws	$(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$ $(p \vee q) \vee r \equiv p \vee (q \vee r)$
Distributive Laws	$(p \vee q) \wedge (p \vee r) \equiv p \vee (q \wedge r)$ $(p \wedge q) \vee (p \wedge r) \equiv p \wedge (q \vee r)$
DeMorgan's Laws	$\neg p \vee \neg q \equiv \neg(p \wedge q)$ $\neg p \wedge \neg q \equiv \neg(p \vee q)$

Proof Overview

The basic structure of a proof is that it is composed of one or more inferences or arguments, and a conclusion. A proof is called valid if, when each or all of the inferences are true, the conclusion is true. Thus, all of the arguments must be true, and the conclusion is also true for the proof to be valid. A proof is invalid only if we come across a case where all of the arguments are true, but the conclusion is false.

A way to state this in a more mathematical form would look something like this:

$$(p_1) \wedge (p_2) \wedge (p_3) \ldots (p_n) \therefore q$$

Another way to display a proof is shown here:

p_1
p_2
p_3
...
p_n
$\therefore q$

For this reason, we can only say that a proof is *valid*, or *invalid*. In the case above, if proposition p_1 is true and p_2 is true and p_3 is true and p_n is true, and q is true, then the proof is valid.

The utility of a proof comes later when we validate it. Thus, if we know that a proof is valid, then if p_1 is true, p_2 is true, p_3 is true and p_n is true, then we can be sure that q is true.

Direct Proof

In *direct proof* the purpose is to exhaustively search a logical expression for evidence of false-ness. The most complete way to do this is to create a truth table.

In the case of direct proof we create a truth table with all of the inferences or arguments, and the conclusion.

Let's take a look at a simple proof.

p
$p \wedge q$

$\therefore q$

We have two arguments or inferences we lay out in the truth table: (Line numbers have been added.)

	$\boldsymbol{p}$	q	$\boldsymbol{p \wedge q}$	q
1	T	T	T	T
2	T	F	F	F
3	F	T	F	T
4	F	F	F	F

In line 1, we see that both arguments, p and $p \wedge q$ are true. The conclusion, q, is also true. We can mark that line with a check to identity that the proof is valid so far.

	p	*q*	***p*** $\wedge$ ***q***	*q*	
1	T	T	T	T	√
2	T	F	F	F	
3	F	T	F	T	
4	F	F	F	F	

In line 2, p is true, but $p \wedge q$ is false. This doesn't mean that the proof is invalid. It just means that we ignore the results of that line. We mark the line with a dash.

	p	*q*	***p*** $\wedge$ ***q***	*q*	
1	T	T	T	T	√
2	T	F	F	F	-
3	F	T	F	T	
4	F	F	F	F	

In line 3, p is false, and $p \wedge q$ is false. Like the previous line, we mark it with a dash.

	p	*q*	***p*** $\wedge$ ***q***	*q*	
1	T	T	T	T	√
2	T	F	F	F	-
3	F	T	F	T	-
4	F	F	F	F	

In line 4, both p and $p \wedge q$ are false. So, again, we mark the line with a dash.

	p	*q*	***p*** $\wedge$ ***q***	*q*	
1	T	T	T	T	√
2	T	F	F	F	-
3	F	T	F	T	-
4	F	F	F	F	-

Because there were no contradicting conditions, this is a *valid* proof. Remember, we are attempting to prove that if p is true, and if $p \wedge q$ is true, then q will be true.

Let's take a look at another proof.

p
$p \wedge q$

$\therefore \neg q$

	$\boldsymbol{p}$	q	$\boldsymbol{p \wedge q}$	$\neg q$
1	T	T	T	F
2	T	F	F	T
3	F	T	F	F
4	F	F	F	T

In line 1, we can see that p is true, and $p \wedge q$ is true, but the conclusion, $\neg q$, is false. We can mark this with an 'x'.

	$\boldsymbol{p}$	q	$\boldsymbol{p \wedge q}$	$\neg q$	
1	T	T	T	F	**x**
2	T	F	F	T	
3	F	T	F	F	
4	F	F	F	T	

Technically, we can stop at this point, because we have already shown the conclusion to be false when all other arguments are true. Therefore, this proof is invalid.

Just for fun, let's complete the truth table.

	$\boldsymbol{p}$	q	$\boldsymbol{p \wedge q}$	$\neg q$	
1	T	T	T	F	**x**
2	T	F	F	T	**-**
3	F	T	F	F	**-**
4	F	F	F	T	**-**

Notice that even where the conclusion was true, if the arguments aren't also true, that line is ignored.

In some cases, you may have to test each line before finding a false conclusion. There is no way to determine in advance whether a line will be valid, ignored, or cause the proof to be invalidated. That is why it is called "exhaustive".

Resolution Proof

In *resolution proof* we are looking to examine a proof for validity by symbolic means. That is, we will manipulate the arguments in a manner similar to algebra in an attempt to arrive at the same conclusion.

Let's look an example:

$$\begin{array}{l} p \\ p \Rightarrow q \\ \hline \therefore q \end{array}$$

If we were to create a truth table, we would see that this is true. In fact, this proof is common enough that mathematicians have given it a name: ***modus ponens***. A table of common rules of inference are listed below. Each can be proven using direct proof.

$\begin{array}{l} p \Rightarrow q \\ p \\ \hline \therefore q \end{array}$	Modus ponens
$\begin{array}{l} p \Rightarrow q \\ \neg q \\ \hline \therefore \neg p \end{array}$	Modus tollens
$\begin{array}{l} p \Rightarrow q \\ p \\ \hline \therefore q \end{array}$	Addition
$\begin{array}{l} p \wedge q \\ \hline \therefore p \end{array}$	Simplification
$\begin{array}{l} p \\ q \\ \hline \therefore p \wedge q \end{array}$	Conjunction
$\begin{array}{l} p \Rightarrow q \\ q \Rightarrow r \\ \hline \therefore p \Rightarrow r \end{array}$	Hypothetical syllogism
$\begin{array}{l} p \vee q \\ \neg p \\ \hline \therefore q \end{array}$	Disjunctive syllogism

Most proofs depend on an additional rule proposed by J. Robinson that looks like this:

if $p \vee q$ and $\neg p \vee r$ are both true, then $q \vee r$ is true

or,

$$\begin{array}{l} p \vee q \\ \neg p \vee r \\ \hline \therefore\ q \vee r \end{array}$$

With that, let's take a look at another example.

1	$a \vee b$
2	$\neg a \vee c$
3	$\neg c \vee d$
	$\therefore\ b \vee d$

By applying Robinson's rule to lines 1 and 2, we can simplify…

1	$a \vee b$		
2	$\neg a \vee c$	4	$b \vee c$
3	$\neg c \vee d$		$\neg c \vee d$
	$\therefore\ b \vee d$		$\therefore\ b \vee d$

…creating a line 4. By applying Robinson's rule to lines 3 and 4, we can simplify again…

1	$a \vee b$				
2	$\neg a \vee c$	4	$b \vee c$		
3	$\neg c \vee d$		$\neg c \vee d$	5	$b \vee d$
	$\therefore\ b \vee d$		$\therefore\ b \vee d$		$\therefore\ b \vee d$

…creating line 5. Line 5 is the same as our conclusion, so we have verified that our proof is valid. QED.

An important concept to keep in mind when working with resolution proof is that you are working towards an equivalent expression. As in algebra, it is easy to get locked into a circular set of identities that lead you nowhere.

Another example:

1	$a \vee \neg b\ c$
	$\neg a$
	$\therefore \neg b$

The expression in line 1 can be interpreted as

$$a \vee (\neg b \wedge c)$$

Using the distributive law, it can be re-written as

$$(a \vee \neg b) \wedge (a \vee c)$$

Now each half can occupy a separate line because they are separated by a logical *and* according to the rule of conjunction.

1	$a \vee \neg b$
2	$a \vee c$
3	$\neg a$
	$\therefore \neg b$

By applying the rule of disjunctive syllogism to lines 1 and 3 we immediately get $\neg b$. Because all of the arguments are assumed to be true, and this matches our conclusion, the proof is valid. QED.

A significant difference between direct proof and resolution proof is that with direct proof it is relatively easy to determine that a proof is not valid. However, with resolution proof, you cannot really determine that a proof is invalid, you can only prove that it *is valid*. This is because there ***may*** be a sequence of inferences that you have overlooked and not determined, but that doesn't mean that one doesn't exist. On the other hand, with direct proof you can be absolutely sure because it tests for every single condition.

A utility of the techniques associated with resolution proof is the ability to simplify a series of conditions. This can have significant beneficial effects on a program or database search.