

PART IIA: PROPOSITIONAL LOGIC

Version 2.43: 2003/02/12

Peggy Fidelman
John Greiner
Ian Barland

This work is produced by The Connexions Project and licensed under the
Creative Commons Attribution License *

Abstract

(Blank Abstract)

Part Iia: Propositional Logic

Recall examples of where we'd like proofs:

- WaterWorld (Is a certain location guaranteed safe?)
- type checking (Does a program call functions in the proper way?)
- circuit verification (Does a circuit always work as advertised?)

After seeing the reasons why proofs are important, we ended with a call for first needing a precise language for writing down statements without the ambiguity of English.

Might a programming language be a good way to specify formal concepts without ambiguity? Programming languages are usually motivated by specifying *how* to do something, rather than simply stating facts. While there is a deep relation between specifying and solving problems, we'll choose instead to use logic to capture what we want to express.

1 A formal vocabulary

Imagine an offer where, for a mere \$6.99, you can get EE, (FF or CF or OB or HB) or CC and PH and BR and GR or WB and PJ. Unfortunately, it's not clear at all how the "and" and "or"s relate. Fundamentally, is " x and y or z " meant to be interpreted as " $(x$ and $y)$ or z ", or as " x and $(y$ or $z)$ "? With some context, we might be able to divine what the author intended: the above offer is the direct translation from the menu of a local diner (House of Pies¹):

QUOTATION: 2 eggs, potatoes (french fries, cottage fries, OBrien or hashed brown) or cottage cheese and peach half (grits before 11am) and choice of bread with gravy or whipped butter and premium jam.

*<http://creativecommons.org/licenses/by/1.0>

¹<http://houston.citysearch.com/profile/9953699/>

Though even with this context, the offer isn't entirely clear to everybody: is it possible to get both cottage cheese and cottage fries? Fortunately, coffee is available before having to decipher the menu. In general, parentheses might be required in a language, to keep it unambiguous. We'll remember later, that a well-specified language might need explicit parentheses. But first, let's look at our distillation of propositions like "your meal includes hashbrowns" to the variable HB:

Definition 1: proposition

A statement which can be either true or false.

Example:

"Your meal will include hashbrowns."

Definition 2: propositional variable

A variable that can either be true or false, representing whether a certain proposition is true or not.

Example:

HB

We will often refer to "propositional variables" as just plain ol' "propositions", since our purpose in studying logic is to abstract away from individual statements and encapsulate them in a single variable, thereon only studying how to work with the variable.

For a proposition or propositional variable X , rather than write " X is true", it is more succinct to simply write " X ". Likewise, " X is false" is indicated as " $\neg X$ ".

ASIDE: Compare this with Boolean variables in a programming language. Rather than `(x == true)` or `(x == false)`, it's idiomatic to instead write `x` or `!x`.

Observe that not all English statements are propositions, since they aren't true/false issues. Which of the following do you think might qualify as propositions? If not, how might you phrase similar statements that are propositions?

- "Crocodiles are smaller than Alligators."
- "What time is it?"
- "Pass the salt, please."
- "Hopefully, the Rice Owls will win tomorrow's game."
- "Mr. Burns is filthy rich."
- "Fresca@@ is the bee's knees."

1.1 A particular vocabulary for WaterWorld

When playing WaterWorld, what particular propositions are involved? To consider this, we think of a generic board, and wonder what the underlying statements are. They are statements like "location A contains a pirate" (" A – unsafe"), "location G has 2 adjacent pirates" (" G – has – 2") and so on. Each of these statements may be true or false, depending on the particular board in question.

Here are all the WaterWorld propositions² that we'll use.

²<http://cnx.rice.edu/content/m10528/latest/>

Remember that $B - \text{unsafe}$ doesn't mean "I'm not sure whether or not B is safe"; rather it means " B is unsafe." You may not be sure whether (the truth of) this proposition follows what you see; that is what we are trying to prove.

Observe that any WaterWorld board has the same set of propositions to describe it: $A - \text{unsafe}$, $B - \text{has} - 2$, etc. However, different boards will have different underlying values of those propositions.

1.2 Connectives

Some statements in the above proof were simple: e.g., the single proposition " $A - \text{has} - 2$ ". Some statements had several parts, though: " $(F - \text{unsafe and } G - \text{unsafe})$ ". We build these more complicated statements out of propositions. If you know both $F - \text{unsafe}$ is false, and $G - \text{unsafe}$ is false, what can you deduce about the truth of the statement " $(F - \text{unsafe and } G - \text{unsafe})$ "? Clearly, it is false. What if $F - \text{unsafe}$ is false, but $G - \text{unsafe}$ is true? If both propositions are true? In fact, we can fill in the following table:

Truth table for \wedge ("and")

a	b	$(a \wedge b)$
false	false	false
false	true	false
true	false	false
true	true	true

Definition 3: truth table

An expression involving, say, two propositions a , b has two input columns (each row containing a different true/false combination of a , b), and one output column: the truth of the entire expression for that row.

Exercise 1:

What do you think the truth table for " a or b " looks like? Hint: To fill out one row of the table, say, for $a=\text{true}$ and $b=\text{false}$, ask yourself "For this row, is it true that (a is true, or b is true)?"

Solution:

Truth table for \vee ("or")

a	b	$(a \vee b)$
false	false	false
false	true	true
true	false	true
true	true	true

Exercise 2:

The above proof also used subexpressions of the form "not b -unsafe". What is the truth table for "not a "?

Solution:

Truth table for \neg ("not")

a	$\neg a$
false	true
true	false

Exercise 3:

What is the truth table for the expression "(not a), or b "?

Solution:

Truth table for \rightarrow ("implies")

a	b	$(a \rightarrow b)$
false	false	true
false	true	true
true	false	false
true	true	true

Definition 4: connective

1. The syntactic operator combining one or more logical expressions into a larger expression.

Example:

Two operators are \wedge and \vee .

2. A function with one or more Boolean inputs and a Boolean result. I.e., the meaning of a syntactic operator.

Example:

The meaning of " \wedge " and " \vee ", e.g., as described by their truth tables.

Example:

"nand" (mnemonic: "not and") takes in two Boolean values a and b , and returns true exactly when $(a \wedge b)$ is *not* true – that is, when $\neg(a \wedge b)$ is true.

The following are the connectives we will use most often. At least some of these should already be familiar from Boolean conditional expressions.

Connectives

Connective	Pronunciation	Meaning	Alternative pronunciations / notations
\neg	not	$\neg a$: a is false	$\text{not}(a)$, $\neg a$, $!a$
\wedge	and	$(a \wedge b)$: a and b are both true	$\text{and}(a,b)$, $a*b$, ab , $a\&b$, $a\&b$
\vee	or	$(a \vee b)$: at least one of $\{a, b\}$ is true	$\text{or}(a,b)$, $a+b$, $a \text{---} b$, $a \text{---} b$
\rightarrow	implies	$(a \rightarrow b)$: equivalent to $(\neg a \vee b)$	if a then b , a only if b , b if a , b is necessary for a

Many other connectives can also be defined. In fact, it turns out that any connective for propositional logic can be defined in terms of those above.

Example 6:

Another connective is **if-and-only-if** or **iff**, written as $(a \leftrightarrow b)$, which is true when a and b have the same truth value. So, as its name implies, it can be defined as $((a \rightarrow b) \wedge (b \rightarrow a))$. It is also commonly known as "a is equivalent to b" and "a is necessary and sufficient for b".

Exercise 4:

Another connective is "exactly-one-of", which is more traditionally called **exclusive-or** or **xor** (since it excludes both a and b holding, unlike the traditional "inclusive" or.) How would you define a "xor" b in terms of the above connectives?

Solution:

Exactly one is true if either (a is true, and b is false) or (a is false, and b is true). So, $(a \text{ "xor" } b) \equiv ((a \wedge \neg b) \vee (\neg a \wedge b))$.

ASIDE: We'll use the symbol \equiv to express the notion of two formulas being **equivalent** – that is, the same for all truth-settings. \equiv is not part of a WFF, but instead says something about two separate WFFs. Use \leftrightarrow if you're inside a formula. Commonly, people use the symbols " \equiv " or " \Leftrightarrow " sometimes as a logical connective, and sometimes as a relation between formulas. This is tolerable for many subjects, but it's rather problematic for logic, where part of what we are studying are the (syntactic) formulas themselves. For the logic portion of this class, we'll avoid using \Leftrightarrow , to duck that ambiguity. Finally, \Rightarrow and \Leftrightarrow operate *between* formulas, not *within* formulas.

1.3 Well-Formed Formulas

Finally, out of simple propositions, we'll build more complicated formulas, but require them to be fully-parenthesized:

Definition 5: Well-Formed Formula (WFF)

1. A constant: true or false. (If you prefer brevity, you can write "T" or "F".)
2. A propositional variable.

Example:

a

3. A negation $\neg\phi$, where ϕ is a WFF.

Example:

$\neg c$

4. A conjunction $(\phi \wedge \psi)$, where ϕ and ψ are WFFs.

Example:

$(a \wedge \neg c)$

5. A disjunction $(\phi \vee \psi)$, where ϕ and ψ are WFFs.

Example:

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

6. An implication $(\phi \rightarrow \psi)$, where ϕ and ψ are WFFs.

Example:

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

Example 12:

We can combine these ways of forming WFFs in arbitrarily complex ways, for example,

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

While *large* WFFs are common, and we will use some, ones with this much nesting are not.

ϕ , ψ , and θ are **meta-variables** standing for any WFF. Similarly, a , b , and c are meta-variables standing for any proposition. They are variables which we the readers substitute with particular values. The literal character " ϕ " doesn't actually show up inside some WFF. *Within* the system, a WFF might be a propositional variable like B .

Variations of well-formed formulas occur routinely in writing programs. While different languages might vary in details of what connectives are allowed, how to express connectives, and whether or not all parentheses are required, they all use WFFs.

Example 13:

When creating the homeworks' web pages, the authors keep the problems and solutions together in one file. Then, a program reads that file, and creates a new one which either excludes the solution (for the problem set), or includes it (for the solution set, and for practice-problems). The condition for deciding whether to include the solutions is a WFF.

```
;; is-a-solution?: paragraph --> boolean
;; A function to tell if we are looking at a "solution" paragraph.
;; Assume this is provided.

;; is-in-a-practice-prob?: paragraph --> boolean
;; A function to tell if Is the current problem a practice problem?
;; Assume this is provided.

;; include-all-solutions?: boolean
;; A variable for the entire file.
;; Assume this is provided.

;; show-or-hide-soln: paragraph --> paragraph
;; Either return the given paragraph,
;; or (if it shouldn't be revealed) return a string saying so.
;;
(define (show-or-hide-soln a-para)
```

```
(if (and (is-a-solution? a-para)
        (not (or include-all-solns? (is-in-a-practice-prob? a-para))))
    "(see solution set)"
    a-para))
```

Note that the Boolean variable `include-all-solutions?` and Boolean values of `(is-a-solution? a-para)` and `(is-in-a-practice-prob? a-para)` play the part of propositions ("is – soln", "include – solns", "is – practice"), respectively. The `if`'s condition boils down to the WFF $(\text{is} - \text{soln} \wedge \neg(\text{include} - \text{solns} \vee \text{is} - \text{practice}))$.

Keep in mind that a WFF is purely a syntactic entity – we'll introduce rules later, for re-writing or reasoning with WFFs, but it's those rules that will be contrived to preserve our meaning of connectives like \wedge or \neg . The truth value of a WFF depends on the truth values we assign to the propositions it involves.

Writing a program about WFFs – verifying syntactic property, calculating a value, counting the number of negations or b , ... Such programs exactly follow the definition of WFF given.

1.4 Some formulas are truer than others

Is the formula $(A - \text{unsafe} \vee A - \text{has} - 2)$ true? Your response is, "Well, it's contingent on the particular board in question". But some formulas are true regardless of the board. For instance, $(A - \text{unsafe} \vee \neg A - \text{unsafe})$: this is clearly true no matter what. Similarly, $(A - \text{unsafe} \wedge \neg A - \text{unsafe})$ can never be satisfied (made true), no matter how you try to set the variable $A - \text{unsafe}$.

Definition 6: tautology

A WFF which is true under any truth assignment (any way of assigning true/false to the propositions).

Example:

$(A - \text{unsafe} \rightarrow A - \text{unsafe})$

Example:

$(a \rightarrow (a \vee b))$

Definition 7: unsatisfiable

A WFF which is false under any truth assignment.

Example:

$\neg(A - \text{unsafe} \rightarrow A - \text{unsafe})$

Example:

$(G - \text{unsafe} \rightarrow \neg G - \text{unsafe})$

Some people use the term **contingency** to mean the things inbetween: formulas which can be either true or false, depending on the truth assignment. Really, tautologies and unsatisfiable formulas are boring. However, trying to determine whether or not a formula is a tautology (or, unsatisfiable) is of interest: really, that's what proofs are all about.

Identify the following Yogi Berra quotes either as tautologies, unsatisfiable, or neither. (Take these exercises with a grain of salt, since the English statements are open to some interpretation.)

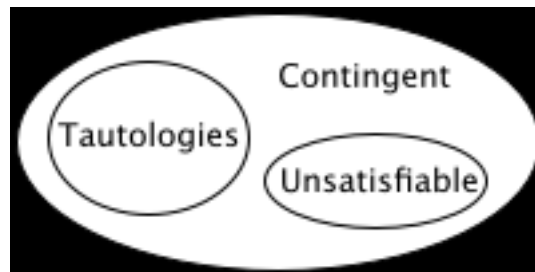


Figure 1: Kinds of formulas: tautologies, contingencies, unsatisfiables

Exercise 5:

Pitching always beats batting – and vice-versa.

Solution:

Unsatisfiable.

Exercise 6:

You can observe a lot just by watchin’.

Solution:

Tautology, arguably.

Exercise 7:

Nobody goes there anymore... it’s too crowded.

Solution:

Unsatisfiable, unless of course you interpret "nobody" as "nobody of note".

Exercise 8:

It sure gets late early out here.

Solution:

Neither. If you interpret "gets late" as a social issue but "early" as a clock issue, then the statement might be true, depending on where "here" is.

Exercise 9:

Always go to other people’s funerals; otherwise they won’t come to yours.

Solution:

Unsatisfiable, except perhaps in a karmic³ sense.

³<http://www.cs.rice.edu/~ian/Rants/karmaIsReal.shtml>

1.5 Game-specific rules

Is $(x \vee y \vee z)$ a tautology? Clearly not – by setting the three propositions each to false, the formula is false. But now consider: Is $(A - \text{has} - 0 \vee A - \text{has} - 1 \vee A - \text{has} - 2)$ a tautology? The answer here is "yes of course, ... well, as long we're interpreting those propositions to refer to a WaterWorld board." We'll capture this notion by listing a bunch of **domain axioms** for WaterWorld: formulas which are true for all WaterWorld boards.

There are a myriad of domain axioms which express the rules of WaterWorld. Here are a few of them:

- $(A - \text{has} - 0 \leftrightarrow (B - \text{safe} \wedge F - \text{safe} \wedge G - \text{safe}))$,
- $(A - \text{has} - 2 \leftrightarrow ((B - \text{safe} \wedge F - \text{safe} \wedge \neg G - \text{safe}) \vee (B - \text{safe} \wedge \neg F - \text{safe} \wedge G - \text{safe}) \vee (\neg B - \text{safe} \wedge F - \text{safe} \wedge G - \text{safe})))$,
- ... ,

A more complete list is here⁴.

⁴<http://cnx.rice.edu/content/m10528/latest/#waterworld-rules>