Supplementary Section 6S.8

Adequacy

CHOOSING OPERATORS FOR FORMAL LANGUAGES

There are many different possible logical languages and formal systems and the same logical truths and inferences may be expressed in many different ways. For example, **PL** has only twenty-six variables, but we can devise methods to construct indefinitely many variables by indexing. We can also choose different logical operators. In this section, we will look at alternative languages for propositional logic, with an eye to reducing the number of operators from the five of **PL** without weakening our language and its powers of expression. Along the way we will do some metalogical proofs that can give you a taste of the kind of work that can come next in logic.

PL uses one unary operator, ~, and four binary operators: •, ∨, ⊃, and ≡. There are only four possible unary operators in a bivalent logic (i.e., a logic with two truth values). Given one variable, and a truth table of two rows, there are only four possible distributions of truth values. Any unary operator will produce one of the four tables U1–U4.

U1

	α
1	1
1	0

U2

	α
1	1
0	0

U3

~	α
0	1
1	0

U4

	α
0	1
0	0

Only the negation, U3, is useful and has a common name. U2 just repeats the value of the given formula, and so is otiose. We could call U1 a truth operator, since it takes the value '1' whatever the value of α. But, if we want a formula that produces a truth no matter what the values of the component variables, we can just use any tautology, like an instance of the law of the excluded middle, at 6S.8.1 and so do not need a truth operator in our language.

6S.8.1 P ∨ ~P

Similarly, U4 is a falsity operator, always giving the value ‘0’. If we want a formula to produce falsity, we can use any contradiction, like 6S.8.2.

6S.8.2 P • ~P

The situation is just a bit more complicated with binary operators. There are sixteen possible combinations of truth values for two propositions, sixteen possible truth tables. The following chart presents all of them.

α	β		$\vee$		$\supset$				$\equiv$				•				
1	**1**	1	1	1	1	0	1	1	1	0	0	0	1	0	0	0	0
1	**0**	1	1	1	0	1	1	0	0	1	1	0	0	1	0	0	0
0	**1**	1	1	0	1	1	0	1	0	1	0	1	0	0	1	0	0
0	**0**	1	0	1	1	1	0	0	1	0	1	1	0	0	0	1	0

Notice that the first column after the variables has ‘1’ in all four rows; the next four columns have three ‘1’s and one ‘0’; the next six columns have all possible distributions of two ‘1’s and two ‘0’s; four columns of three ‘0’s and one ‘1’ follow; lastly, there is the single column of all ‘0’s. There are no other possible distributions of ‘1’s and ‘0’s in a four-row truth table.

Only four of the possible sixteen possible truth tables have names in **PL**. We could give names to others and include them in our language. For example, we could add an operator for exclusive disjunction, the column just after the one for the ≡. But the more operators we include, the more truth tables we have to remember. A language can get clunky and awkward with too many elements. Furthermore, when we want to prove theorems about our formal language, it is useful to have as few elements of the language as possible. Given our five operators, we can produce formulas that yield every one of the twelve unnamed columns above, which shows that we need not add more operators to **PL** in order to increase its expressive capacity. It is kind of a fun task.

In the other direction, we might want to reduce the vocabulary of our language. Metalinguistic proofs tend to be easier with a smaller vocabulary. But we want to maintain the expressive capacity of our language. It’s fine to remove some of the vocabulary as long as there are other ways of saying what we want to say. To use an analogy from natural language, we could ban the word ‘bachelor’ from English as long as we still had the words ‘unmarried’ and ‘man’. But, if we got rid of all ways of saying that some man is unmarried, then we would not be able to express some propositions.

When designing logical languages and formal systems, then, we have to balance the ease of using the language with the ease of constructing metalinguistic proofs about the language. We want to make sure both that the language is manageable and that it allows us to say what we want to say.

EXERCISES 6S.8a

There are sixteen possible truth tables for propositions with two variables, listed in the previous chart . We have simple ways to represent four of the sixteen, with the four binary operators: $\alpha \bullet \beta$; $\alpha \vee \beta$; $\alpha \supset \beta$; $\alpha \equiv \beta$. Devise sentences to represent each of the other twelve combinations of truth values. (Hint: You need not use two variables for each; sometimes a formula with a single variable will work.)

1. 1111
2. 1101
3. 0111
4. 1100
5. 1010
6. 0110
7. 0101
8. 0011
9. 0100
10. 0010
11. 0001
12. 0000

ELIMINATING THE BICONDITIONAL AND CONDITIONAL

One natural way to reduce the vocabulary of our language is to eliminate operators, as long as we can construct statements with the same truth values that those operators produced. We have seen how to eliminate the biconditional by defining it in terms of the conditional and the conjunction. This was the first of the rules we called material equivalence. Call an operator **superfluous**, relative to a given language, if it can be defined in terms of other operators of the language. With this definition, we can prove the metatheorem 6S.8.3.

6S.8.3 The biconditional is superfluous in **PL**.

To prove 6S.8.3, we just need to show that '$\alpha \equiv \beta$' and '$(\alpha \supset \beta) \bullet (\beta \supset \alpha)$' are logically equivalent. We can do this by method of truth tables.

α	$\equiv$	β
1	**1**	1
1	**0**	0
0	**0**	1
0	**1**	0

$(\alpha$	$\supset$	$\beta)$	$\bullet$	$(\beta$	$\supset$	$\alpha)$
1	1	1	**1**	1	1	1
1	0	0	**0**	0	1	1
0	1	1	**0**	1	0	0
0	1	0	**1**	0	1	0

Any statement of the form '$\alpha \equiv \beta$' can thus be replaced with a more complex statement that does not use '$\equiv$'.

Notice that the conditional is superfluous also, which we can see by considering the equivalence we called the rule of material implication.

6S.8.4 The conditional is superfluous in **PL**.

α	$\supset$	β
1	**1**	1
1	**0**	0
0	**1**	1
0	**1**	0

$\sim$	α	$\vee$	β
0	1	**1**	1
0	1	**0**	0
1	0	**1**	1
1	0	**1**	0

Let's call the above method of proving 6S.8.3 and 6S.8.4 semantic, for their uses of the truth values of **PL**. An alternative way of proving 6S.8.4 uses metalinguistic versions of conditional and indirect proof but also appeals to the semantics for **PL**. To show that two statements are logically equivalent, in this second way, we show that each entails the other. For 6S.8.4, first we assume '$\alpha \supset \beta$' is true and show that the truth of '$\sim\alpha \vee \beta$' follows. Assume not. Then some formula of the form '$\sim\alpha \vee \beta$' is false. Then the formula replacing α will have to be true (to make $\sim\alpha$ false) and the formula replacing β will have to be false. But, those values will make '$\alpha \supset \beta$' false, contradicting our assumption.

Next, we assume '$\sim\alpha \vee \beta$' is true and show that the truth of '$\alpha \supset \beta$' follows. Assume that some formula of the form '$\alpha \supset \beta$' is false. Then the value of the formula replacing α must be true and the value of the formula replacing β must be false. But, on those values, '$\sim\alpha \vee \beta$' is false, again contradicting our assumption. QED.

We can prove 6S.8.3 by the same method. First we assume '$\alpha \equiv \beta$' and show that '$(\alpha \supset \beta) \bullet (\beta \supset \alpha)$' follows. Then, we assume '$(\alpha \supset \beta) \bullet (\beta \supset \alpha)$' and show that '$\alpha \equiv \beta$' follows. I leave the details to the reader.

Both of these semantic methods for proving 6S.8.3 and 6S.8.4 produce the same results, since they depend on the same truth values. A third method of proving the equivalence of two statements, one that does not invoke semantics directly, is to derive one from the other using the rules of inference we introduced in chapter 3. For 6S.8.4, for example, we assume '$\alpha \supset \beta$ and derive '$\sim\alpha \vee \beta$'; then we assume '$\sim\alpha \vee \beta$' and derive '$\alpha \supset \beta$'. In this case, our proof will use metalinguistic formulas rather than formulas of **PL**, but the same rules apply.

So, as a methodological observation, we have on our hands two distinct notions of logical equivalence.

LE_1 Two statements are logically equivalent if, and only if, they have the same values in every row of the truth table.

LE_2 Two statements are logically equivalent if, and only if, each is derivable from the other.

We hope that LE_1 and LE_2 yield the same results. In order to show that this is the case, we must show that our formal system is sound.

Combining 6S.8.3 and 6S.8.4, we discover that any sentence that can be written as a biconditional can be written in terms of negation, conjunction, and disjunction. To eliminate the biconditional and the conditional from a sentence like the first at 6S.8.5, which is naturally translated using the ≡, we use two steps. After regimenting the proposition directly as a biconditional, we eliminate the biconditional and then get rid of the conditional.

6S.8.5 Dogs bite if, and only if, they are startled.

$B \equiv S$

$(B \supset S) \bullet (S \supset B)$

$(\sim B \vee S) \bullet (\sim S \vee B)$

Several questions may arise from these considerations about eliminating operators. First, how can we be sure that all sentences can be written with just the five (or, now, three) operators of **PL**? Second, can we eliminate more operators? What is the fewest number of operators that we need? We will answer these questions in the remainder of this section.

DEFINING ADEQUACY AND DISJUNCTIVE NORMAL FORM

A set of operators is called **adequate** if, and only if, corresponding to every possible truth table there is at least one sentence using only those operators. By "every possible truth table," I mean every combination of '1's and '0's in the column under the main operator. We want our operators to be adequate so that we can construct formulas with all possible truth conditions. If our set of operators is adequate, then our propositional logic will be able to say anything that any propositional logic can say.

To give you a taste of what we are after, consider a severely limited adequacy result.

6S.8.6 Negation and conjunction are adequate in languages with only one propositional variable.

We can prove 6S.8.6 by sheer force. There are only four possible truth tables: 11, 10, 01, 00. Here are statements for each of them that use no operators other than negation and conjunction.

~	(α	•	~	α)
1	1	0	0	1
1	0	0	1	0

α
1
0

~	α
0	1
1	0

α	•	~	α
1	**0**	0	1
0	**0**	1	0

Returning to our discussion of languages with any number of variables, we want to demonstrate the general theorem that the five operators of **PL** are adequate. By 6S.8.3 and 6S.8.4, we know that the five operators are adequate if, and only if, the three (negation, conjunction, and disjunction) are adequate.

In order to prove a general adequacy theorem, we will consider the set of wffs of **PL** that are in *disjunctive normal form* (DNF). A sentence is in DNF if it is:

a single letter or a negation of a single letter; or
a conjunction of single letters or negations of single letters; or
a disjunction of single letters, negations of single letters, or conjunctions of single letters or negations of single letters; or
a series of such disjunctions.

6S.8.7 lists some sentences in DNF.

6S.8.7 Some statements that are in DNF
P
P • ~Q
~P ∨ Q
(P • Q) ∨ (~P • Q)
~P ∨ ~Q ∨ (~P • ~Q)

Note that we can drop brackets here among three or more conjuncts or disjuncts, though we still need brackets when conjoining disjunctions or disjoining conjunctions. 6S.8.8 lists some sentences that are not in DNF.

6S.8.8 Some statements that are *not* in DNF
~(P • Q)
P ⊃ Q
(P • ~Q) ∨ (~P ≡ Q)
(P ∨ Q) • (~P ∨ ~Q)
P ∨ ~Q ∨ ~(P ∨ Q)

Notice that the first and last sentences in 6S.8.8 are logically equivalent to related sentences in DNF. '~(P • Q)' is logically equivalent to '~P ∨ ~Q', which is in DNF. 'P ∨ ~Q ∨ ~(P ∨ Q)' is logically equivalent to 'P ∨ ~Q ∨ (~P • ~Q)', which, again, is in DNF. These equivalences are easily shown by constructing the appropriate truth tables or applying De Morgan's law.

EXERCISES 6S.8b

Which of the following sentences are in DNF?

1. (P • ~Q) ∨ (P • Q)
2. (P • Q • R) ∨ (~P • ~Q • ~R)
3. ~P ∨ Q ∨ R
4. (P ∨ Q) • (P ∨ ~R)
5. (P • Q) ∨ (P • ~Q) ∨ (~P • Q) ∨ (~P • ~R)
6. (~P • Q) • (P • R) ∨ (Q • ~R)
7. (P • ~Q • R) ∨ (Q • ~R) ∨ ~Q
8. ~(P • Q) ∨ (P • R)
9. P • Q
10. ~P

PROVING ADEQUACY AND INADEQUACY FOR FAMILIAR OPERATORS

We have seen that the set of five operators of **PL** is adequate if the set of three {~, •, ∨} is. The proof of 6S.8.9 will show that both sets are indeed adequate.

6S.8.9 The set of negation, conjunction, and disjunction {~, •, ∨} is adequate.

The proof of 6S.8.9 proceeds by cases. We will see a way to construct a sentence using only the three propositional operators for any possibility of combinations of truth values in any truth table.

For any size truth table, with any number of propositional operators, there are three possibilities for the column under the main operator.

Case 1: Every row is false.
Case 2: There is one row that is true, and every other row is false.
Case 3: There is more than one row that is true. (Perhaps even all the rows are true.)

For Case 1, conjoin any variable with its negation. If you want to use all of a set of variables, you can conjoin them to the resulting contradiction. So, if you have variables P, Q, R, S, and T, you can write, 'P • ~P • Q • S • T'. The resulting formula, in DNF, is false in every row, since each row contains a contradiction. It uses only conjunction and negation.

For Case 2, consider the row in which the statement is true. Then, write a conjunction which uses of all of the propositional variables in the original formula or their negations in the following way. If the variable is true in the row in which the complex proposition is true, use the variable itself. If the variable is false in that row, use its negation instead.

The resulting formula is in DNF and is true in only the prescribed row. For example, consider a formula with two variables, P and Q, and the column under the main operator, shown below. The formula may be as complicated as we wish. We could be considering '~(~P ∨ ~~Q)' or '~[(P ∨ P) ⊃ (Q • ~~Q)]', each of which yields the given truth table. We are concerned to construct a formula, in DNF, which matches the single column under the main operator in any such formula of **PL**.

P	Q	Main Operator
1	1	0
1	0	1
0	1	0
0	0	0

We consider the second row only, in which P is true and Q is false. Our conjunction will be 'P • ~Q'. This formula is in DNF and it is logically equivalent, by definition, to whatever the original sentence was, no matter which of the five propositional operators it used. In addition to the two I mentioned above, many different formulas will yield the same truth table. In fact, there are infinitely many ways to produce each truth table. We need just one for this proof.

For Case 3, we just repeat the method from Case 2 for each row in which the statement is true. Then, we form the disjunction of all the resulting formulas. Again, the resulting formula will be in DNF and will be logically equivalent to the original formula, no matter which operators it used. Here is an example.

P	Q	R	Main Operator
1	1	1	1
1	1	0	0
1	0	1	0
1	0	0	1
0	1	1	0
0	1	0	0
0	0	1	0
0	0	0	0

To construct a formula with that truth table, we need to consider only the first and fourth rows. In the first row, all variables are true. In the fourth, 'P' is true, but 'Q' and 'R' are false. Our resultant formula will be '(P • Q • R) ∨ (P • ~Q • ~R)'. Punctuation can easily be added to make the formula well formed in **PL**.

Thus, we have a method for producing a formula of **PL**, in DNF, for any possible truth table. Since DNF uses only negation, conjunction, and disjunction, the set {~, •, ∨} is adequate, as 6S.8.9 claims. QED.

Given 6S.8.9 and the methods used in the proofs of the theorems at 6S.8.3 and 6S.8.4, we can easily prove several other sets of operators adequate.

6S.8.10 The set {∨, ~} is adequate.

To prove 6S.8.10, we use can use the method in the proof of 6S.8.9 to write a formula for any truth table using as operators only those in the set {∨, •, ~}. Any statement of the form 'α • β' is equivalent to one of the form '~(~α ∨ ~β)'. So, we can replace any occurrence of '•' in any formula, according to the above equivalence. Whitehead and Russell take the set {∨, ~} as their official operators in *Principia Mathematica*, though they introduce other operators by definition.

The proofs of 6S.8.11 and 6S.8.12 are just as straightforward. For 6S.8.11, we require a formula using conjunction and negation that is logically equivalent to '$\alpha \vee \beta$'. 6S.8.12 is a little trickier. I leave the proofs of both to the reader.

6S.8.11 The set {•, ~} is adequate.
6S.8.12 The set {~, ⊃} is adequate.

We have seen that some pairs of operators are adequate to express any truth table. Not all sets of pairs of operators are adequate. To show that a set of operators is inadequate, we can show that there is some truth table that cannot be constructed using those operators.

6S.8.13 The set {⊃, ∨} is inadequate.

Recall that both '$\alpha \supset \beta$' and '$\alpha \vee \beta$' are true when α and β are both true. Thus, using these operators we cannot construct a truth table with a false first row. {⊃, ∨} is inadequate, as 6S.8.13 says.

All of the sets of single operators in **PL** are inadequate. 6S.8.14 is an example.

6S.8.14 The set {⊃} is inadequate.

As with 6S.8.13, we cannot construct a truth table with 0 in the first row using just ⊃. To see this argument in a bit more careful detail, consider the truth table for conjunction. We want to construct a formula, using ⊃ as the only operator, which yields the same truth table, which we can write as 1000. Imagine that we have such a formula, and imagine the smallest such formula. Since the only way to get a 0 with ⊃ is with a false consequent, the truth table of the consequent of our formula must either be 1000 or 0000. Since we are imagining that our formula is the smallest formula that yields 1000, the consequent of our formula must be the latter, a contradiction. But you cannot form a contradiction using ⊃ alone, since $1 \supset 1$ and $0 \supset 0$ are both true. Since we cannot construct the contradiction, we cannot construct the conjunction.

We will need one more inadequate set.

6S.8.15 The set {∼} is inadequate.

To prove 6S.8.15, we need only one variable. The only possible truth tables with one variable and ∼ are 10 and 01. Thus, we cannot generate 11 or 00.

ADEQUACY FOR NEW OPERATORS

Despite the inadequacy of our single propositional operators, there are sets of single operators that are adequate. Consider the Sheffer stroke, '|', which is also called alternative denial, or not-both.

α	|	β
1	0	1
1	1	0
0	1	1
0	1	0

6S.8.16 The set {|} is adequate.

To prove 6S.8.16, notice that '$\sim\alpha$' is logically equivalent to '$\alpha \mid \alpha$' and that '$\alpha \bullet \beta$' is logically equivalent to '$(\alpha \mid \beta) \mid (\alpha \mid \beta)$'. By 6S.8.11, {∼, •} is adequate. 6S.8.16 follows.

There is one more adequate single-membered set consisting of just the Peirce arrow, '$\downarrow$', also called joint denial, or neither-nor.

α	$\downarrow$	β
1	0	1
1	0	0
0	0	1
0	1	0

6S.8.17 The set $\{\downarrow\}$ is adequate.

The proof of 6S.8.17 uses 6S.8.10 and the equivalence of '$\sim\alpha$' to '$\alpha \downarrow \alpha$' and of '$\alpha \vee \beta$' to '$(\alpha \downarrow \beta) \downarrow (\alpha \downarrow \beta)$'.

Both | and $\downarrow$ were initially explored by C. S. Peirce, though Henry Sheffer's name is attached to the former for his independent work on it. Given that both the Sheffer stroke and the Peirce arrow are adequate, we could build systems of propositional logic around just those operators. Translations between such logical languages and English would be difficult, and our propositions would get complex quickly. We have to balance the virtues of having fewer operators with the virtues of languages with which it is easier to work. Also, we could easily add either the Sheffer stroke or the Peirce arrow to **PL**; they would be superfluous just like the biconditional.

THE LIMIT OF ADEQUACY

Lastly, we can prove that there are no other single, adequate operators, at 6S.8.18.

6S.8.18 $\downarrow$ and | are the only operators that are adequate by themselves.

Imagine we had another adequate operator, #. We know the first rows must be false, by the reasoning in the proofs of 6S.8.13 and 6S.8.14. Similar reasoning fills in the last row.

α	#	β
1	0	1
1		0
0		1
0	1	0

Thus, '~α' is equivalent to 'α # α'. Now, we need to fill in the other rows. If the remaining two rows are 11, then we have '|'. If the remaining two rows are 00, then we have '↓'. So, the only other possibilities are 10 and 01. 01 yields 0011, which is just '~α'. 10 yields 0101, which is just '~β'. By 6S.8.15, {~} is inadequate. QED.

Summary

PL uses five operators, though we can express the same claims with more operators or fewer, as few as the single Sheffer stroke or Peirce arrow. With more operators, our formulas can be shorter; with fewer operators, our formulas lengthen. Our selections of operators for a logic are to some degree arbitrary. As with our choices of rules of inference and equivalence, we construct systems that are aesthetically pleasing and that represent, to some extent, operations in natural language.

TELL ME MORE

- What are the constraints on adequacy called soundness and completeness? See 6.4: Metalogic.

For Further Research and Writing

1. While there are no other adequate single sets of operators, there are other binary operators. Are there other adequate pairs? If so, which? If not, why not?
2. What are the meanings of the other possible binary operators? Can a good argument be made to use any others in translation from natural language into a formal language?
3. Why are there only unary and binary propositional operators?
4. Construct a formal language with the same expressive powers as **PL**, but with none of the standard operators. Discuss the challenges and results.
5. For a little more formal work, you may:
    a. Use the metalinguistic, semantic form that I used to prove 6S.8.4 to prove 6S.8.3;
    b. Prove 6S.8.11; or
    c. Prove 6S.8.12.

Suggested Readings

Haack, Susan. *Philosophy of Logics*. Cambridge: Cambridge University Press, 1978. Chapter 3 has a nice discussion of adequacy.

Hunter, Geoffrey. *Metalogic*. Berkeley: University of California Press, 1971. The results above are mostly contained in section 21. The references below are mostly found there as well. His notation is a bit less friendly, but the book is wonderful and could be the source of lots of papers.

Mendelson, Elliott. *Introduction to Mathematical Logic*, 4th ed. Boca Raton, FL: Chapman & Hall/CRC, 1997. Mendelson discusses adequacy in section 1.3. His notation is less friendly than Hunter's, but the exercises lead you through some powerful results.

Peirce, Charles Sanders. "A Boolean Algebra with One Constant." In *Collected Papers*, volume 4, edited by Charles Hartshorne and Paul Weiss, sections 12–20; see also section 265. Cambridge, MA: Harvard University Press, 1933. Peirce's work on singly adequate unary operators was unpublished, but preceded Post's proof by forty years.

Post, Emil. "Introduction to a General Theory of Elementary Propositions." In *From Frege to Gödel: A Source Book in Mathematical Logic, 1879–1931*, edited by Jean Van Heijenoort, 264–283. Cambridge, MA: Harvard University Press, 1967. The notation is different, but the concepts are not too difficult. It would be interesting to translate into a current notation and present some of the results.

Whitehead, Alfred North, and Bertrand Russell. *Principia Mathematica to *56*. Cambridge: Cambridge University Press, 1997. Whitehead and Russell use disjunction and negation as their basic propositional operators, introducing the others by definition.

SOLUTIONS TO EXERCISES 6S.8A

There are infinitely many possibilities for each, but these are among the simplest. For more on this topic, see Ludwig Wittgenstein, *Tractatus Logico-Philosophicus*, section 5.101.

1. 1111 $\alpha \vee {\sim}\alpha$
2. 1101 $\alpha \vee {\sim}\beta$
3. 0111 ${\sim}(\alpha \bullet \beta)$
4. 1100 α
5. 1010 β
6. 0110 ${\sim}\alpha \equiv \beta$
7. 0101 ${\sim}\beta$
8. 0011 ${\sim}\alpha$
9. 0100 $\alpha \bullet {\sim}\beta$
10. 0010 ${\sim}\alpha \bullet \beta$
11. 0001 ${\sim}\alpha \bullet {\sim}\beta$
12. 0000 $\alpha \bullet {\sim}\alpha$

SOLUTIONS TO EXERCISES 6S.8B

Only 4, 6, and 8 are not in DNF, though 8 could be quickly put into DNF.