6 Validity and Provability in Monadic Predicate Logic

1 Semantics for the quantifiers

We now wish to give a completely precise characterization of the difference between valid and invalid argument-forms in LMPL, that is, of the difference between the case where the conclusion of an LMPL argument-form is a semantic consequence of its premises and the case where it is not. To make our way into this topic, we consider two of the simplest invalid argument-forms:

A: $(\exists x)Fx \,\&\, (\exists x)Gx$
$\therefore (\exists x)(Fx \,\&\, Gx)$

and

B: $(\forall x)(Fx \vee Gx)$
$\therefore (\forall x)Fx \vee (\forall x)Gx$.

It is easy to see that these LMPL forms are invalid, since we can find English instantiations of each which have actually true premises and actually false conclusions. For example, with quantifiers relativized to the domain of people, A is the translation of

A_E: Someone is male and someone is female
∴ Someone is both male and female

while B is the translation of

B_E: Everyone is either male or female
∴ Either everyone is male or everyone is female

The problem with A, then, is that its premise does not require that it be the *same* thing which satisfies 'F' and 'G', while its conclusion does require this; and the problem with B is that its premise does not require every object to satisfy 'F' and does not require every object to satisfy 'G', while its conclusion requires one or the other of these conditions to hold.

The procedure illustrated by A_E and B_E is one way of establishing the invalidity of an argument-form in LMPL: we find an English argument with true premises and false conclusion which has the form in question. But this procedure rapidly becomes useless as argument-forms become more complex. Consequently, we will develop a method of showing that LMPL argument-forms such as A and B are invalid, one which does not depend on finding English arguments with true premises and false conclusions which have those LMPL forms.

The idea behind the method is to describe in an abstract way the essential structure of any situation in which the premises of a given LMPL argument-form are true and its conclusion is false. For instance, returning to A, we can see that if its premise '(∃x)Fx & (∃x)Gx' is to be true and its conclusion '(∃x)(Fx & Gx)' false, there would have to be an object which satisfies 'F' and an object which satisfies 'G', but at the same time there would have to be no object which satisfies both. The *minimum* number of objects needed to realize such a situation is two; there could be more, but the extra objects would be inessential, and we are trying to describe what is *essential* to any situation where the premise of A holds and the conclusion does not. What is essential, then, is that there be two objects, one satisfying 'F' and the other satisfying 'G', thereby verifying the two conjuncts of A's premise; in addition, the object which satisfies 'F' should not satisfy 'G', and the object which satisfies 'G' should not satisfy 'F', thereby falsifying A's conclusion.

What we are calling a 'situation' we shall refer to more formally as an *interpretation* of the argument-form. The collection of objects which occur in the situation, which must include at least one object, is called the *domain of discourse* or *universe of discourse* of the interpretation (since we are dealing with formal arguments in LMPL, we can think of these objects as being of any kind); and when we say which objects satisfy 'F' and which satisfy 'G' we are specifying the *extensions* of the predicates 'F' and 'G', that is, we are saying what things they apply to or 'extend over'. There is a standard way of describing an interpretation, which we can illustrate in connection with A. We use lowercase Greek letters as names of objects in the domain and set-braces '{' and '}' to specify the sets which constitute the domain and the extensions of the predicates. Then the interpretation which shows A to be invalid is the following.

Interpretation: $D = \{\alpha,\beta\}$
$\text{Ext}(F) = \{\alpha\}$
$\text{Ext}(G) = \{\beta\}$

'D' abbreviates 'domain of discourse' and 'Ext(F)' abbreviates "the extension of 'F'" (so the parentheses do duty for the single quotes around 'F'). There is also another way in which the same information can be represented which is perhaps more revealing, at least initially, and that is in the form of a matrix, where we write the domain on the side and the predicate-letters along the top, and indicate by the entries in the matrix which objects satisfy which predicates. The matrix representation of the interpretation just given is displayed at the top of page 172. A '+' at a position in the matrix indicates that the object on the row in question satisfies the predicate of the column in question, a '–' that it does

	F	G
α	+	–
β	–	+

not satisfy it (the technique is borrowed from Gustason and Ulrich, Ch. 5.1).

We can do something similar for argument-form B; we could use the very same interpretation to show that B is invalid, or else this slightly different one:

Interpretation: D = $\{\alpha,\beta\}$
Ext(F) = $\{\beta\}$
Ext(G) = $\{\alpha\}$

The premise of B is true because each object satisfies 'F' or satisfies 'G', but the conclusion is false since both its disjuncts are false: it is not true that both objects satisfy 'F', nor that both satisfy 'G'.

Establishing the invalidity of an argument-form in monadic predicate logic is a two-stage process. First we state the interpretation which shows the argument-form to be invalid, and secondly we *explain why* the interpretation establishes invalidity. Since anyone can fill in a matrix with pluses and minuses, the second stage is required to show that we understand why our proposed interpretation does what we claim it does. In the examples we have been discussing, we have only roughly explained why the interpretations work, whereas what we need is some reliable format in which we can give complete explanations, no matter how complex the example.

Underlying our intuitive grasp of why the two preceding interpretations demonstrate the invalidity of A and B are implicit principles about when quantified sentences are true or false on interpretations. We will now make these principles explicit, and the rest of this section will be devoted to understanding how they are applied—we will return to the construction of 'counterexamples' to other invalid argument-forms in the next section. First we need to define the notion of a sentence's being an *instance* of a quantified sentence *in* an interpretation. In this chapter the domains we consider will all be nonempty finite sets, but we will not presuppose finiteness in our definitions. So let $\ulcorner(Q\nu)\phi\nu\urcorner$ be a (closed) quantified sentence, where Q is either '∀' or '∃', ν is an individual variable, $\phi\nu$ is a wff with all and only occurrences of ν free, and Q is the main connective of $\ulcorner(Q\nu)\phi\nu\urcorner$. Then if $\mathcal{I}$ is an interpretation with domain D = $\{\alpha, \beta, \gamma,\ldots\}$, we form instances of $\ulcorner(Q\nu)\phi\nu\urcorner$ by choosing individual constants t_i from the lexicon of LMPL, a different one for each element of the domain of $\mathcal{I}$, and form the sentence $\phi(t_i/\nu)$. That is, each instance of $\ulcorner(Q\nu)\phi\nu\urcorner$ for a given interpretation $\mathcal{I}$ is obtained by deleting the quantifier prefix $\ulcorner(Q\nu)\urcorner$ and replacing every occurrence of ν in $\phi\nu$ by an LMPL name, using the *same* name throughout, so that we end up with one instance of $\ulcorner(Q\nu)\phi\nu\urcorner$ for each element of the domain of $\mathcal{I}$. Conventionally, we use 'a', 'b', 'c',… as LMPL individual constants for α, β, γ,… That is, we match the English and Greek letters alphabetically. This is called the *alphabetic convention.* So in the preceding interpretations,

(i) the instances of '(∃x)Fx' and '(∀x)Fx' are 'Fa' and 'Fb';
(ii) the instances of '(∃x)Gx' and '(∀x)Gx' are 'Ga' and 'Gb';
(iii) the instances of '(∀x)(Fx ∨ Gx)' are 'Fa ∨ Ga' and 'Fb ∨ Gb';
(iv) the instances of '(∃x)(Fx & Gx)' are 'Fa & Ga' and 'Fb & Gb'.

Thus the number of instances of a quantified sentence in an interpretation $\mathcal{I}$ is determined by the number of objects in the domain of $\mathcal{I}$.

Let us be clear about the difference between the roman and Greek letters. When we specify the domain of an interpretation, we use the Greek letters as if they are metalanguage (English) names of specific objects—indeed, instead of using Greek letters we could just use actual English names, such as numerals or city names or names of rivers, to specify domains. By contrast, the letters 'a', 'b', 'c' and so on are taken to be names in the object language, LMPL. Unlike 'Socrates', '3' or (we are supposing) 'α', the individual constants of LMPL are uninterpreted: in advance of stipulating some interpretation we cannot say what they stand for, and we can stipulate any reference we like. This difference is a little obscured by the alphabetic convention and the use of artificial names in the metalanguage to specify domains, but these are simply matters of convenience and the reader who wishes to can use numerals to specify domains and assign LMPL names to numbers any way he or she pleases.

We can now state the rules which determine the truth-values of quantified sentences in interpretations. What we are doing here for the quantifiers is analogous to giving truth-tables for the sentential connectives, with two differences. A truth-table for a connective tells us how on an interpretation $\mathcal{I}$ the truth-value of a compound sentence with that connective as main connective is determined by the truth-values on $\mathcal{I}$ of the main subsentences. In predicate logic, an interpretation is not merely an assignment of truth-values to sentence-letters, but also includes a specification of a nonempty domain of discourse and an assignment of extensions to predicates. Moreover, the truth-value of $\ulcorner(\mathrm{Q}\nu)\phi\nu\urcorner$ on an interpretation $\mathcal{I}$ is not determined by the *syntactic* component to which $\ulcorner(\mathrm{Q}\nu)\urcorner$ connects, since that component, $\phi\nu$, does not have a truth-value, being an open sentence. But we can think of the instances as components in a semantic sense, so the dissimilarity with our procedure in sentential logic is not too great (we give a more precise description of the difference in §1 of Chapter 8, using the notion of *full compositionality*).

The semantic rules for quantifiers in predicate logic are these:

(∀⊤) A universal sentence is *true* on an interpretation $\mathcal{I}$ if *all* its instances in $\mathcal{I}$ are true.
(∀⊥) A universal sentence is *false* on an interpretation $\mathcal{I}$ if *at least one* of its instances in $\mathcal{I}$ is false.
(∃⊤) An existential sentence is *true* on an interpretation $\mathcal{I}$ if *at least one* of its instances in $\mathcal{I}$ is true.
(∃⊥) An existential sentence is *false* on an interpretation $\mathcal{I}$ if *all* its instances in $\mathcal{I}$ are false.

Each rule states a sufficient condition for a quantified sentence to have a particular truth-value, but granted the intuitive meaning of ‘all’ and ‘some’, the stated conditions are clearly necessary as well as sufficient. Indeed, in view of the Principle of Bivalence, each pair of rules could be expressed in a single rule: a universal sentence is true if and only if all its instances are true, an existential sentence is true if and only if at least one of its instances is true. However, in constructing and reasoning about interpretations, it is useful to have an explicit, separate formulation of the truth condition and the falsity condition for each kind of quantified sentence. The reader should pay careful attention to the rule ($\exists\bot$). The most common mistake in constructing an interpretation with a view to making an existential sentence false is forgetting that this requires *all* the instances to be false. To see the justification for this requirement, note that the falsity of ‘At least one U.S. president was a spy’ requires that it be false that Bush was a spy, that Reagan was a spy, that Carter was a spy and so on back to Washington.

In principle, we should supplement the four quantifier rules with rules for each of the other syntactic categories of sentence of LMPL: atomic sentences, negations, conjunctions, disjunctions, conditionals and biconditionals. However, the rules for these five kinds of sentence are the same as in sentential logic, where we embodied them in truth-tables. For example, a conjunction is true on an interpretation $\mathcal{I}$ if both of its conjuncts are true on $\mathcal{I}$, and false on $\mathcal{I}$ if at least one conjunct is false on $\mathcal{I}$; a biconditional is true on $\mathcal{I}$ if both of its sides have the same truth-value on $\mathcal{I}$, false if they do not. And if an atomic sentence is a sentence-letter, it is true or false on $\mathcal{I}$ according to whether $\mathcal{I}$ assigns it $\top$ or $\bot$ (remember that every interpretation $\mathcal{I}$ assigns $\bot$ to ‘⋏’). Since there is nothing new in this, we will not bother to recapitulate the principles for the connectives in separate clauses. However, atomic sentences built out of predicates and individual constants do represent something new, so we should make explicit how the truth-value on an interpretation $\mathcal{I}$ of such an atomic sentence relates to what $\mathcal{I}$ says about the extension of the relevant predicate. The rule, which is simple, is stated so as to include the sentential case:

- (A$\top$) An atomic sentence λt is true on an interpretation $\mathcal{I}$ if the object referred to by the individual constant t *belongs* to the extension in $\mathcal{I}$ of λ; a sentence-letter p is true on $\mathcal{I}$ if $\mathcal{I}$ assigns $\top$ to p.
- (A$\bot$) An atomic sentence λt is false on an interpretation $\mathcal{I}$ if the object named by the individual constant t does *not* belong to the extension in $\mathcal{I}$ of λ; a sentence-letter p is false on $\mathcal{I}$ if $\mathcal{I}$ assigns $\bot$ to p; every $\mathcal{I}$ assigns $\bot$ to ‘⋏’.

For example, ‘Socrates is wise’ is true on an interpretation $\mathcal{I}$ if the object named by ‘Socrates’ is in the extension in $\mathcal{I}$ of ‘is wise’; if $\mathcal{I}$ is the real world, this just means that Socrates is one of those who are wise.

We can now use these six rules to give a complete demonstration of the invalidity of arguments A and B. For both arguments, we will use our first interpretation: D = $\{\alpha,\beta\}$, Ext(F) = $\{\alpha\}$, Ext(G) = $\{\beta\}$. We have to show that in each case, the premise of the argument-form is true and the conclusion is false.

Example 1: Show that the LMPL argument-form A is invalid.

A: $(\exists x)Fx \; \& \; (\exists x)Gx$
$\therefore (\exists x)(Fx \; \& \; Gx)$

Interpretation: $D = \{\alpha,\beta\}$, $Ext(F) = \{\alpha\}$, $Ext(G) = \{\beta\}$, or as a matrix, below; 'a' refers to α and 'b' refers to β. *Explanation:* The premise is true because both con-

	F	G
α	+	–
β	–	+

juncts are true. '(∃x)Fx' is true because 'Fa' is true (rule ∃⊤). '(∃x)Gx' is true because 'Gb' is true (rule ∃⊤). 'Fa' is true because 'a' denotes α and α belongs to Ext(F), and 'Gb' is true because 'b' denotes β and β belongs to Ext(G) (rule A⊤). '(∃x)(Fx & Gx)' is false because 'Fa & Ga' is false *and* 'Fb & Gb' is false (rule ∃⊥). 'Fa & Ga' is false because 'Ga' is false, and 'Fb & Gb' is false because 'Fb' is false (we assume knowledge of the truth-table for '&'). Finally, 'Ga' is false because 'a' denotes α and α does not belong to Ext(G) and 'Fb' is false because 'b' denotes β and β does not belong to Ext(F).

Since this is our first example of a demonstration of invalidity for an argument-form in LMPL, we have spelled it out in complete detail. But in future, steps in the explanation which appeal to truth-tables to account for the truth-values of sentential combinations of atomic formulae (e.g., 'Fb & Gb') will be omitted. As abbreviations, we will use the standard mathematical symbols '∈' and '∉' for 'belongs to' and 'does not belong to' respectively. And we will not explicitly state the part of the interpretation that is covered by the alphabetic convention, in this case that 'a' denotes α and 'b' denotes β, unless the individual constant of LMPL in question actually occurs in the argument-form for which an interpretation is being given. If it does so occur, we state its reference in the form ⌜Ref(*t*) = x⌝, which abbreviates ⌜*t* refers to x⌝. And in the subsequent reasoning, we will not cite which of the quantifier rules we are appealing to when we use them. All this enables us to give a more succinct demonstration of B's invalidity.

Example 2: Show that the LMPL argument-form B is invalid.

B: $(\forall x)(Fx \vee Gx)$
$\therefore (\forall x)Fx \vee (\forall x)Gx.$

Interpretation: As in Example 1. *Explanation:* The premise is true because both its instances '(Fa ∨ Ga)' and '(Fb ∨ Gb)' are true, since $\alpha \in Ext(F)$ and $\beta \in Ext(G)$. The conclusion is false because both its disjuncts are: '(∀x)Fx' is false since 'Fb' is false ($\beta \notin Ext(F)$), and '(∀x)Gx' is false since 'Ga' is false ($\alpha \notin Ext(G)$).

The most important points to notice about these examples are how we explain the falsity of the conclusion in A and the truth of the premise in B. In explaining why the conclusion of A, '(∃x)(Fx & Gx)', is false, we cite the falsity of *both* instances. It would not be sufficient to say that '(∃x)(Fx & Gx)' is false because 'Fa & Ga' is false, since that is not sufficient for the falsity of '(∃x)(Fx & Gx)' in a domain which contains other objects (recall the example 'at least one U.S. president was a spy'). Similarly, it would not be sufficient to explain the truth of '(∀x)(Fx ∨ Gx)' to cite merely the truth of one instance: the truth of *all* instances has to be cited.

As noted earlier, in forming the instances of a quantified sentence we use LMPL individual constants 'a', 'b' and so on, whose interpretation we stipulate. In A and B none of these names occur in the arguments themselves, but in §1 of Chapter 5 (page 149) we considered the intuitively invalid English arguments

D: (1) Socrates is wise
(2) ∴ Everyone is wise

and

E: (1) Someone is happy
(2) ∴ Plato is happy.

We now translate these into LMPL and demonstrate their invalidity.

Example 3: Show that the LMPL argument-form D.s is invalid.

D.s: Wa
∴ (∀x)Wx

Interpretation: D = {α,β}, Ext(W) = {α}, Ref(a) = α. *Explanation:* '(∀x)Wx' is false because 'Wb' is false (since $\beta \notin$ Ext(W)), while 'Wa' is true since Ref(a) ∈ Ext(W).

Example 4: Show that the LMPL argument-form E.s is invalid.

E.s (∃x)Hx
∴ Hb

Interpretation: D = {α,β}, Ext(H) = {α}, Ref(b) = β. *Explanation:* 'Hb' is false since $\beta \notin$ Ext(H), while '(∃x)Hx' is true since 'Ha' is true, since $\alpha \in$ Ext(W).

In these symbolizations of the English arguments, the interpreted metalanguage proper names 'Socrates' and 'Plato' are rendered by the uninterpreted LMPL individual constants 'a' and 'b' and then an interpretation is specified for the resulting argument-forms in which 'a' and 'b' denote α and β (whatever *they* are!) rather than Socrates and Plato. This further emphasizes the fact that the validity or invalidity of the original English arguments turns on their forms, not their subject matter.

We can now explicitly state the conception of interpretation and validity underlying the four demonstrations of invalidity just given. Since the object language LMPL includes LSL, the account of interpretation has to include the kind of interpretation appropriate for LSL as well as the kind appropriate for the new apparatus.

> An *interpretation* of an argument-form in LMPL consists in a specification of a nonempty domain of discourse together with an assignment of extensions to the predicate-letters, if any, in the argument-form, of references to the individual constants, if any, in the argument-form, and of truth-values to the sentence-letters, if any, in the argument-form. An extension for a predicate in an interpretation $\mathcal{I}$ is a subset of the domain of $\mathcal{I}$. '$\curlywedge$' is always assigned $\perp$.

It should be noted that since the empty set is a subset of every set, this definition allows us to assign an empty extension to a predicate-letter; in other words, we can have predicate-letters which are true of no objects, just as we can have English predicates which are true of no objects (consider 'is a Martian').[1] A standard symbol for the empty set is '$\varnothing$'.

> An argument-form in LMPL is *valid* if and only if there is no interpretation of it on which all its premises are true and its conclusion is false.
>
> An English argument is *monadically* valid if and only if it translates into a valid argument-form of LMPL.

As in the sentential case, a monadically valid English argument is valid absolutely. A monadically invalid English argument may be absolutely invalid, or it may translate into a valid argument-form of a more powerful kind of logic. D and E have just been shown to be monadically invalid. Although we cannot *prove* that they are invalid absolutely, it seems plausible that they are, since their translations into LMPL appear to capture all relevant aspects of the structure of the English sentences.

The definition of LMPL validity is like the definition of LSL validity; what has changed is just the notion of interpretation in the two definitions. Another definition which carries over from LSL is that of logical equivalence:

> If *p* and *q* are sentences of LMPL, *p* and *q* are said to be *logically equivalent* if and only if, for each interpretation $\mathcal{I}$, the truth-value of *p* on $\mathcal{I}$ is the same as the truth-value of *q* on $\mathcal{I}$.

[1] Why is the empty set a subset of every set? To say that X is a subset of Y is to say that every member of X is a member of Y, or in symbols, '$(\forall x)(x \in X \rightarrow x \in Y)$'. If X is the empty set, then no matter what the domain of discourse, every instance of '$(\forall x)(x \in X \rightarrow x \in Y)$' will be true since the antecedent of every instance will be false. Consequently, no matter what set Y is, if X is empty, X is a subset of Y.

We now wish to apply these techniques to demonstrate particular LMPL arguments to be invalid. But before beginning on this, we need to familiarize ourselves further with the truth and falsity conditions of sentences of LMPL. Here is an interpretation, chosen completely at random, which is rather more complex than the two we have considered so far:

	F	G	H	I	J
α	+	+	–	+	–
β	–	–	–	+	+
γ	+	–	–	–	+

Thus D = $\{\alpha,\beta,\gamma\}$, Ext(F) = $\{\alpha,\gamma\}$, Ext(G) = $\{\alpha\}$, Ext(H) = $\emptyset$, Ext(I) = $\{\alpha,\beta\}$, Ext(J) = $\{\beta,\gamma\}$. The following six sentences are also chosen at random, and we have to determine their truth-values in this interpretation.

(1) ~Ja
(2) Fc → Ic
(3) (∃x)(Jx ↔ Hx)
(4) (∀x)(Jx → (Gx ∨ Fx))
(5) (∃x)Gx → (∀y)(Fy ∨ Gy)
(6) (∃y)(∀x)(Gy & (Jx → (Ix ∨ Fx)))

(1) '~Ja' is true because 'Ja' is false, since $\alpha \notin$ Ext(J).
(2) 'Fc → Ic' is false because $\gamma \in$ Ext(F) and $\gamma \notin$ Ext(I).
(3) '(∃x)(Jx ↔ Hx)' is true because 'Ja ↔ Ha' is true, since $\alpha \notin$ Ext(J) and $\alpha \notin$ Ext(H).
(4) '(∀x)(Jx → (Gx ∨ Fx))' is false because 'Jb → (Gb ∨ Fb)' is false, since $\beta \in$ Ext(J) but $\beta \notin$ Ext(G) and $\beta \notin$ Ext(F).
(5) '(∃x)Gx → (∀y)(Fy ∨ Gy)' is false because '(∃x)Gx' is true and '(∀y)(Fy ∨ Gy)' is false. '(∃x)Gx' is true because 'Ga' is true, since $\alpha \in$ Ext(G). '(∀y)(Fy ∨ Gy)' is false because 'Fb ∨ Gb' is false, since $\beta \notin$ Ext(F) and $\beta \notin$ Ext(G).
(6) '(∃y)(∀x)(Gy & (Jx → (Ix ∨ Fx)))' is true because '(∀x)(Ga & (Jx → (Ix ∨ Fx)))' is true, in turn because '(Ga & (Ja → (Ia ∨ Fa)))', '(Ga & (Jb → (Ib ∨ Fb)))' and '(Ga & (Jc → (Ic ∨ Fc)))' are all true. '(Ga & (Ja → (Ia ∨ Fa)))' is true because $\alpha \in$ Ext(G) and $\alpha \notin$ Ext(J). '(Ga & (Jb → (Ib ∨ Fb)))' is true because $\alpha \in$ Ext(G) and $\beta \in$ Ext(I). And '(Ga & (Jc → (Ic ∨ Fc)))' is true because $\alpha \in$ Ext(G) and $\gamma \in$ Ext(F).

In all our evaluations we follow the syntactic structure of the formula: we begin with the main subformulae and then work down through their main subformulae, and so on until we arrive at atomic formulae. (5) should be noted. (5) is a conditional, *not* a quantified sentence, and it would be a mistake to begin its evaluation by applying a quantifier rule. Rather, since (5) is a conditional, we calculate the truth-values of its antecedent and consequent separately, and then use the truth-table for '→'.

Special attention should be paid to (6), in which two quantifiers prefix the body of the formula. This makes the search process longer. Since (6) is existential, we can find out whether it is true or false by finding out whether or not it has a true instance. The instances of (6) are

(6a) (∀x)(Ga & (Jx → (Ix ∨ Fx)))
(6b) (∀x)(Gb & (Jx → (Ix ∨ Fx)))
(6c) (∀x)(Gc & (Jx → (Ix ∨ Fx))).

Each of these instances is a universal sentence, and each of them in turn has three instances. For example, the instances of (6b) are:

($6b_1$) Gb & (Ja → (Ia ∨ Fa))
($6b_2$) Gb & (Jb → (Ib ∨ Fb))
($6b_3$) Gb & (Jc → (Ic ∨ Fc))

Had (6) been false, therefore, we would have had to consider a total of nine quantifier-free sentences to confirm this. Fortunately, (6) is true, and this is shown by its instance (6a).[2]

❑ Exercise

Evaluate the numbered formulae in the displayed interpretation. Explain your reasoning in the same way as in (1)–(6) above, accounting for the truth-values of quantified sentences in terms of the truth-values of their instances.

	F	G	H	I	J
α	+	–	+	–	+
β	+	–	+	–	–
γ	+	–	–	+	+

(1) (Ha ∨ Hc) → Ib
(2) (Ha & Hc) ∨ (Ja & Jc)
*(3) (∃x)(Fx & Gx)
(4) ~(∃x)Gx
(5) (∃x)(Ix → Hx)
(6) (∀x)((Hx ∨ Ix) → Fx)
(7) (∀x)((Fx & Hx) → Jx)
*(8) (∀x)(Hx → (∃y)(Jx & Iy))
(9) (∀x)(∃y)(Fx → (Hx ∨ Jy))
(10) (∃x)Ix → (∀x)(Jx → Ix)
(11) (∃x)(Ix → (∀y)(Jy → Iy))
(12) (∀x)(∀y)((Fx ↔ Gy) ↔ (∃w)(∃z)(Hw & Jz))

[2] The rules of this section explain the term 'logical constant' mentioned earlier, which is applied both to quantifiers and to sentential connectives. Domains and extensions of predicates *vary* from interpretation to interpretation, but the evaluation rules for connectives and quantifiers are *constant* across all interpretations. Any expression which has a constant evaluation rule is called a logical constant.

2 Constructing counterexamples

Now that we understand how sentences of LMPL are evaluated in interpretations, we turn to the question of how to find interpretations which show invalid LMPL argument-forms to be invalid. An interpretation which shows an LMPL argument-form to be invalid is called a *counterexample* to the argument-form. The question, then, is how to go about constructing counterexamples. We will illustrate the techniques in connection with a number of examples. But first we reintroduce the double-turnstile notation, which we are going to use to express semantic consequence exactly as we did for LSL in §4 of Chapter 3:

- For any sentences $p_1,\ldots,p_n$ and q of LMPL, we write $p_1,\ldots,p_n \vDash q$ to mean that q is a semantic consequence of $p_1,\ldots,p_n$, that is, that no interpretation of $p_1,\ldots,p_n$ and q makes all of $p_1,\ldots,p_n$ true and q false.
- For any sentences $p_1,\ldots,p_n$ and q of LMPL, we write $p_1,\ldots,p_n \nvDash q$ to mean that q is *not* a semantic consequence of $p_1,\ldots,p_n$, that is, that *some* interpretation of $p_1,\ldots,p_n$ and q makes all of $p_1,\ldots,p_n$ true and q false.
- For any sentence q of LMPL, we write $\vDash q$ to mean that there is no interpretation that makes q false, or in other words, that every interpretation makes q true. Such a q is said to be *logically true*. An example: $\vDash (\forall x)((Fx \,\&\, Gx) \rightarrow Fx)$.

Logical truth is the special case $n = 0$ of semantic consequence, in that a logical truth is a semantic consequence of the empty set of premises.

To give a counterexample to an argument-form with premises $p_1,\ldots,p_n$ and conclusion q is to show that $p_1,\ldots,p_n \nvDash q$, and this will be our preferred way of expressing our goal.

Example 1: Show $(\forall x)(Fx \rightarrow Gx), (\forall x)(Fx \rightarrow Hx) \nvDash (\forall x)(Gx \rightarrow Hx)$.

Provisionally, we begin by setting up a domain $D = \{\alpha\}$. No matter what the problem, this can always be the first step, since every interpretation must have a nonempty domain. We say that the specification of D at this stage is provisional because in the course of making the conclusion false and the premises true, it may be necessary to add further objects to the domain.

When the conclusion formula is a universal sentence, it is clear what we must do: we must arrange that some object in the domain provides a false instance of the universal sentence. Such an object is also known as a counterexample—not a counterexample to the argument-form but to the universal sentence. In this particular case, what is required is an object in the extension of ‘G’ which is not in the extension of ‘H’. So we set $\text{Ext(G)} = \{\alpha\}$, $\text{Ext(H)} = \varnothing$; again, this is merely provisional, since it may later be necessary to add to the extensions of ‘G’ and ‘H’. Our interpretation now makes the conclusion false, but in order to evaluate the premises in it we have to specify an extension for ‘F’. To make both premises true, what we must avoid is having an object in the extension of 'F' which is not in the extension of 'G' or not in the extension of 'H'. In the current setup, the simplest way of avoiding such an object is to let the extension of 'F' be empty. So we do not need to add to the domain or to the extensions of the other two predicates. The interpretation we arrive at is: $D = \{\alpha\}$, $Ext(F) = \emptyset$, $Ext(G) = \{\alpha\}$ and $Ext(H) = \emptyset$, with the matrix displayed below. To

	F	G	H
α	–	+	–

complete the solution to the problem, we explain why this interpretation is a counterexample. '$(\forall x)(Gx \rightarrow Hx)$' is false because '$Ga \rightarrow Ha$' is false, since $\alpha \in Ext(G)$ and $\alpha \notin Ext(H)$. '$(\forall x)(Fx \rightarrow Gx)$' and '$(\forall x)(Fx \rightarrow Hx)$' are both true because '$Fa \rightarrow Ga$' and '$Fa \rightarrow Ha$' are both true, since $\alpha \notin Ext(F)$.

Example 2: Show $(\exists x)(Fx \,\&\, Gx), (\exists x)(Fx \,\&\, Hx), (\forall x)(Gx \rightarrow \sim Hx) \nvDash (\forall x)(Fx \leftrightarrow (Gx \vee Hx))$.

We begin as before with $\{\alpha\}$ as provisional domain. A counterexample to '$(\forall x)(Fx \leftrightarrow (Gx \vee Hx))$' requires either (i) an object in Ext(F) which is in neither Ext(G) nor Ext(H), or else (ii) an object in at least one of Ext(G) and Ext(H) which is not in Ext(F). Since premise 1 will require an object in Ext(F) anyway, we start with (i) (if we can find no way of making all the premises true, we will have to come back to (ii)). Provisionally, then, we set $Ext(F) = \{\alpha\}$ and put nothing into the extensions of 'G' and 'H'; this gives us the false conclusion-instance '$Fa \leftrightarrow (Ga \vee Ha)$'. Turning to the premises, we see that to make all three true we have to set up Ext(F), Ext(G) and Ext(H) so that true instances of the first two premises are provided and at the same time no object is in both Ext(G) and Ext(H) (otherwise premise 3 would be false). We could obtain a true instance of premise 1 by adding α to Ext(G), but this would defeat what we have already done to ensure the falsity of the conclusion. Consequently, we have to add a second object to the domain to provide a true instance of premise 1. Thus we now put $D = \{\alpha,\beta\}$, $Ext(F) = \{\alpha,\beta\}$, $Ext(G) = \{\beta\}$ (making '$Fb \,\&\, Gb$' true) and add nothing to the extension of 'H', as in the matrix below. This means that we do

	F	G	H
α	+	–	–
β	+	+	–

not yet have a true instance for premise 2. If we set $Ext(H) = \{\alpha\}$ we have a true instance for premise 2, but we will have made '$Fa \leftrightarrow (Ga \vee Ha)$' true, and as '$Fb \leftrightarrow (Gb \vee Hb)$' is also true, we would have made the conclusion true. And if we set $Ext(H) = \{\beta\}$, we have an object in both Ext(G) and Ext(H), which is exactly

	F	G	H
α	+	–	–
β	+	+	–
γ	+	–	+

what we must avoid in order to verify premise 3. Consequently, we must add a third object to D to provide a true instance for premise 2 that does not refute premise 3. The interpretation at which we arrive, therefore, is D = $\{\alpha,\beta,\gamma\}$, Ext(F) = $\{\alpha,\beta,\gamma\}$, Ext(G) = $\{\beta\}$, Ext(H) = $\{\gamma\}$, displayed above, and we confirm that this is right with the following explanation: '(∀x)(Fx ↔ (Gx ∨ Hx))' is false because 'Fa ↔ (Ga ∨ Ha)' is false, since $\alpha \in$ Ext(F), $\alpha \notin$ Ext(G) and $\alpha \notin$ Ext(H). '(∃x)(Fx & Gx)' is true because 'Fb & Gb' is true, since $\beta \in$ Ext(F) and $\beta \in$ Ext(G). '(∃x)(Fx & Hx)' is true because 'Fc & Hc' is true, since $\gamma \in$ Ext(F) and $\gamma \in$ Ext(H). '(∀x)(Gx → ~Hx)' is true because all three of 'Ga → ~Ha', 'Gb → ~Hb' and 'Gc → ~Hc' are true. First, 'Ga → ~Ha' is true because $\alpha \notin$ Ext(G); next, 'Gb → ~Hb' is true because $\beta \in$ Ext(G) and $\beta \notin$ Ext(H); last, 'Gc → ~Hc' is true because $\gamma \notin$ Ext(G).

Example 2 illustrates the usual reason for increasing the size of the domain: a number of existential premises require different objects to provide true instances, since using the same object for all those premises would make another premise false or the conclusion true.

The next example illustrates how we handle arguments which contain sentence-letters as well as predicates and quantifiers.

Example 3: Show (∃x)(Fx → A) $\nvDash$ (∃x)Fx → A.

It is important to note the different forms of the premise and conclusion: the premise is an existential sentence in which '→' is within the scope of '∃', while the conclusion is a conditional in which '∃' is within the scope of '→'. Since the conclusion has antecedent '(∃x)Fx' and consequent 'A', we require an interpretation in which '(∃x)Fx' is true and 'A' is false. However, the interpretation $\mathcal{I}$ with D = $\{\alpha\}$, Ext(F) = $\{\alpha\}$, and ⊥ assigned to 'A' also makes the premise false, since there is only the instance 'Fa → A'. To make the premise true, we have to provide another instance, which means adding an object to the domain. But since 'A' is false and we want the new object to provide a true instance of the premise, we should *not* add the new object to the extension of 'F'. So the interpretation at which we arrive is: D = $\{\alpha,\beta\}$, Ext(F) = $\{\alpha\}$, ⊥ assigned to 'A', as displayed below. *Explanation:* '(∃x)Fx → A' is false because 'A' is false and '(∃x)Fx'

	F
α	+
β	–

'A' is false

is true; ‘$(\exists x)Fx$’ is true because $\alpha \in \text{Ext}(F)$. And ‘$(\exists x)(Fx \rightarrow A)$’ is true because ‘$Fb \rightarrow A$’ is true, since $\beta \notin \text{Ext}(F)$.

Finally we give an example involving successive quantifiers.

Example 4: Show $(\exists x)(\forall y)(Fx \rightarrow Gy) \nvDash (\forall x)(\exists y)(Fx \rightarrow Gy)$.

To make the conclusion false, we must have at least one of its instances false. On any interpretation, the instances of ‘$(\forall x)(\exists y)(Fx \rightarrow Gy)$’ are existential sentences of the form ‘$(\exists y)(F\_ \rightarrow Gy)$’, where the blank is filled by an individual constant. One of these instances must be false for ‘$(\forall x)(\exists y)(Fx \rightarrow Gy)$’ to be false. Since our domain will contain α anyway, we may as well begin by making ‘$(\exists y)(Fa \rightarrow Gy)$’ false. This means, by $(\exists\bot)$, that all *its* instances have to be false, so in particular ‘$(Fa \rightarrow Ga)$’ must be false. Hence $\alpha \in \text{Ext}(F)$, $\alpha \notin \text{Ext}(G)$. This gives us the interpretation immediately below. However, as things stand in this

	F	G
α	+	–

interpretation, the premise of Example 4 is false. The premise is an existential sentence, and so only requires one true instance for it to be true itself. But with just α in the domain, the premise has only one instance, ‘$(\forall y)(Fa \rightarrow Gy)$’, and this universal sentence is false because it has the false instance ‘$(Fa \rightarrow Ga)$’. Since we do not want to alter any entries we have already made in the interpretation (they were required to make the conclusion false) it follows that to make the premise ‘$(\exists x)(\forall y)(Fx \rightarrow Gy)$’ true, we should provide it with another instance ‘$(\forall y)(Fb \rightarrow Gy)$’ and ensure that this instance is true.

There are now two constraints to satisfy simultaneously: we have to make ‘$(\forall y)(Fb \rightarrow Gy)$’ true, which means making both its instances ‘$Fb \rightarrow Ga$’ and ‘$Fb \rightarrow Gb$’ true, and at the same time we have to avoid doing anything that would make the conclusion true. Since ‘Ga’ is false, our only option for making ‘$Fb \rightarrow Ga$’ true is to make ‘Fb’ false as well. Consequently, we set $\beta \notin \text{Ext}(F)$, and then ‘$Fb \rightarrow Ga$’ and ‘$Fb \rightarrow Gb$’ are both true, so ‘$(\forall y)(Fb \rightarrow Gy)$’ is true as desired, which in turn makes the premise ‘$(\exists x)(\forall y)(Fx \rightarrow Gy)$’ true. As for the conclusion, the important thing is to keep its previously false instance ‘$(\exists y)(Fa \rightarrow Gy)$’ still false. The new instance of this existential sentence is ‘$Fa \rightarrow Gb$’, and since ‘Fa’ is true, we have to set $\beta \notin \text{Ext}(G)$ to make this conditional false. So the final interpretation is $D = \{\alpha,\beta\}$, $\text{Ext}(F) = \{\alpha\}$, $\text{Ext}(G) = \emptyset$, as exhibited below:

	F	G
α	+	–
β	–	–

To summarize: ‘$(\exists x)(\forall y)(Fx \rightarrow Gy)$’ has two instances, (i) ‘$(\forall y)(Fa \rightarrow Gy)$’ and (ii) ‘$(\forall y)(Fb \rightarrow Gy)$’, and it is true because (ii) is true. (ii) is true because it has two instances, ‘$Fb \rightarrow Ga$’ and ‘$Fb \rightarrow Gb$’ and both are true since both have false antecedents. On the other hand, ‘$(\forall x)(\exists y)(Fx \rightarrow Gy)$’ is false. It has two instances, (iii) ‘$(\exists y)(Fa \rightarrow Gy)$’ and (iv) ‘$(\exists y)(Fb \rightarrow Gy)$’, and (iii) is false. (iii) is false because both its instances, ‘$(Fa \rightarrow Ga)$’ and ‘$Fa \rightarrow Gb$’, are false, since both have true antecedent and false consequent.

It is noticeable that all our problems of showing failure of semantic consequence have been solved with small domains, whereas in Chapter 5, the domains with respect to which our symbolizations are relativized are large: people, places, things. But counterexamples with small domains to argument-forms derived from symbolizations of English relativized to large domains are not irrelevant to English arguments, for if the argument-form can be shown to be invalid by an interpretation with a small domain, then it *is* shown to be invalid, and if it is the form of an English argument, it follows that that English argument is monadically invalid. Moreover, a counterexample with a small domain can be ‘blown up’ into one with a large domain by a duplication process (see Exercise II.2), so our preference for simplicity does not entail irrelevance.

❑ Exercises

I Show the following, with explanations:

(1) $(\forall x)(Fx \rightarrow Gx) \nvDash (\forall x)(Gx \rightarrow Fx)$
(2) $(\forall x)(Fx \vee Gx), (\forall x)(Fx \vee Hx) \nvDash (\forall x)(Gx \vee Hx)$
(3) $(\forall x)(Fx \rightarrow \sim Gx), (\forall x)(Gx \rightarrow Hx) \nvDash (\forall x)(Fx \rightarrow \sim Hx)$
*(4) $(\forall x)((Fx \;\&\; Gx) \rightarrow Hx) \nvDash (\forall x)(Fx \vee Gx) \vee (\forall x)(Fx \vee Hx)$
(5) $(\exists x)(Fx \;\&\; \sim Hx), (\exists x)(Gx \;\&\; \sim Hx) \nvDash (\exists x)(Fx \;\&\; Gx)$
(6) $(\exists x)(Fx \leftrightarrow Gx) \nvDash (\exists x)(Fx \vee Gx)$
(7) $(\exists x)(Fx \;\&\; Gx), (\forall x)(Gx \rightarrow Hx) \nvDash (\forall x)(Fx \rightarrow Hx)$
(8) $(\forall x)Fx \rightarrow (\exists x)Gx \nvDash (\forall x)(Fx \rightarrow Gx)$
(9) $(\exists x)(Fx \vee Gx), (\forall x)(Fx \rightarrow \sim Hx), (\exists x)Hx \nvDash (\exists x)Gx$
(10) $(\forall x)(Fx \rightarrow Gx) \nvDash \sim(\forall x)(Fx \rightarrow \sim Gx)$
(11) $(\exists x)\sim Fx \nvDash \sim(\exists x)Fx$
(12) $\sim(\forall x)Fx \nvDash (\forall x)\sim Fx$
*(13) $(\forall x)(Fx \rightarrow Gx) \rightarrow (\forall x)(Hx \rightarrow Jx) \nvDash (\exists x)(Fx \;\&\; Gx) \rightarrow (\forall x)(Hx \rightarrow Jx)$
(14) $(\exists x)(Fx \rightarrow A), (\exists x)(A \rightarrow Fx) \nvDash (\forall x)(A \leftrightarrow Fx)$
(15) $\sim(A \rightarrow (\forall x)Fx) \nvDash (\forall x)(A \rightarrow \sim Fx)$
(16) $(\forall x)Fx \leftrightarrow A \nvDash (\forall x)(Fx \leftrightarrow A)$
(17) $(\forall x)Fx \rightarrow (\forall x)Gx \nvDash Fa \rightarrow (\forall x)Gx$
(18) $Fa \rightarrow (\exists x)Gx \nvDash (\exists x)Fx \rightarrow (\exists x)Gx$
(19) $(\forall x)Fx \leftrightarrow (\forall x)Gx \nvDash (\exists x)(Fx \leftrightarrow Gx)$
*(20) $(\forall x)Fx \rightarrow (\exists y)Gy \nvDash (\forall x)(Fx \rightarrow (\exists y)Gy)$
(21) $(\exists x)(Fx \rightarrow (\forall y)Gy) \nvDash (\exists x)Fx \rightarrow (\forall y)Gy$
(22) $\sim(\exists x)Fx \vee \sim(\exists x)Gx \nvDash \sim(\exists x)(Fx \vee Gx)$

(23) $(\exists x)(Fx \leftrightarrow Gx), (\forall x)(Gx \rightarrow (Hx \rightarrow Jx)) \nvDash (\exists x)Jx \vee \sim(\exists x)Fx$
(24) $(\forall x)(\exists y)(Fy \rightarrow Gx) \nvDash (\forall x)(\exists y)(Gy \rightarrow Fx)$
(25) $(\exists x)(Fx \rightarrow (\exists y)Gy) \nvDash (\exists x)(\forall y)(Fx \rightarrow Gy)$
(26) $(\forall x)(\exists y)(Fx \rightarrow Gy) \nvDash (\exists x)(\forall y)(Fx \rightarrow Gy)$
*(27) $(\forall x)(\exists y)(Gy \rightarrow Fx) \nvDash (\forall x)[(\exists y)Gy \rightarrow Fx]$
(28) $(\forall x)[(\forall y)Gy \rightarrow Fx] \nvDash (\forall x)(\forall y)(Gy \rightarrow Fx)$
(29) $(\forall x)(Fx \rightarrow (\exists y)Gy) \nvDash (\forall x)(\forall y)(Fx \rightarrow Gy)$
(30) $(\exists x)(\forall y)(Fx \rightarrow Gy) \nvDash (\exists y)(\forall x)(Fx \rightarrow Gy)$

II Show $(\exists x)(Fx \& Gx) \& (\exists x)(Fx \& \sim Gx) \& (\exists x)(\sim Fx \& Gx) \nvDash (\forall x)(Fx \vee Gx)$. Then evaluate the following two statements as true or false. Explain your answer.

(1) If a sentence is true on at least one interpretation whose domain has n members ($n \geqslant 2$), it is true on at least one interpretation whose domain has $n - 1$ members.
*(2) If a sentence is true on at least one interpretation whose domain has n members ($n \geqslant 2$), it is true on at least one interpretation whose domain has $n + 1$ members. (Hint: think of how you could define the notion of two objects being *indistinguishable* in an interpretation.)

3 Deductive consequence: quantifiers in NK

It is perfectly natural to respond to an English argument by saying that the conclusion does not follow from the premises and in support of this to describe a possible situation in which the premises of the argument would be true and the conclusion false. Our abstract model of this procedure, as described in the previous section, is therefore quite realistic. However, we do not usually *advance* an argument by stating our premises and conclusion and defying an opponent to describe a situation in which the premises would be true and the conclusion false (the strategy is not totally unnatural: it is embodied in the rhetorical question 'How could it *fail* to follow?'). In advancing an argument it is much more common to try to reason from the premises to the conclusion. So we now wish to extend our formal model of this procedure, the natural-deduction system NK, to those arguments which involve quantification and predication. Actual arguments in natural language are taking place in an interpreted language, of course. But the reasoning principles involved in giving an argument are independent of interpretation and can be stated for an uninterpreted formal language like LMPL.

The new logical constants are the two quantifiers, so at the very least we shall have to add two I and two E rules for these symbols to NK. However, we shall continue to call the system 'NK' rather than give it a new name to indicate the presence of new rules; whenever we want to contrast NK before and after the new rules are added we will speak of sentential NK versus quantificational NK. This section introduces quantificational NK through three of the four new