CHAPTER TWO: MATHESE

1 Introduction

Mathematicians (well, English-speaking ones, anyway) talk and write about things logical and mathematical (including set theory and anything they construct inside it) in a mixture of ordinary colloquial English and a special purpose dialect of English, which we will refer to as **Mathese**. Mathese is intended to avoid the ambiguity, vagueness, and imprecision of much ordinary colloquial English. It is a good idea to get into the habit of judiciously using Mathese when writing about formally rigorous linguistic theory for an audience with a reasonable degree of mathematical sophistication; e.g. when writing up problem sets for this course. (Alert: it is every bit as important *not* to write this way for a general linguistic audience!) Of course, unless you have an unusually strong mathematical background, it takes some time to get the hang of Mathese, so we will not require immediate mastery; and of course it's also okay to use ordinary English as long as the meaning is completely clear.

In its most basic form, all Mathese has is a few "logicky" expressions and some basic predicates for talking about set membership and equality. Fortunately, it's permissible to add new predicates and names to the language as needed, as long as you take care to *define* them in terms of expressions that are already in the language, as will be explained below. (Without such abbreviations, Mathese quickly becomes opaque to the point of sheer incomprehensibility.) There are also symbols for abbreviating expressions, which are mostly used in displayed calculations and inside of set descriptions; the abbreviations (especially the logicky ones) are usually *not* used in writing Mathese prose (which is what you will usually be writing proofs in).[1]

2 "Logicky" expressions

2.1 Variables

These are upper- or lower-case roman letters (usually italicized in typing), with or without numerical subscripts, used roughly as pronouns or as names of arbitrary sets, e.g. x, y, x_0, x_1, X, Y, etc.

[1] Later on, we'll introduce some formal languages, called *first-order languages*, which consist *entirely* of such symbols. By then, you'll have a good intutitive feeling for what such symbols mean. If you've taken a basic course in predicate logic, you'll already be familar with these.

2.2 And

Mathese 'and' is abbreviated using the **conjunction** symbol $\wedge$. It is used mainly for combining sentences, as in:

S_1 and S_2. (Abbreviated form: $S_1 \wedge S_2$)

A sentence formed this way is called a **conjunctive** sentence. Here S_1 is called the **first conjunct** and S_2 is called the **second conjunct**. A conjunctive sentence is considered to be true if both conjuncts are true; otherwise it is false.

2.3 Or

Mathese 'or' is abbreviated using the **disjunction** symbol $\vee$. Like 'and', it is used mainly for combining sentences, as in:

S_1 or S_2. (Abbreviated form: $S_1 \vee S_2$)

A sentence formed this way is called a **disjunctive** sentence. Here S_1 is called the **first disjunct** and S_2 is called the **second disjunct**. Mathese *or* is **inclusive** disjunction, so that a disjunctive sentence is true if *either or both* of the disjuncts are true, and it is false otherwise.

2.4 Implies

Mathese 'implies' is abbreviated using one of the two **implication** symbols $\rightarrow$ or $\supset$. A synonym for 'implies' is 'if ... then ...'. It too is used for combining sentences, as in:

S_1 implies S_2. (Abbreviated forms: $S_1 \rightarrow S_2$ or $S_1 \supset S_2$)

A sentence formed this way is called a **conditional** or **implicative** sentence. Here S_1 is called the **antecedent** and S_2 is called the **consequent**. Caution: this does not mean quite exactly the same thing as *if* S_1 *then* S_2 in ordinary English. One difference is that a conditional Mathese sentence is considered to be true if the consequent is true, no matter whether the antecedent is true or false and even if the antecedent and the consequent seem to have nothing to do with each other, e.g.

If there does not exist a set with no members, then $0 = 0$.

is true. Another difference is that a conditional Mathese sentence is considered to be true if the antecedent is false, no matter whether the consequent is true or false, e.g.

If $0 \neq 0$ then $1 \neq 1$.

is true!

2.5 If and only if

Mathese 'if and only if', usually written simply as 'iff', is abbreviated using the **biimplication** symbol $\leftrightarrow$. It is used to combine sentences as in:

S_1 iff S_2. (Abbreviated form: $S_1 \leftrightarrow S_2$)

A sentence of this form is called a **biconditional**. S_1 iff S_2 can be thought of as shorthand for:

S_1 implies S_2, and S_2 implies S_1.

Consequently, a sentence of this form is considered to be true if either (1) both S_1 and S_2 are true, or (2) both S_1 and S_2 are false. Otherwise, it is false.

2.6 It is not the case that

Mathese 'it is not the case that' is abbreviated using one of the two **negation** symbols $\neg$ or $\sim$. It is placed before a sentence in order to negate it, as in:

It is not the case that S. (Abbreviated forms: $\neg$S or $\sim$ S)

A sentence of this form is called a **negative** sentence. Here S is called the **scope** of the negation. Unsurprisingly, a negative sentence is considered to be true if the scope is false, and false if the scope is true. For any sentence S, the sentence *it is not the case that* S is called the **negation** of S, or, equivalently, the **denial** of S.

Note that often, the effect of negation with *it is not the case that* can be achieved by ordinary English **verb negation**, which (simplifying slightly) involves replacing the finite verb (the one that agrees with the subject) V with 'does not V' if V is not an auxiliary verb (such as *has* or *is*), or negating V with a following *not* or *-n't* if it *is* an auxiliary. Thus, for example, these pairs of sentences are equivalent (express the same thing):

It is not the case that 2 belongs to 1.
2 does not belong to 1.

It is not the case that 1 is empty.
1 isn't empty.

But negation by *it is not the case that* and verb negation cannot be counted on to produce equivalent effects if the verb is in the scope of a quantifier (see following two sections). For example, these are not equivalent:

It is not the case that for every x, x belongs to x.
For every x, x doesn't belong to x.

For the first is clearly true (for example, 0 doesn't belong to 0), but the truth of the second cannot be determined on the basis of the assumptions in Chapter 1, and in fact different ways of adding further set-theoretic assumptions resolve the issue in different ways.

Note that for predicates with an abbreviatory symbol, such as *equals* ($=$) and *belongs to* ($\in$), the effect of verb negation is accomplished by a diagonal slash, e.g. $\neq$ 'is not equal to', $\notin$ 'is not a member of'.

2.7 For all

Mathese 'for all', abbreviated by the **universal quantifier** symbol $\forall$, forms a sentence by combining first with a variable and then with a sentence, as in:

For all x, S (abbreviated form: $\forall x$S).

The variable x is said to be **bound** by the quantifier, and the sentence S is called the **scope** of the quantifier. Synonyms of 'for all' include 'for each', 'for every', and 'for any'. Usually the bound variable also occurs in the scope; if it doesn't, then the quantification is said to be **vacuous**.

A sentence formed in this way is said to be **universally quantified**, or simply **universal**.

As long as we are using Mathese only to talk about set theory, we can assume that the bound variable in a universal sentence ranges over all sets, that is, 'for all x' is implicitly understood as 'for all sets x'.

However, often we want to universally quantify not over *every* set, but just over the sets that satisfy some condition on x, $\mathrm{S}_1[x]$. Then we say:

For every x with $\mathrm{S}_1[x]$, $\mathrm{S}_2[x]$.

This is understood to be shorthand for

For every x, $\mathrm{S}_1[x]$ implies $\mathrm{S}_2[x]$. (Abbreviated form: $\forall x(\mathrm{S}_1[x] \rightarrow \mathrm{S}_2[x])$)

If such a sentence is true, then we say that $\mathrm{S}_1[x]$ is a **sufficient condition** for $\mathrm{S}_2[x]$, or, equivalently, that $\mathrm{S}_2[x]$ is a **necessary condition** for $\mathrm{S}_1[x]$. A special case of this is that a sentence of this format is true if, no matter what x is, $\mathrm{S}_1[x]$ is false. Such a sentence is said to be **vacuously true**. For example, the sentence

For every x with $x \neq x$, $x = 2$.

is (vacuously) true.

If a universal sentence of the form

For every x, $\mathrm{S}_1[x]$ iff $\mathrm{S}_2[x]$

(i.e. whose scope is a biconditional) is true, then we say $S_1[x]$ is a **necessary and sufficient condition** for $S_2[x]$.

2.8 There exists ... such that

Mathese 'there exists ... such that', abbreviated by the **existential quantifier** symbol $\exists$, forms a sentence by combining first with a variable and then with a sentence, as in:

There exists x such that S (abbreviated form: $\exists x$S).

The variable x is said to be **bound** by the quantifier, and the sentence S is called the **scope** of the quantifier. Synonyms of 'there exists ... such that' include 'for some' and 'there is a(n) ... such that'. Usually the bound variable also occurs in the scope; if it doesn't, then the quantification is said to be **vacuous**.

A sentence formed in this way is said to be **existentially quantified**, or simply **existential**.

As long as we are using Mathese only to talk about set theory, we can assume that the bound variable in an existential sentence ranges over all sets, that is, 'there exists x' is implicitly understood as 'there exists a set x'.

However, often we want to existentially quantify not over *every* set, but just over the sets that satisfy some condition $S_1[x]$. Then we say:

There exists x with $S_1[x]$, such that $S_2[x]$.

This is understood to be shorthand for

There exists x such that $S_1[x]$ and $S_2[x]$.
(Abbreviated form: $\exists x(S_1[x] \wedge S_2[x])$)

Note here the use of parentheses for disambiguation. Without the parentheses, it would be hard to be sure whether the scope of the quantifier was the conjunctive sentence or just its first conjunct. This is a common device in Mathese. Both round and square parentheses can be used, and multiple sets of parentheses can be used in the same sentence.

If a sentence contains variables which are not bound by any quantifier, those variables are called **free**. A sentence is called **closed** if it has no free variables, and **open** otherwise. A sentence whose free variables are $x_0, \ldots, x_n$ is often called a **condition** on $x_0, \ldots, x_n$. The number of free variables in a condition is called its **arity**. Thus conditions might be nullary (no free variables, i.e. a closed sentence), unary (one free variable), binary (two free variables, ternary (three free variables), etc.

2.9 There exists unique ... such that

In Mathese, 'there exists unique ... such that' (abbreviated form: $\exists!x$) combines first with a variable, then with a sentence, as in:

There exists unique x such that S. (Abbreviated form: $\exists!x$ S)

This is understood to be shorthand for:

$\exists x(\mathrm{S}[x] \wedge \forall y(\mathrm{S}[y] \rightarrow y = x))$

3 Defining Predicates

At the outset, the only predicates in Mathese are *equals* (abbreviated $=$) or synonyms such as *is the same as* or *is identical to*, and *in* (abbreviated $\in$) or synonyms such as *is a member of*, *belongs to*, *is an element*, or *is contained in*. But we can *define* new predicates in terms of these and other predicates which have already been defined. The **arity** of a defined predicate is the smallest arity condition that could be used to define it. For example, we define "x is **empty**" to mean $\forall y(y \notin x)$, and "x is a **singleton**" to mean $\exists!y(y \in x)$; these are unary predicates. In "x **is a subset** of y" (abbreviation: $x \subseteq y$), $\subseteq$ is a binary predicate defined by the condition $\forall z(z \in x \rightarrow z \in y)$.

4 Defining Names

If we can prove (i.e. provide a persuasive valid argument based only on our assumptions about set theory and other things that have already been proved) that there exists a unique set x such that $\mathrm{S}[x]$, where $\mathrm{S}[x]$ is some condition on x, then we permit ourselves to bestow a name on that set. For example, it is easy to show that there is a unique set x such that x is empty. (The existence part of the proof is by the Empty Set assumption, and the uniqueness part of the proof is an application of Extensionality.) In this case, as we saw in Chapter One, the set in question is named $\emptyset$ (read "the empty set").

5 Defining Functional Names

Often we can show that for any set y, there exists a unique set x satisfying some condition $\mathrm{S}[x, y]$. In such cases, we permit ourselves to introduce a **functional name**, which is basically a scheme which, for each y, provides

a name for the unique set x such that $\mathrm{S}[x, y]$. To make an analogy with real life: obviously everybody has a mother, so we can use the functional name y**'s mom** to refer to the unique individual x such that x is a mother of y, no matter who y is. Returning to sets, it is easy to prove that for any set y, there is a unique set x such that y is the only member of x. This justifies introducing the functional name $\mathsf{singleton}(y)$, abbreviated $\{y\}$. Likewise, we introduce the functional name $\mathsf{successor}(y)$, abbreviated $s(y)$ which, for each set y, names the unique set x that satisfies the binary condition $x = y \cup \{y\}$.

This naming convention extends to names that depend on more than one variable. Again, to take a real-life example, we might introduce the functional name x**'s seniority over** y: for any two individuals x and y this is defined to be the number of days (rounded off) from x's birthdate to y's birthdate (this is a negative integer if y's birthdate precedes x's). The general principle is that if, for some positive natural number n and some $(n+1)$-ary condition $\mathrm{S}[x_0, \ldots x_n]$ we can prove

$$\forall x_1 \ldots \forall x_n \exists! x_0 \mathrm{S}[x_0, \ldots x_n]$$

then we are allowed to make up a functional name $\mathsf{name}(x_1, \ldots, x_n)$ which for each choice of values for the n variables $x_1, \ldots, x_n$ provides a name for the unique set which satisfies the condition for that choice of values.