ELEMENTARY RECURSION THEORY AND ITS APPLICATIONS TO FORMAL SYSTEMS

By Professor Saul Kripke

Department of Philosophy, Princeton University

Notes by Mario Gómez-Torrente,
revising and expanding notes by John Barker

CONTENTS

Lecture I

First Order Languages

In a first order language L, all the primitive symbols are among the following:

Connectives: $\sim$, $\supset$.

Parentheses: (,).

Variables: $x_1, x_2, x_3, \ldots$.

Constants: $a_1, a_2, a_3, \ldots$.

Function letters:	$f^1_1, f^1_2, \ldots$	(one-place);
	$f^2_1, f^2_2, \ldots$	(two-place);
	:	
	:	
Predicate letters:	$P^1_1, P^1_2, \ldots$	(one-place);
	$P^2_1, P^2_2, \ldots$	(two-place);
	:	
	:	

Moreover, we place the following constraints on the set of primitive symbols of a first order language L. L must contain *all* of the variables, as well as the connectives and parentheses. The constants of L form an initial segment of $a_1, a_2, a_3, \ldots$, i.e., either L contains all the constants, or it contains all and only the constants $a_1, \ldots, a_n$ for some n, or L contains no constants. Similarly, for any n, the n-place predicate letters of L form an initial segment of $P^n_1, P^n_2, \ldots$ and the n-place function letters form an initial segment of $f^n_1, f^n_2, \ldots$. However, we require that L contain at least one predicate letter; otherwise, there would be no formulae of L.

(We could have relaxed these constraints, allowing, for example, the constants of a language L to be $a_1, a_3, a_5, \ldots$. However, doing so would not have increased the expressive power of first order languages, since by renumbering the constants and predicates of L, we could rewrite each formula of L as a formula of some language L' that meets our constraints. Moreover, it will be convenient later to have these constraints.)

A first order language L is determined by a set of primitive symbols (included in the set described above) together with definitions of the notions of a term of L and of a formula of L. We will define the notion of a *term* of a first order language L as follows:

(i) Variables and constants of L are terms of L.
(ii) If t_1, ..., t_n are terms of L and f_i^n is a function letter of L, then $f_i^n t_1 ... t_n$ is a term of L.
(iii) The terms of L are only those things generated by clauses (i) and (ii).

Note that clause (iii) (the "extremal clause") needs to be made more rigorous; we shall make it so later on in the course.

An *atomic formula* of L is an expression of the form $P_i^n t_1 ... t_n$, where P_i^n is a predicate letter of L and t_1, ..., t_n are terms of L. Finally, we define *formula* of L as follows:

(i) An atomic formula of L is a formula of L.
(ii) If A is a formula of L, then so is ~A.
(iii) If A and B are formulae of L, then $(A \supset B)$ is a formula of L.
(iv) If A is a formula of L, then for any i, (x_i) A is a formula of L.
(v) The formulae of L are only those things that are required to be so by clauses (i)-(iv).

Here, as elsewhere, we use 'A', 'B', etc. to range over formulae.

Let x_i be a variable and suppose that $(x_i)B$ is a formula which is a part of a formula A. Then B is called the *scope* of the particular occurrence of the quantifier (x_i) in A. An occurrence of a variable x_i in A is *bound* if it falls within the scope of an occurrence of the quantifier (x_i), or if it occurs inside the quantifier (x_i) itself; and otherwise it is *free*. A *sentence* (or *closed formula*) of L is a formula of L in which all the occurrences of variables are bound.

Note that our definition of formula allows a quantifier (x_i) to occur within the scope of another occurrence of the same quantifier (x_i), e.g. $(x_1)(P_1^1 x_1 \supset (x_1)\ P_2^1 x_1)$. This is a bit hard to read, but is equivalent to $(x_1)(P_1^1 x_1 \supset (x_2)\ P_2^1 x_2)$. Formulae of this kind could be excluded from first order languages; this could be done without loss of expressive power, for example, by changing our clause (iv) in the definition of formula to a clause like:

(iv') If A is a formula of L, then for any i, (x_i) A is a formula of L, provided that (x_i) does not occur in A.

(We may call the restriction in (iv') the "nested quantifier restriction"). Our definition of formula also allows a variable to occur both free and bound within a single formula; for example, $P_1^1 x_1 \supset (x_1)\ P_2^1 x_1$ is a well formed formula in a language containing P_1^1 and P_2^1. A restriction excluding this kind of formulae could also be put in, again without loss of expressive power in the resulting languages. The two restrictions mentioned were adopted by Hilbert and Ackermann, but it is now common usage not to impose them in the definition of formula of a first order language. We will follow established usage, not imposing the

restrictions, although imposing them might have some advantages and no important disadvantadge.

We have described our official notation; however, we shall often use an unofficial notation. For example, we shall often use 'x', 'y', 'z', etc. for variables, while officially we should use 'x_1', 'x_2', etc. A similar remark applies to predicates, constants, and function letters. We shall also adopt the following unofficial abbreviations:

$(A \vee B)$ for $(\sim A \supset B)$;
$(A \wedge B)$ for $\sim(A \supset \sim B)$;
$(A \equiv B)$ for $((A \supset B) \wedge (B \supset A))$;
$(\exists x_i)\ A$ for $\sim(x_i)\ \sim A$.

Finally, we shall often omit parentheses when doing so will not cause confusion; in particular, outermost parentheses may usually be omitted (e.g. writing $A \supset B$ for $(A \supset B)$). It is important to have parentheses in our official notation, however, since they serve the important function of *disambiguating* formulae. For example, if we did not have parentheses (or some equivalent) we would have no way of distinguishing the two readings of $A \supset B \supset C$, viz. $(A \supset (B \supset C))$ and $((A \supset B) \supset C)$. Strictly speaking, we ought to prove that our official use of parentheses successfully disambiguates formulae. (Church proves this with respect to his own use of parentheses in his *Introduction to Mathematical Logic*.)

Eliminating Function Letters

In principle, we are allowing function letters to occur in our languages. In fact, in view of a famous discovery of Russell, this is unnecessary: if we had excluded function letters, we would not have decreased the expressive power of first order languages. This is because we can eliminate function letters from a formula by introducing a new n+1-place predicate letter for each n-place function letter in the formula. Let us start with the simplest case. Let f be an n-place function letter, and let F be a new n+1-place predicate letter. We can then rewrite

$$f(x_1, ..., x_n) = y$$

as

$$F(x_1, ..., x_n, y).$$

If P is a one-place predicate letter, we can then rewrite

$$P(f(x_1, ..., x_n))$$

as

$$(\exists y)\ (F(x_1, ..., x_n, y) \wedge P(y)).$$

The general situation is more complicated, because formulae can contain complex terms like f(g(x)); we must rewrite the formula f(g(x)) = y as $(\exists z)\ (G(x, z) \wedge F(z, y))$. By repeated applications of Russell's trick, we can rewrite all formulae of the form t = x, where t is a term. We can then rewrite all formulae, by first rewriting

$$A(t_1, ..., t_n)$$

as

$$(\exists x_1)...(\exists x_n)\ (x_1 = t_1 \wedge ... \wedge x_n = t_n \wedge A(x_1, ..., x_n)),$$

and finally eliminating the function letters from the formulae $x_i = t_i$.

Note that we have two different ways of rewriting the negation of a formula $A(t_1,...,t_n)$. We can either simply negate the rewritten version of $A(t_1, ..., t_n)$:

$$\sim(\exists x_1)...(\exists x_n)\ (x_1 = t_1 \wedge ... \wedge x_n = t_n \wedge A(x_1, ..., x_n));$$

or we can rewrite it as

$$(\exists x_1)...(\exists x_n)\ (x_1 = t_1 \wedge ... \wedge x_n = t_n \wedge \sim A(x_1, ..., x_n)).$$

Both versions are equivalent. Finally, we can eliminate constants in just the same way we eliminated function letters, since $x = a_i$ can be rewritten P(x) for a new unary predicate P.

Interpretations

By an *interpretation* of a first order language L (or a *model* of L, or a *structure appropriate for* L), we mean a pair <D, F>, where D (the *domain*) is a nonempty set, and F is a function that assigns appropriate objects to the constants, function letters and predicate letters of L. Specifically,

- F assigns to each constant of L an element of D;
- F assigns to each n-place function letter an n-place function with domain D^n and range included in D; and

- F assigns to each n-place predicate letter of L an n-place relation on D (i.e., a subset of D^n).

Let I = <D, F> be an interpretation of a first order language L. An *assignment* in I is a function whose domain is a subset of the set of variables of L and whose range is a subset of D (i.e., an assignment that maps some, possibly all, variables into elements of D). We now define, for given I, and for all terms t of L and assignments s in I, the function Den(t,s) (the *denotation* (in I) of a term t with respect to an assignment s (in I)), that (when defined) takes a term and an assignment into an element of D, as follows:

(i) if t is a constant, Den(t, s)=F(t);
(ii) if t is a variable and s(t) is defined, Den(t, s)=s(t); if s(t) is undefined, Den(t, s) is also undefined;
(iii) if t is a term of the form $f_i^n(t_1, ..., t_n)$ and $Den(t_j,s)=b_j$ (for j = 1, ..., n), then Den(t, s)=$F(f_i^n)(b_1, ..., b_n)$; if $Den(t_j,s)$ is undefined for some $j\leq n$, then Den(t,s) is also undefined.

Let us say that an assignment s is *sufficient* for a formula A if and only if it makes the denotations of all terms in A defined, if and only if it is defined for every variable occurring free in A (thus, note that all assignments, including the empty one, are sufficient for a sentence). We say that an assignment s in I *satisfies* (in I) a formula A of L just in case

(i) A is an atomic formula $P_i^n(t_1, ..., t_n)$, s is sufficient for A and $<Den(t_1,s),...,Den(t_n,s)> \in F(P_i^n)$; or
(ii) A is ~B, s is sufficient for B but s does not satisfy B; or
(iii) A is $(B \supset C)$, s is sufficient for B and C and either s does not satisfy B or s satisfies C; or
(iv) A is $(x_i)B$, s is sufficient for A and for every s' that is sufficient for B and such that for all $j\neq i$, $s'(x_j)=s(x_j)$, s' satisfies B.

We also say that a formula A is *true* (in an interpretation I) *with respect to an assignment* s (in I) iff A is satisfied (in I) by s; if s is sufficient for A and A is not true with respect to s, we say that A is *false with respect to* s.

If A is a sentence, we say that A is *true in* I iff all assignments in I satisfy A (or, what is equivalent, iff at least one assignment in I satisfies A).

We say that a formula A of L is *valid* iff for every interpretation I and all assignments s in I, A is true (in I) with respect to s (we also say, for languages L containing P_1^2, that a formula A of L is *valid in the logic with identity* iff for every interpretation I=<D,F> where $F(P_1^2)$ is the identity relation on D, and all assignments s in I, A is true (in I) with respect to

s). More generally, we say that A is a *consequence* of a set Γ of formulas of L iff for every interpretation I and every assignment s in I, if all the formulas of Γ are true (in I) with respect to s, then A is true (in I) with respect to s. Note that a sentence is valid iff it is true in all its interpretations iff it is a consequence of the empty set. We say that a formula A is *satisfiable* iff for some interpretation I, A is true (in I) with respect to some assignment in I. A sentence is satisfiable iff it is true in some interpretation.

For the following definitions, let an interpretation I=<D,F> be taken as fixed. If A is a formula whose only free variables are x_1, ..., x_n, then we say that the n-tuple <a_1, ..., a_n> ($\in D^n$) *satisfies* A (in I) just in case A is satisfied by an assignment s (in I), where $s(x_i) = a_i$ for i = 1, ..., n. (In the case n = 1, we say that a satisfies A just in case the 1-tuple <a> does.) We say that A *defines* (in I) the relation R ($\subseteq D^n$) iff R={<b_1, ..., b_n>: <b_1,...,b_n> satisfies A}. An n-place relation R ($\subseteq D^n$) is *definable* (in I) in L iff there is a formula A of L whose only free variables are x_1, ..., x_n, and such that A defines R (in I). Similarly, if t is a term whose free variables are x_1, ..., x_n, then we say that t defines the function h, where $h(a_1, ..., a_n) = b$ just in case Den(t,s)=b for some assignment s such that $s(x_i) = a_i$. (So officially formulae and terms only define relations and functions when their free variables are x_1, ..., x_n for some n; in practice we shall ignore this, since any formula can be rewritten so that its free variables form an initial segment of all the variables.)

The Language of Arithmetic

We now give a specific example of a first order language, along with its *standard* or *intended* interpretation. The language of arithmetic contains one constant a_1, one function letter f^1_1, one 2-place predicate letter P^2_1, and two 3-place predicate letters P^3_1, and P^3_2. The standard interpretation of this language is <**N**, F> where **N** is the set {0, 1, 2, ...} of natural numbers, and where

$F(a_1) = 0$;
$F(f^1_1)$ = the successor function $s(x) = x+1$;
$F(P^2_1)$ = the identity relation {<x, y>: x = y};
$F(P^3_1)$ = {<x, y, z>: x + y = z}, the graph of the addition function;
$F(P^3_2)$ = {<x, y, z>: x·y = z}, the graph of the multiplication function.

We also have an unofficial notation: we write

0 for a_1;
x' for f^1_1x;
x = y for P^2_1xy;
A(x, y, z) for P^3_1xyz;

M(x, y, z) for P_2^3xyz.

This presentation of the language of arithmetic is rather atypical, since we use a function letter for successor but we use predicates for addition and multiplication. Note, however, that formulae of a language involving function letters for addition and multiplication instead of the corresponding predicate letters could be rewritten as formulae of the language of arithmetic via Russell's trick.

A *numeral* is a term of the form **0**'⋯', i.e. the constant **0** followed by zero or more successor function signs. The numeral for a number n is zero followed by n successor function signs; we shall use the notation $\mathbf{0}^{(n)}$ for the numeral for n (note that 'n' is not a variable of our formal system, but a variable of our informal talk). It may be noted that the only terms of the language of arithmetic, as we have set it up, are the numerals and expressions of the form $x_i'\cdots'$.

Finally, note that for the language of arithmetic, we can define satisfaction in terms of truth and substitution. This is because a k-tuple $<n_1, ..., n_k>$ of numbers satisfies $A(x_1, ..., x_k)$ just in case the sentence $A(\mathbf{0}^{(n1)}, ..., \mathbf{0}^{(nk)})$ is true (where $A(\mathbf{0}^{(n1)}, ..., \mathbf{0}^{(nk)})$ comes from A by substituting the numeral $\mathbf{0}^{(ni)}$ for all of the free occurrences of the variable x_i).

Lecture II

The Language RE

We shall now introduce the language RE. This is not strictly speaking a first order language, in the sense just defined. However, it can be regarded as a fragment of the first order language of arithmetic.

In RE, the symbols $\wedge$ and $\vee$ are the primitive connectives rather than $\sim$ and $\supset$. RE further contains the quantifier symbol $\exists$ and the symbol $<$ as primitive. The terms and atomic formulae of RE are those of the language of arithmetic as presented above. Then the notion of formula of RE is defined as follows:

(i) An atomic formula of RE is a formula.
(ii) If A and B are formulae, so are $(A \wedge B)$ and $(A \vee B)$.
(iii) If t is a term not containing the variable x_i, and A is a formula, then $(\exists x_i)$ A and $(x_i < t)$ A are formulae.
(iv) Only those things generated by the previous clauses are formulae.

The intended interpretation of RE is the same as the intended interpretation of the first order language of arithmetic (it is the same pair <D,F>). Such notions as truth and satisfaction for formulae of RE and definability by formulae of RE are defined in a way similar to that in which they would be defined for the language of arithmetic using our general definitions of truth and satisfaction; in the appropriate clause, the quantifier $(x_i < t)$ is intuitively interpreted as "for all x_i less than t..." (it is a so called “bounded universal quantifier”).

Note that RE does not contain negation, the conditional or unbounded universal quantification. These are not definable in terms of the primitive symbols of RE. The restriction on the term t of $(x_i < t)$ in clause (iii) above is necessary if we are to exclude unbounded universal quantification from RE, because $(x_i < x_i')$ B is equivalent to (x_i) B.

The Intuitive Concept of Computability and its Formal Counterparts

The importance of the language RE lies in the fact that with its help we will offer a definition that will try to capture the intuitive concept of computability. We call an n-place relation on the set of natural numbers *computable* if there is an effective procedure which, when given an arbitrary n-tuple as input, will in a finite time yield as output 'yes' or 'no' as the n-tuple is or isn't in the relation. We call an n-place relation *semi-computable* if there is an effective

procedure such that, when given an n-tuple which is in the relation as input, it eventually yields the output 'yes', and which when given an n-tuple which is not in the relation as input, does not eventually yield the output 'yes'. We do *not* require the procedure to eventually yield the output 'no' in this case. An n-place total function ϕ is called *computable* if there is an effective procedure that, given an n-tuple $<p_1,...,p_n>$ as input, eventually yields $\phi(p_1,...,p_n)$ as output (unless otherwise noted, an n-place function is defined for *all* n-tuples of natural numbers (or all natural numbers if n = 1) —this is what it means for it to be total; and only takes natural numbers as values.)

It is important to note that we place no time limit on the length of computation for a given input, as long as the computation takes place within a finite amount of time. If we required there to be a time limit which could be effectively determined from the input, then the notions of computability and semi-computability would collapse. For let S be a semi-computable set, and let P be a semi-computation procedure for S. Then we could find a computation procedure for S as follows. Set P running on input x, and determine a time limit L from x. If $x \in S$, then P will halt sometime before the limit L. If we reach the limit L and P has not halted, then we will know that $x \notin P$. So as soon as P halts or we reach L, we give an output 'yes' or 'no' as P has or hasn't halted. We will see later in the course, however, that the most important basic result of recursion theory is that the unrestricted notions of computability and semi-computability do not coincide: there are semi-computable sets and relations that are not computable.

The following, however, is true (the *complement* of an n-place relation R (-R) is the collection of n-tuples of natural numbers not in R):

Theorem: A set S (or relation R) is computable iff S (R) and its complement are semi-computable.
Proof: If a set S is computable, there is a computation procedure P for S. P will also be a semi-computation procedure for S. To semi-compute the complement of S, simply follow the procedure of changing a ‘no’ delivered by P to a ‘yes’. Now suppose we have semi-computation procedures for both S and its complement. To compute whether a number n is in S, run simultaneously the two semi-computation procedures on n. If the semi-computation procedure for S delivers a ‘yes’, the answer is yes; if the semi-computation procedure for -S delivers a ‘yes’, the answer is no.

We intend to give formal definitions of the intuitive notions of computable set and relation, semi-computable set and relation, and computable function. Formal definitions of these notions were offered for the first time in the thirties. The closest in spirit to the ones that will be developed here were based on the formal notion of λ-definable function presented by Church. He invented a formalism that he called ‘λ-calculus’, introduced the notion of a function definable in this calculus (a λ-definable function), and put forward the thesis that the computable functions are exactly the λ-definable functions. This is Church’s

thesis in its original form. It states that a certain formal concept correctly captures a certain intuitive concept.

Our own approach to recursion theory will be based on the following form of Church's thesis:

Church's Thesis: A set S (or relation R) is semi-computable iff S (R) is definable in the language RE.

We also call the relations definable in RE *recursively enumerable* (or *r.e.*). Given our previous theorem, we can define a set or relation to be *recursive* if both it and its complement are r.e.

Our version of Church's Thesis implies that the recursive sets and relations are precisely the computable sets and relations. To see this, suppose that a set S is computable. Then, by the above theorem, S and its complement are semi-computable, and hence by Church's Thesis, both are r.e.; so S is recursive. Conversely, suppose S is recursive. Then S and -S are both r.e., and therefore by Church's Thesis both are semi-computable. Then by the above theorem, S is computable.

The following theorem will be of interest for giving a formal definition of the remaining intuitive notion of computable function:

Theorem: A total function $\phi(m_1,...,m_n)$ is computable iff the n+1 place relation $\phi(m_1,...,m_n)=p$ is semi-computable iff the n+1 place relation $\phi(m_1,...,m_n)=p$ is computable.
Proof: If $\phi(m_1,...,m_n)$ is computable, the following is a procedure that computes (and hence also semi-computes) the n+1 place relation $\phi(m_1,...,m_n)=p$. Given an input $<p_1,...,p_n,p>$, compute $\phi(p_1,...,p_n)$. If $\phi(p_1,...,p_n)=p$, the answer is yes; if $\phi(p_1,...,p_n)\neq p$, the answer is no. Now suppose that the n+1 place relation $\phi(m_1,...,m_n)=p$ is semi-computable (thus the following would still follow under the assumption that it is computable); then to compute $\phi(p_1,...,p_n)$, run the semi-computation procedure on sufficient n+1 tuples of the form $<p_1,...,p_n,m>$, via some time-sharing trick. For example, run five steps of the semi-computation procedure on $<p_1,...,p_n,0>$, then ten steps on $<p_1,...,p_n,0>$ and $<p_1,...,p_n,1>$, and so on, until you get the n+1 tuple $<p_1,...,p_n,p>$ for which the 'yes' answer comes up. And then give as output p.

A partial function is a function defined on a subset of the natural numbers which need not be the set of all natural numbers. We call an n-place partial function *partial computable* iff there is a procedure which delivers $\phi(p_1,...,p_n)$ as output when ϕ is defined for the argument tuple $<p_1,...,p_n>$, and that does not deliver any output if ϕ is undefined for the argument tuple $<p_1,...,p_n>$. The following result, partially analogous to the above, still holds:

Theorem: A function $\phi(m_1,...,m_n)$ is partial computable iff the n+1 relation $\phi(m_1,...,m_n)=p$

is semi-computable.

Proof: Suppose $\phi(m_1,...,m_n)$ is partial computable; then the following is a semi-computation procedure for the n+1 relation $\phi(m_1,...,m_n)=p$: given an argument tuple $<p_1,...,p_n,p>$, apply the partial computation procedure to $<p_1,...,p_n>$; if and only if it eventually delivers p as output, the answer is yes. Now suppose that the n+1 relation $\phi(m_1,...,m_n)=p$ is semi-computable. Then the following is a partial computation procedure for $\phi(m_1,...,m_n)$. Given an input $<p_1,...,p_n>$, run the semi-computation procedure on n+1 tuples of the form $<p_1,...,p_n,m>$, via some time-sharing trick. For example, run five steps of the semi-computation procedure on $<p_1,...,p_n,0>$, then ten steps on $<p_1,...,p_n,0>$ and $<p_1,...,p_n,1>$, and so on. If you get an n+1 tuple $<p_1,...,p_n,p>$ for which the 'yes' answer comes up, then give as output p.

But it is not the case anymore that a function $\phi(m_1,...,m_n)$ is partial computable iff the n+1 relation $\phi(m_1,...,m_n)=p$ is computable. There is no guarantee that a partial computation procedure will provide a computation procedure for the relation $\phi(m_1,...,m_n)=p$; if ϕ is undefined for $<p_1,...,p_n>$, the partial computation procedure will never deliver an output, but we may have no way of telling that it will not.

In view of these theorems, we now give formal definitions that intend to capture the intuitive notions of computable function and partial computable function. An n-place partial function is called *partial recursive* iff its graph is r.e. An n-place total function is called *total recursive* (or simply *recursive*) iff its graph is r.e. Sometimes the expression 'general recursive' is used instead of 'total recursive', but this is confusing, since the expression 'general recursive' was originally used not as opposed to 'partial recursive' but as opposed to 'primitive recursive'.

It might seem that we can avoid the use of partial functions entirely, say by replacing a partial function ϕ with a total function ψ which agrees with ϕ wherever ϕ is defined, and which takes the value 0 where ϕ is undefined. Such a ψ would be a total extension of ϕ, i.e. a total function which agrees with ϕ wherever ϕ is defined. However, this will not work, since there are some partial recursive functions which are not totally extendible, i.e. which do not have any total extensions which are recursive functions. (We shall prove this later on in the course.)

Our version of Church's Thesis implies that a function is computable iff it is recursive. To see this, suppose that ϕ is a computable function. Then, by one of the theorems above, its graph is semi-computable, and so by Church's Thesis, it is r.e., and so ϕ is recursive. Conversely, suppose that ϕ is recursive. Then ϕ's graph is r.e., and by Church's Thesis it is semi-computable; so by the same theorem, ϕ is computable.

Similarly, our version of Church's Thesis implies that a function is partial computable iff it is partial recursive.

We have the result that if a total function has a semi-computable graph, then it has a computable graph. That means that the complement of the graph is also semi-computable.

We should therefore be able to show that the graph of a recursive function is also recursive. In order to do this, suppose that ϕ is a recursive function, and let R be its graph. R is r.e., so it is defined by some RE formula $B(x_1, ..., x_n, x_{n+1})$. To show that R is recursive, we must show that -R is r.e., i.e. that there is a formula of RE which defines -R. A natural attempt is the formula

$$(\exists x_{n+2})(B(x_1, ..., x_n, x_{n+2}) \wedge x_{n+1} \neq x_{n+2}).$$

This does indeed define -R as is easily seen, but it is not a formula of RE, for its second conjunct uses negation, and RE does not have a negation sign. However, we can fix this problem if we can find a formula of RE that defines the nonidentity relation $\{<m,n>:m\neq n\}$.

Let us define the formula

$$\text{Less } (x, y) =_{df.} (\exists z)\, A(x, z', y).$$

Less (x, y) defines the less-than relation $\{<m, n>: m < n\}$. We can now define inequality as follows:

$$x \neq y =_{df.} \text{Less}(x, y) \vee \text{Less } (y, x).$$

This completes the proof that the graph of a total recursive function is a recursive relation, and also shows that the less-than and nonidentity relations are r.e., which will be useful in the future.

While we have not introduced bounded existential quantification as a primitive notation of RE, we can define it in RE, as follows:

$$(\exists x < t)\, B =_{df.} (\exists x)\, (\text{Less}(x, t) \wedge B).$$

In practice, we shall often write 'x < y' for 'Less (x, y)'. However, it is important to distinguish the defined symbol '<' from the primitive symbol '<' as it appears within the bounded universal quantifier. We also define

$$(\exists x \leq t)\, B(x) =_{df.} (\exists x < t)\, B(x) \vee B(t);$$
$$(x \leq t)\, B(x) =_{df.} (x < t)\, B(x) \wedge B(t).$$

The Status of Church's Thesis

Our form of Church's thesis is that the intuitive notion of semi-computability and the formal notion of recursive enumerability coincide. That is, a set or relation is semi-computable iff it

is r.e. Schematically:

r.e. = semi-computable.

The usual form of Church's Thesis is: recursive = computable. But as we saw, our form of Church's Thesis implies the usual form.

In some introductory textbooks on recursion theory Church's Thesis is assumed in proofs, e.g. in proofs that a function is recursive that appeal to the existence of an effective procedure (in the intuitive sense) that computes it. (Hartley Rogers' *Theory of Recursive Functions and Effective Computability* is an example of this.) There are two advantages to this approach. The first is that the proofs are intuitive and easier to grasp than very "formal" proofs. The second is that it allows the student to cover relatively advanced material fairly early on. The disadvantage is that, since Church's Thesis has not actually been proved, the student never sees the proofs of certain fundamental theorems. We shall therefore not assume Church's Thesis in our proofs that certain sets or relations are recursive. (In practice, if a recursion theorist is given an informal effective procedure for computing a function, he or she will regard it as proved that that function is recursive. However, an experienced recursion theorist will easily be able to convert this proof into a rigorous proof which makes no appeal whatsoever to Church's Thesis. So working recursion theorists should not be regarded as appealing to Church's Thesis in the sense of assuming an unproved conjecture. The beginning student, however, will not in general have the wherewithal to convert informal procedures into rigorous proofs.)

Another usual standpoint in some presentations of recursion theory is that Church's Thesis is not susceptible of proof or disproof, because the notion of recursiveness is a precise *mathematical* notion and the notion of computability is an *intuitive* notion. Indeed, it has not in fact been proved (although there is a lot of evidence for it), but in the author's opinion, no one has shown that it is *not susceptible* of proof or disproof. Although the notion of computability is not taken as primitive in standard formulations of mathematics, say in set theory, it does have many intuitively obvious properties, some of which we have just used in the proofs of perfectly rigorous theorems. Also, $y = x!$ is evidently computable, and so is $z=x^y$ (although it is not immediately obvious that these functions are recursive, as we have defined these notions). So suppose it turned out that one of these functions was not recursive. That would be an absolute disproof of Church's Thesis. Years before the birth of recursion theory a certain very wide class of computable functions was isolated, that later would come to be referred to as the class of "primitive recursive" functions. In a famous paper, Ackermann presented a function which was evidently computable (and which is in fact recursive), but which was not primitive recursive. If someone had conjectured that the computable functions are the primitive recursive functions, Ackermann's function would have provided an absolute disproof of that conjecture. (Later we will explain what is the class of primitive recursive functions and we will define Ackermann's function.) For

another example, note that the composition of two computable functions is intuitively computable; so, if it turned out that the formal notion of recursiveness was not closed under composition, this would show that Church's Thesis is wrong.

Perhaps some authors acknowledge that Church's Thesis is open to absolute disproof, as in the examples above, but claim that it is not open to proof. However, the conventional argument for this goes on to say that since computability and semi-computability are merely intuitive notions, not rigorous mathematical notions, a proof of Church's Thesis could not be given. This position, however, is not consistent if the intuitive notions in question cannot be used in rigorous mathematical arguments. Then a disproof of Church's Thesis would be impossible also, for the same reason as a proof. In fact, suppose for example that we could give a list of principles intuitively true of the computable functions and were able to prove that the only class of functions with these properties was exactly the class of the recursive functions. We would then have a proof of Church's Thesis. While this is in principle possible, it has not yet been done (and it seems to be a very difficult task).

In any event, we can give a perfectly rigorous proof of *one half* of Church's thesis, namely that every r.e relation (or set) is semi-computable.

Theorem: Every r.e. relation (or set) is semi-computable.
Proof: We show by induction on the complexity of formulae that for any formula B of RE, the relation that B defines is semi-computable, from which it follows that all r.e. relations are semi-computable. We give, for each formula B of RE, a procedure P_B which is a semi-computation of the relation defined by B.

If B is atomic, then it is easy to see that an appropriate P_B exists; for example, if B is the formula $x_1''' = x_2'$, then P_B is the following procedure: add 3 to the first input, then add 1 to the second input, and see if they are the same, and if they are, halt with output 'yes'.

If B is $(C \wedge D)$, then P_B is the following procedure: first run P_C, and if it halts with output 'yes', run P_D; if that also halts, then halt with output 'yes'.

If B is $(C \vee D)$, then P_B is as follows. Run P_C and P_D simultaneously via some time-sharing trick. (For example, run 10 steps of P_C, then 10 steps of P_D, then 10 more steps of P_C, ...) As soon as one answers 'yes', then let P_B halt with output 'yes'.

Suppose now that B is $(y < t)\ C(x_1, ..., x_n, y)$. If t is a numeral $\mathbf{0}^{(p)}$, then $<m_1, ..., m_n>$ satisfies B just in case all of $<m_1, ..., m_n, 0>$ through $<m_1, ..., m_n, p-1>$ satisfy C, so run P_C on input $<m_1, ..., m_n, 0>$; if P_C answers yes, run P_C on input $<m_1, ..., m_n, 1>$, ... If you reach p-1 and get an answer yes, then $<m_1, ..., m_n>$ satisfies B, so halt with output 'yes'. If t is a term $x_i'\cdots'$, then the procedure is basically the same. Given an input which includes the values $m_1, ..., m_n$ of $x_1, ..., x_n$, as well as the value of x_i, first calculate the value p of the term t, and then run P_C on $<m_1, ..., m_n, 0>$ through $<m_1, ..., m_n, p-1>$, as above. So in either case, an appropriate P_B exists.

Finally, if $B = (\exists y)\ C(x_1, ..., x_n, y)$, then P_C is as follows: given input $<m_1, ..., m_n>$, run P_C on $<m_1, ..., m_n, k>$ simultaneously for all k and wait for P_C to deliver 'yes' for some k.

Again, we use a time-sharing trick; for example: first run P_C on $<m_1, ..., m_n, 0>$ for 10 steps, then run P_C on $<m_1, ..., m_n, 0>$ and $<m_1, ..., m_n, 1>$ for 20 steps each, then .... Thus, an appropriate P_B exists in this case as well, which completes the proof.

This proof cannot be formalized in set theory, so in that sense the famous thesis of the logicists that all mathematics can be done in set theory might be wrong. But a weaker thesis that every intuitive mathematical notion can always be replaced by one definable in set theory (and coextensive with it) might yet be right.

Kreisel's opinion—in a review—appears to be that computability is a legitimate primitive only for intuitionistic mathematics. In classical mathematics it is not a primitive, although (*pace* Kreisel) it could be taken to be one. In fact the above argument, that the recursive sets are all computable, is not intuitionistically valid, because it assumes that a number will be either in a set or in its complement. (If you don't know what intuitionism is, don't worry.)

It is important to notice that recursiveness (and recursive enumerability) is a property of a set, function or relation, not a *description* of a set, function or relation. In other words, recursiveness is a property of extensions, not intensions. To say that a set is r.e. is just to say that there exists a formula in RE which defines it, and to say that a set is recursive is to say that there exists a pair of formulae in RE which define it and its complement. But you don't necessarily have to know what these formulae are, contrary to the point of view that would be taken on this by intuitionistic or constructivist mathematicians. We might have a theory of recursive descriptions, but this would not be conventional recursive function theory. So for example, we know that any finite set is recursive; every finite set will be defined in RE by a formula of the form $x_1=\mathbf{0}^{(k1)}\vee...\vee x_n=\mathbf{0}^{(kn)}$, and its complement by a formula of the form $x_1\neq\mathbf{0}^{(k1)}\wedge...\wedge x_n\neq\mathbf{0}^{(kn)}$. But we may have no procedure for deciding whether something is in a certain finite set or not - finding such a procedure might even be a famous unsolved problem. Consider this example: let S = {n: at least n consecutive 7's appear in the decimal expansion of π}. Now it's hard to say what particular n's are in S (it's known that at least four consecutive 7's appear, but we certainly don't know the answer for numbers much greater than this), but nonetheless S is recursive. For, if $n \in S$ then any number less than n is also in S, so S will either be a finite initial segment of the natural numbers, or else it will contain all the natural numbers. Either way, S is recursive.

There is, however, an intensional version of Church's Thesis that, although hard to state in a rigorous fashion, seems to be true in practice: whenever we have an intuitive procedure for semi-computing a set or relation, it can be "translated" into an appropriate formula of the formalism RE, and this can be done in some sense effectively (the "translation" is intuitively computable). This version of Church's Thesis operates with the notion of arbitrary descriptions of sets or relations (in English, or in mathematical notation, say), which is somewhat vague. It would be good if a more rigorous statement of this version of Church's Thesis could be made.

The informal notion of computability we intend to study in this course is a notion different from a notion of analog computability that might be studied in physics, and for which there is no reason to believe that Church's Thesis holds. It is not at all clear that every function of natural numbers computable by a physical device, that can use analog properties of physical concepts, is computable by a digital algorithm. There have been some discussions of this matter in a few papers, although the ones known to the author are quite complicated. Here we will make a few rather unsophisticated remarks.

There are certain numbers in physics known as universal constants. Some of these numbers are given in terms of units of measure, an are different depending on the system of units of measures adopted. Some other of these numbers, however, are not given in terms of units of measure, for example, the electron-proton mass ratio; that is, the ratio of the mass of an electron to the mass of a proton. We know that the electron-proton mass ratio is a positive real number *r* less than 1 (the proton is heavier than the electron). Consider the following function ψ: $\psi(k)$ = the kth number in the decimal expansion of *r*. (There are two ways of expanding finite decimals, with nines at the end or with zeros at the end; in case *r* is finite, we arbitrarily stipulate that its expansion is with zeros at the end.) As far as I know, nothing known in physics allows us to ascribe to *r* any mathematical properties (e.g., being rational or irrational, being algebraic or transcendental, even being a finite or an infinite decimal). Also, as far as I know, it is not known whether this number is recursive, or Turing computable.

However, people do attempt to measure these constants. There might be problems in carrying out the measurement to an arbitrary degree of accuracy. It might take longer and longer to calculate each decimal place, it might take more and more energy, time might be finite, etc. Nevertheless, let us abstract from all these difficulties, assuming, e.g., that time is infinite. Then, as far as I can see, there is no reason to believe that there cannot be any physical device that would actually calculate each decimal place of *r*. But this is not an algorithm in the standard sense. ψ might even then be uncomputable in the standard sense.

Let us review another example. Consider some quantum mechanical process where we can ask, e.g., whether a particle will be emitted by a certain source in the next second, or hour, etc. According to current physics, this kind of thing is not a deterministic process, and only relevant probabilities can be given that a particle will be emitted in the next second, say. Suppose we set up the experiment in such a way that there is a probability of 1/2 for an emission to occur in the next second, starting at some second s_0. We can then define a function $\chi(k) = 1$ if an emission occurs in s_k, and = 0 if an emission does not occur in s_k. This is not a universally defined function like ψ, but if time goes on forever, this experiment is a physical device that gives a universally defined function. There are only a denumerable number of recursive functions (there are only countably many strings in RE, and hence only countably many formulae). In terms of probability theory, for any infinite sequence such as the one determined by χ there is a probability of 1 that it will lie outside any denumerable set (or set of measure zero). So in a way we can say with certainty that χ, even though

"computable" by our physical device, is not recursive, or, equivalently, Turing computable. (Of course, χ may turn out to be recursive if there is an underlying deterministic structure to our experiment, but assuming quantum mechanics, there is not.) This example again illustrates the fact that the concept of physical computability involved is not the informal concept of computability referred to in Church's Thesis.

Lecture III

The Language Lim

In the language RE, we do not have a negation operator. However, sometimes, the complement of a relation definable by a formula of RE is definable in RE by means of some trick. We have already seen that the relation defined by $t_1 \neq t_2$ (where t_1, t_2 are two terms of RE) is definable in RE, and whenever B defines the graph of a total function, the complement of this graph is definable.

In RE we also do not have the conditional. However, if A is a formula whose negation is expressible in RE, say by a formula A* (notice that A need not be expressible in RE), then the conditional $(A \supset B)$ would be expressible by means of $(A^* \vee B)$ (provided B is a formula of RE); thus, for example, $(t_1 = t_2 \supset B)$ is expressible in RE, since $t_1 \neq t_2$ is. So when we use the conditional in our proofs by appeal to formulae of RE, we'll have to make sure that if a formula appears in the antecedent of a conditional, its negation is expressible in the language. In fact, this requirement is too strong, since a formula appearing in the antecedent of a conditional may appear without a negation sign in front of it when written out only in terms of negation, conjunction and disjunction. Consider, for example, a formula

$$(A \supset B) \supset C,$$

in which the formula A appears as a part in the antecedent of a conditional. This conditional is equivalent to

$$(\sim A \vee B) \supset C,$$

and in turn to

$$\sim(\sim A \vee B) \vee C,$$

and to

$$(A \wedge \sim B) \vee C.$$

In the last formula, in which only negation, conjunction and disjunction are used, A appears purely positively, so it's not necessary that its negation be expressible in RE in order for $(A \supset B) \supset C$ to be expressible in RE.

A bit more rigorously, we give an inductive construction that determines when an

occurrence of a formula A in a formula F whose only connectives are ~ and ⊃ is positive or negative: if A is F, A's occurrence in F is positive; if F is ~B, A's occurrence in F is negative if it is positive in B, and vice versa; if F is (B⊃C), an occurrence of A in B is negative if positive in B, and vice versa, and an occurrence of A in C is positive if positive in C, and negative if negative in C.

It follows from this that if an occurrence of a formula appears as a part in another formula in an even number of antecedents (e.g., A in the formula of the example above), the corresponding occurrence will be positive in an ultimately reduced formula employing only negation, conjunction and disjunction. If an occurrence of a formula appears as a part in another formula in an odd number of antecedents (e.g., B in the formula above), the corresponding occurrence will appear with a negation sign in front of it in the ultimately reduced formula (i.e., it will be negative) and we will have to make sure that the negated formula is expressible in RE.

In order to avoid some of these complications involved in working within RE, we will now define a language in which we have unrestricted use of negation, but such that all the relations definable in it will also be definable in RE. We will call this language *Lim*. Lim has the same primitive symbols as RE, plus a symbol for negation (~). The terms and atomic formulae of Lim are just those of RE. Then the notion of formula of Lim is defined as follows:

(i) An atomic formula of Lim is a formula of Lim;
(ii) If A and B are formulae of Lim, so are ~A, $(A \wedge B)$ and $(A \vee B)$;
(iii) If t is a term not containing the variable x_i, and A is a formula of Lim, then $(\exists x_i<t))$ A and $(x_i < t)$ A are formulae of Lim;
(iv) Only those things generated by the previous clauses are formulae.

Notice that in Lim we no longer have unbounded existential quantification, but only bounded existential quantification. This is the price of having negation in Lim.

Lim is weaker than RE in the sense that any set or relation definable in Lim is also definable in RE. This will mean that if we are careful to define a relation using only bounded quantifiers, its complement will be definable in Lim, and hence in RE, and this will show that the relation is recursive. Call two formulae with the same free variables *equivalent* just in case they define the same set or relation. (So closed formulae, i.e. sentences, are equivalent just in case they have the same truth value.) To show that Lim is weaker than RE, we prove the following

Theorem: Any formula of Lim is equivalent to some formula of RE.
Proof: We show by induction on the complexity of formulae that if B is a formula of Lim, then both B and ~B are equivalent to formulae of RE. First, suppose B is atomic. B is then a formula of RE, so obviously B is equivalent to some RE formula. Since inequality is an

r.e. relation and the complement of the graph of any recursive function is r.e., ~B is equivalent to an RE formula. If B is ~C, then by inductive hypothesis C is equivalent to an RE formula C* and ~C is equivalent to an RE formula C**; then B is equivalent to C** and ~B (i.e., ~~C) is equivalent to C*. If B is (C $\wedge$ D), then by the inductive hypothesis, C and D are equivalent to RE formulae C* and D*, respectively, and ~C, ~D are equivalent to RE formulae C** and D**, respectively. So B is equivalent to (C* $\wedge$ D*), and ~B is equivalent to (C** $\vee$ D**). Similarly, if B is (C $\vee$ D), then B and ~B are equivalent to (C* $\vee$ D*) and (C** $\wedge$ D**), respectively. If B is ($\exists x_i < t$) C, then B is equivalent to ($\exists x_i$)(Less(x_i, t)$\wedge$C*), and ~B is equivalent to ($x_i < t$) ~C and therefore to ($x_i < t$) C**. Finally, the case of bounded universal quantification is similar.

A set or relation definable in Lim is recursive: if B defines a set or relation in Lim, then ~B is a formula of Lim that defines its complement, and so by the foregoing theorem both it and its complement are r.e. (Once we have shown that not all r.e. sets are recursive, it will follow that Lim is *strictly* weaker than RE, i.e. that not all sets and relations definable in RE are definable in Lim.) Since negation is available in Lim, the conditional is also available, as indeed are all truth-functional connectives. Because of this, showing that a set or relation is definable in Lim is a particularly convenient way of showing that it is recursive; in general, if you want to show that a set or relation is recursive, it is a good idea to show that it is definable in Lim (if you can).

We can expand the language Lim by adding extra predicate letters and function letters and interpreting them as recursive sets and relations and recursive functions. If we do so, the resulting language will still be weaker than RE:

Theorem: Let Lim' be an expansion of Lim in which the extra predicates and function letters are interpreted as recursive sets and relations and recursive functions. Then every formula of Lim' is equivalent to some formula of RE.

Proof: As before, we show by induction on the complexity of formulae that each formula of Lim' and its negation are equivalent to RE formulae. The proof is analogous to the proof of the previous theorem. Before we begin the proof, let us note that every term of Lim' stands for a recursive function; this is simply because the function letters of Lim' define recursive functions, and the recursive functions are closed under composition. So if t is a term of Lim', then both t = y and ~(t = y) define recursive relations and are therefore equivalent to formulae of RE.

Suppose B is the atomic formula $P(t_1, ..., t_n)$, where $t_1, ..., t_n$ are terms of Lim' and P is a predicate of Lim' defining the recursive relation R. Using Russell's trick, we see that B is equivalent to $(\exists x_1)...(\exists x_n)(t_1 = x_1 \wedge ... \wedge t_n = x_n \wedge P(x_1, ..., x_n))$, where $x_1, ..., x_n$ do not occur in any of the terms $t_1, ..., t_n$. Letting C_i be an RE formula which defines the relation defined by $t_i = x_i$, and letting D be an RE formula which defines the relation that P defines, we see that B is equivalent to the RE formula $(\exists x_1)...(\exists x_n)(C_1(x_1) \wedge ... C_n(x_n) \wedge D(x_1, ...,$

x_n)). To see that ~B is also equivalent to an RE formula, note that R is a recursive relation, so its complement is definable in RE, and so the formula $(\exists x_1)...(\exists x_n)(t_1 = x_1 \wedge ... \wedge t_n = x_n \wedge \sim P(x_1, ..., x_n))$, which is equivalent to ~B, is also equivalent to an RE formula.

The proof is the same as the proof of the previous theorem in the cases of conjunction, disjunction, and negation. In the cases of bounded quantification, we have to make a slight adjustment, because the term t in $(x_i < t)$ B or $(\exists x_i < t)$ B might contain new function letters. Suppose B and ~B are equivalent to the RE formulae B* and B**, and let t = y be equivalent to the RE formula C(y). Then $(x_i < t)$ B is equivalent to the RE formula $(\exists y)$ $(C(y) \wedge (x_i < y)$ B*)), and ~$(x_i < t)$ B is equivalent to $(\exists x_i < t)$ ~B, which is in turn equivalent to the RE formula $(\exists y)$ $(C(y) \wedge (\exists x_i < y)$ B**). The case of bounded existential quantification is similar.

This fact will be useful, since in RE and Lim the only bounds we have for the bounded quantifiers are terms of the forms $\mathbf{0}^{(n)}$ and $x_i'\cdots'$. In expanded languages containing function letters interpreted as recursive functions there will be other kinds of terms that can serve as bounds for quantifiers in formulae of the language, without these formulae failing to be expressible in RE.

There is a variant of Lim that should be mentioned because it will be useful in future proofs. *Lim*$^+$ is the language which is just like Lim except that it has function letters rather than predicates for addition and multiplication. (So in particular, quantifiers in Lim$^+$ can be bounded by terms containing + and ·.) It follows almost immediately from the previous theorem that every formula of Lim$^+$ is equivalent to some formula of RE. We call a set or relation *limited* if it is definable in the language Lim$^+$. We call it *strictly limited* if it is definable in Lim.

Pairing Functions

We will define a *pairing function* on the natural numbers to be a dominating total binary recursive function ϕ such that for all m_1, m_2, n_1, n_2, if $\phi(m_1, m_2) = \phi(n_1, n_2)$ then $m_1 = n_1$ and $m_2 = n_2$ (that a binary function ϕ is dominating means that for all m, n, $m \leq \phi(m, n)$ and $n \leq \phi(m, n)$). Pairing functions allow us to code pairs of numbers as individual numbers, since if p is in the range of a pairing function ϕ, then there is exactly one pair (m, n) such that $\phi(m, n) = p$, so the constituents m and n of the pair that p codes are uniquely determined by p alone.

We are interested in finding a pairing function. If we had one, that would show that the theory of recursive functions in two variables essentially reduces to the theory of recursive functions in one variable. This will be because it is easily proved that for all binary relations R, if ϕ is a pairing function, R is recursive (r.e.) iff the *set* $\{\phi(m, n): R(m, n)\}$ is recursive (r.e.). We are going to see that there are indeed pairing functions, so that there is no

essential difference between the theories of recursive binary relations and of recursive sets.

This is in contrast to the situation in the topologies of the real line and the plane. Cantor discovered that there is a one-to-one function from the real line onto the plane. This result was found to be surprising by Cantor himself and by others, since the difference between the line and the plane seemed to lie in the fact that points in the plane could only be specified or uniquely determined by means of pairs of real numbers, and Cantor's result seemed to imply that every point in the plane could be identified by a single real number. But the real line and the plane are topologically distinct, that is, there is no homeomorphism of the real line onto the plane, which means that they are essentially different topological spaces. In fact, Brouwer proved a theorem from which the general result follows that there is no homeomorphism between m-dimensional Euclidean space and n-dimensional Euclidean space (for $m \neq n$).

The following will be our pairing function. Let us define [x, y] to be $(x+y)^2+x$. This function is evidently recursive, since it is limited, as it is defined by the Lim^+ formula $z = (x + y)\cdot(x + y) + x$, and is clearly dominating. Let us show that it is a pairing function, that is, that for all z, if z = [x, y] for some x and y, then x and y are uniquely determined. Let $z = (x+y)^2+x$. $(x+y)^2$ is uniquely determined, and it is the greatest perfect square $\leq z$: if it weren't, then we would have $(x + y + 1)^2 \leq z$, but $(x + y + 1)^2 = (x + y)^2 + 2x + 2y + 1 > (x + y)^2 + x = z$. Let s=x+y, so that $s^2=(x+y)^2$. Since $z>s^2$, we can put $x=z-s^2$, which is uniquely determined, and $y=s-x=s-(z-s^2)$, which is uniquely determined. This completes the proof that [x,y] is a pairing function. Note that it is not onto, i.e. some numbers do not code pairs of numbers. For our purposes this will not matter.

(The earliest mention of this pairing function known to the author is in Goodstein's *Recursive Number Theory*. Several years later, the same function was used by Quine, who probably thought of it independently.)

Our pairing function can be extended to n-place relations. First, note that we can get a recursive tripling function by letting [x, y, z] = [[x, y], z]. We can similarly get a recursive n-tupling function, $[m_1, ..., m_n]$, and we can prove an analogous result to the above in the case of n-place relations: for all n-place relations R, if ϕ is a recursive n-tupling function, R is recursive (r.e.) iff the *set* $\{\phi(m_1,...,m_n): R(m_1,...,m_n)\}$ is recursive (r.e.).

Our pairing function has recursive inverses, i.e. there are recursive functions K_1 and K_2 such that $K_1([x, y]) = x$ and $K_2([x, y]) = y$ for all x and y. When z does not code any pair, we could let K_1 and K_2 be undefined on z; here, however, we let K_1 and K_2 have the value 0 on z. (So we can regard z as coding the pair <0, 0>, though in fact $z \neq [0, 0]$.) Intuitively, K_1 and K_2 are computable functions, and indeed they are recursive. To see this, note that K_1's graph is defined by the formula of Lim $(\exists y \leq z)\ (z = [x, y]) \vee (x = \mathbf{0} \wedge \sim(\exists y \leq z)\ (\exists w \leq z)\ z = [w, y])$; similarly, K_2's graph is defined by the formula of Lim $(\exists x \leq z)\ (z = [x, y]) \vee (y = \mathbf{0} \wedge \sim(\exists x \leq z)\ (\exists w \leq z)\ z = [x, w])$.

Coding Finite Sequences

We have seen that for any n, there is a recursive n-tupling function; or in other words, we have a way of coding finite sequences of fixed length. Furthermore, all these n-tupling functions have recursive inverses. This does not, however, give us a single, one-to-one function for coding finite sequences of arbitrary length. One of the things Cantor showed is that there is a one-to-one correspondence between the natural numbers and the set of finite sequences of natural numbers, so a function with the relevant property does exist. What we need to do, in addition, is to show that an effective way of assigning different numbers to different sequences exists, and such that the decoding of the sequences from their codes can be done also effectively.

A method of coding finite sequences of variable length, due to Gödel, consists in assigning to an n-tuple $<m_1, ..., m_n>$ the number $k=2^{m_1+1}\cdot 3^{m_2+1}\cdot \ldots \cdot p_n^{m_n+1}$ as code (where $p_1=2$ and p_{i+1}=the first prime greater than p_i). It is clear that k can be uniquely decoded, since every number has a unique prime factorization, and intuitively the decoding function is computable. If we had exponentiation as a primitive of RE, it would be quite easy to see that the decoding function is recursive; but we do not have it as a primitive. Although Gödel did not take exponentiation as primitive, he found a trick, using the Chinese Remainder Theorem, for carrying out the above coding with only addition, multiplication and successor as primitive. We could easily have taken exponentiation as a primitive — it is not essential to recursion theory that the language of RE have only successor, addition and multiplication as primitive and other operations as defined. If we had taken it as primitive, our proof of the easy half of Church's thesis, i.e. that all r.e. relations are semi-computable, would still have gone through, since exponentiation is clearly a computable function. Similarly, we could have added to RE new variables to range over finite sets of numbers, or over finite sequences. In fact, doing so might have saved us some time at the beginning of the course. However, it is traditional since Gödel's work to take quantification over numbers, and successor, addition, and multiplication as primitive and to show how to define the other operations in terms of them.

We will use a different procedure for coding finite sequences, the basic idea of which is due to Quine. If you want to code the sequence <5, 4, 7>, why not use the number 547? In general, a sequence of positive integers less than 10 can be coded by the number whose decimal expansion is the sequence. Unfortunately, if you want to code sequences containing numbers larger than or equal to 10, this won't quite work. (Also, if the first element of a sequence $<m_1, ..., m_n>$ is 0, its code will be the same as the code for the sequence $<m_2, ..., m_n>$; this problem is relatively minor compared to the other). Of course, it is always possible to use a larger base; if you use a number to code its base-100 expansion, for example, then you can code sequences of numbers as large as 99. Still, this doesn't provide a single method for coding sequences of arbitrary length.

To get around this, we shall use a modification of Quine's trick, due to the author. The

main idea is to use a variable base, so that a number may code a different sequence to a different base. It also proves convenient in this treatment to use only prime bases. Another feature of our treatment is that we will code finite sets first, rather than finite sequences; this will mean that every finite set will have many different codes (thus, using base 10 only for purposes of motivation, 547 and 745 would code the same set {4, 5, 7}). We will not allow 0 as the first digit of a code (in a base p) of a set, because otherwise 0 would be classified as a member of the set, whether it was in it or not (of course, 0 will be allowed as an intermediate or final digit).

Our basic idea is to let a number n code the set of all the numbers that appear as digits in n's base-p expansion, for appropriate prime p. No single p will do for all sets, since for any prime p, there is a finite set containing numbers larger than p, and which therefore cannot be represented as a base-p numeral. However, in view of a famous theorem due to Euclid, we can get around this.

Theorem (Euclid): There are infinitely many primes.
Proof. Let n be any number, and let's show that there are primes greater than n . n! + 1 is either prime or composite. If it is prime, it is a prime greater than n. If it is composite, then it has some prime factor p; but then p must be greater than n, since n!+1 is not divisible by any prime less than or equal to n. Either way, there is a prime number greater than n; and since n was arbitrary, there are arbitrarily large primes.

So for any finite set S of numbers, we can find a prime p greater than any element of S, and a number n such that the digits of the base-p expansion of n are the elements of S. (To give an example, consider the finite set {1, 2}. This will have as "codes" in base 3 the numbers denoted by '12' and '21' in base 3 notation, that is, 5 and 7; it will have as "codes" in base 5 the numbers 7 and 11, etc.) We can then take [n, p] as a code of the set S (so, in the example, [5,3], [7,3], [7,5] and [11,5] are all codes of {1,2}). In this fashion different finite sets will never be assigned the same code. Further, from a code the numbers n and p are uniquely determined and effectively recoverable, and from n and p the set S is determined uniquely.

We will now show how to carry out our coding scheme in RE. To this effect, we will show that a number of relations are definable in Lim or Lim^+ (and hence not only r.e, but also recursive). Before we begin, let us note that the relation of nonidentity is definable in Lim and in Lim^+, for we can define a formula Less*(x,y) equivalent to the formula Less(x,y) of RE with only bounded quantification: $Less^*(x,y) =_{df.} (\exists z<y)(x+z'=y)$ (an even simpler formula defining the less than relation in Lim and Lim^+ would be $(\exists z<y)(x=z)$). Now, let's put

$$Pr\ (x) =_{df.} x \neq \mathbf{0} \wedge x \neq \mathbf{0}' \wedge (y \leq x)(z \leq x)(M(y,z,x) \supset (y=x \vee z=x)).$$

Pr(x) defines the set of primes in Lim, as is easily seen. We want next to define the relation *w is a power of p*, for prime numbers p. This is done by

$$\text{Ppow}\ (p, w) =_{\text{df.}} \text{Pr}(p) \wedge w \neq \mathbf{0} \wedge (x \leq w)(y \leq w)((M(x,y,w) \wedge \text{Pr}(x)) \supset x = p).$$

Ppow (p, w) says that p is w's only prime factor, and that $w \neq 0$; this only holds if $w = p^k$ for some k and p is prime. Note that if p is not prime, then this trick won't work.

Next, we want to define a formula Digp (m, n, p), which holds iff m is a digit in the base-p expansion of n and p is prime. How might we go about this? Let's use base 10 again for purposes of illustration. Suppose $n > 0$, and let d be any number < 10. If d is the first digit of n's decimal expansion, then $n = d \cdot 10^k + y$, for some k and some $y < 10^k$, and moreover $d \neq 0$. (For example, $4587 = 4 \cdot 10^3 + 587$.) Conversely, if $n = d \cdot 10^k + y$ for some k and some $y < 10^k$ and if $d \neq 0$, then d is the initial digit of the decimal expansion of n. If d is an intermediate or final digit in n's decimal expansion, then $n = x \cdot 10^{k+1} + d \cdot 10^k + y$ for some k, x and y with $y < 10^k$ and $x \neq 0$, and conversely. (This works for final digits because we can always take $y = 0$.) So if $d < 10$ and $n \neq 0$, then d is a digit of n iff d is either an initial digit or an intermediate or final digit, iff there exist x, k, and y with $y < 10^k$ and such that either $d \neq 0$ and $n = d \cdot 10^k + y$, or $x \neq 0$ and $n = x \cdot 10^{k+1} + d \cdot 10^k + y$. If $10 \leq$ d then d is not a digit of n's decimal expansion, and we allow 0 to occur in its own decimal expansion. The restrictions $d \neq 0$ and $x \neq 0$ are necessary, since otherwise 0 would occur in the decimal expansion of every number: $457 = 0 \cdot 10^3 + 457 = 0 \cdot 10^4 + 0 \cdot 10^3 + 457$; and if we want to code any finite sets that do *not* have 0 as an element, we must prevent this. Noting that none of this depends on the fact that the base 10 was used, and finding bounds for our quantifiers, we can define a formula Digp*(m, n, p) in Lim^+, which is true of m,n,p iff m is a digit in the base-p expansion of n and p is prime:

$$\begin{aligned}\text{Digp}^*\ (m, n, p) =_{\text{df.}}\ & \{\, n \neq \mathbf{0} \wedge m < p \wedge \\ & \quad [[m \neq \mathbf{0} \wedge (\exists w \leq n)(\exists z < w)(n = m \cdot w + z \wedge \text{Ppow}\ (p, w))] \vee \\ & \quad (\exists w \leq n)(\exists z_1 \leq n)(\exists z_2 < w)(z_1 \neq \mathbf{0} \wedge n = z_1 \cdot w \cdot p + m \cdot w + z_2 \\ & \qquad \wedge \text{Ppow}\ (p, w))]\} \\ & \vee \\ & \quad (m = \mathbf{0} \wedge n = \mathbf{0} \wedge \text{Pr}(p)).\end{aligned}$$

This formula mirrors the justification given above. However, much of it turns out to be redundant. Specifically, the less complicated formula

$$\begin{aligned}\text{Digp}\ (m, n, p) =_{\text{df.}}\ & (n \neq \mathbf{0} \wedge m < p \wedge \\ & \quad (\exists w \leq n)(\exists z_1 \leq n)(\exists z_2 < w)(n = z_1 \cdot w \cdot p + m \cdot w + z_2 \wedge \\ & \qquad \text{Ppow}\ (p, w))])\end{aligned}$$

$$\vee (m = \mathbf{0} \wedge n = \mathbf{0} \wedge \mathrm{Pr}(p))$$

is equivalent to Digp* (this remark is due to John Barker). To see this, suppose first that Digp*(m,n,p), m<p and n≠0. Then $n = z_1 \cdot p^{k+1} + m \cdot p^k + z_2$ for some k, z_1 and some $z_2 < p^k$. This includes initial digits (let $z_1 = 0$) and final digits (let $z_2 = 0$). So Digp (m, n, p) holds. Conversely, suppose Digp (m, n, p) holds, and assume that m<p and n≠0. Then $n = z_1 \cdot p^{k+1} + m \cdot p^k + z_2$ for some k, z_1 and some $z_2 < p^k$, and moreover $p^k \leq n$. If $z_1 > 0$, then m must be an intermediate or final digit of n, so suppose $z_1 = 0$. Then m > 0: for if m = 0, then $n = 0 \cdot p^{k+1} + 0 \cdot p^k + z_2 = z_2$, but $z_2 < p^k$ and $p^k \leq n$, and so n < n. So m must be the first digit of n.

We can now define

$$x \in y =_{\mathrm{df.}} (\exists n \leq y)(\exists p \leq y)(y = [n, p] \wedge \mathrm{Digp}\ (x, n, p)).$$

$x \in y$ is true of two numbers a,b if b codes a finite set S and a is a member of S. Note that Digp(m,n,p) and $x \in y$ are formulae of Lim^+. We could have carried out the construction in Lim, but it would have been more tedious, and would not have had any particular advantage for the purposes of this course.

There are two special cases we should check to make sure our coding scheme works: namely, we should make sure that the sets {0} and Ø have codes. If y is not in the range of our pairing function, then $x \in y$ will be false for all x; so y will code Ø. And since Digp(0, 0, p) holds for any p, [0, p] codes the set {0}.

Lecture IV

Let us now note a few bounding tricks that will be useful in the future. The function $z = [x, y]$ is monotone in both variables: i.e. if $x \leq x_1$ and $y \leq y_1$ then $[x, y] \leq [x_1, y_1]$. Moreover, $x, y \leq [x, y]$. Finally, if n codes a set S, and $x \in S$, then $x \leq n$: if n codes S, then n is [k, p] for some k and p, so $k \leq n$; and x is a digit in k's base-p expansion, so $x \leq k$. So we can introduce some new bounded quantifiers into Lim^+:

$$(x \in y)\, B =_{df.} (x \leq y)\, (x \in y \supset B);$$
$$(\exists x \in y)\, B =_{df.} (\exists x \leq y)\, (x \in y \wedge B).$$

Note also that if n codes a set S and $S' \subseteq S$, then there is an $m \leq n$ which codes S'. (This is because, if the elements of S are the digits of the base-p expansion of k, then there is a number $j \leq k$ such that the digits in j's base-p expansion are the elements of S'; since $j \leq k$, $[j, p] \leq [k, p]$ and [j, p] codes S'.) We can therefore define

$$x \subseteq y =_{df.} (z \in x)\, z \in y;$$
$$(x \subseteq y)\, B =_{df.} (x \leq y)\, (x \subseteq y \supset B);$$
$$(\exists x \subseteq y)\, B =_{df.} (\exists x \leq y)\, (x \subseteq y \wedge B).$$

Now that we can code finite sets of numbers, it is easy to code finite sequences. For a sequence $\langle m_1, ..., m_n \rangle$ is simply a function ϕ with domain $\{1, ..., n\}$ and with $\phi(i) = m_i$; we can identify functions with their graphs, which are relations, i.e. sets of ordered pairs, which we can in turn identify with sets of numbers, since we can code up ordered pairs as numbers. (So, for example, we can identify the sequence $\langle 7, 5, 10 \rangle$ with the set $\{[1, 7], [2, 5], [3, 10]\}$.) Finally, those sets can themselves be coded up as numbers. We define a formula Seql (s, n) of Lim^+ which holds just in case s codes a sequence of length n:

$$\begin{aligned}\text{Seql}\,(s, n) =_{df.}\ & (x \in s)(\exists m_1 \leq s)(\exists m_2 \leq s)(x = [m_1, m_2] \wedge m_1 \neq \mathbf{0} \wedge m_1 \leq n) \wedge \\ & (m_1 \leq s)(m_2 \leq s)(m_3 \leq s)(([m_1, m_2] \in s \wedge [m_1, m_3] \in s) \supset m_2 = m_3) \\ & \wedge (m_1 \leq n)(\exists m_2 \leq s)(m_1 \neq \mathbf{0} \supset [m_1, m_2] \in s).\end{aligned}$$

The first conjunct simply says that every element of s is a pair whose first member is a positive integer $\leq n$; the second says that s is single valued, i.e. is (the graph of) a function; and the third says that every positive integer $\leq n$ is in s's domain.

We can also define a formula Seq (s), which says that s codes a finite sequence of some length or other:

$$Seq(s) =_{df.} (\exists n \leq s)\ Seql\ (s, n).$$

We can bound the initial quantifier, because if s codes a sequence of length n, then $[n, x] \in$ s for some x, and so $n \leq [n, x] \leq s$. Also, if x is the ith element of some sequence s, then $x \leq [i, x] \leq s$; we can use this fact to find bounds for quantifiers.

The following formula holds of two numbers if the second codes a sequence and the first occurs in that sequence:

$$x \text{ on } s =_{df.} Seq(s) \wedge (\exists y \leq s)\ ([y,x] \in s).$$

Gödel Numbering

We can use our method of coding up finite sequences of numbers to code up finite strings of symbols. As long as we have a countable alphabet, we will be able to find a 1-1 correspondence between our primitive symbols and the natural numbers; we can thus code up our primitive symbols as numbers. We can then identify strings of symbols with sequences of numbers, which we then identify with individual numbers. A scheme for coding strings of symbols numerically is called a *Gödel numbering*, and a numerical code for a symbol or expression is called a *Gödel number* for it.

Exactly how we do this is arbitrary. One way of doing it is this: if $S = s_1...s_n$ is a string of symbols, and $a_1, ..., a_n$ are the numerical codes for $s_1, ..., s_n$, then $<a_1, ..., a_n>$ is a sequence of numbers, and it therefore has a code number p; we can take p to be a Gödel number of S. (Note that, on our way of coding finite sequences, each sequence will have many different code numbers, so we must say "*a* Gödel number" rather than "*the* Gödel number.") Call this the *simple-minded* coding scheme.

We shall adopt a slightly more complicated coding scheme, which will make things easier later on. First, we code the terms of the language via the simple-minded scheme. Then, when coding formulae, we again use as a code for a string of symbols a code for the corresponding sequence of codes of symbols, except that now we treat terms as single symbols. So if a, b, c, d are the codes of the primitive symbols P_1^1, f_1^2, x_1, x_2, then any code p for <b, c, d> is a code for the term $f_1^2x_1x_2$, and any code for <a, p> codes $P_1^1f_1^2x_1x_2$.

We want a single coding scheme for all the languages we shall consider, namely, the various first-order languages and the languages RE and Lim (and its variants). So we shall need to take all of the symbols (,), $\supset$, $\sim$, $\wedge$, $\vee$, $<$, and $\exists$ as primitive, and provide code numbers for all of them. We also need code numbers for the constants, variables, predicates, and function letters. Our general scheme for doing this is to code a symbol s by a pair [x, y], where x represents s's grammatical category, and y represents additional information about s (e.g. its sub- and superscript). For definiteness, we make the following

our official Gödel numbering:

Individual symbols:	(	)	$\exists$	<	$\supset$	~	$\wedge$	$\vee$
	[0, 0]	[0, 1]	[0, 2]	[0, 3]	[0, 4]	[0, 5]	[0, 6]	[0, 7]

Variables: [1, i] codes x_i

Constants: [2, i] codes a_i

(Special constants,
or "choice" constants: [3, i] codes b_i)

Function letters: [4, [n, i]] codes f_i^n

Predicate letters: [5, [n, i]] codes P_i^n

(We do not have special constants in the languages we have developed so far; but in case we need them, we have codes for them.) Note that this coding scheme is open-ended; we could add extra individual symbols, or even extra grammatical categories (e.g. new styles of variables), without disruption.

Identification

Strictly speaking, when we use an entity A to code an entity B, A and B are (in general) different entities. However, we often speak as though they were the same; for example, we say that the number 105 = [5, [1, 1]] *is* the symbol P_1^1, whereas strictly speaking we should say that it *codes* P_1^1. (Similarly, we will say, for example, that a certain predicate is true of exactly the formulae, or of exactly the terms, where we should say that it is true of the codes of formulae, or of the codes of terms). This has the problem that, since we have many different codes for a single expression, many different numbers are identified with the same expression. In order to avoid this talk of identification, we might modify our coding scheme so as to make the coding correspondence one-to-one, for example taking the least number among the codes to be the real code.

According to Geach's doctrine of relative identity, this talk of identification would be not only harmless, but absolutely legitimate. For Geach, it does not make sense to say simply that two objects are the same, this being only a disguised way of saying that they are the same F, for some property F. In this sense there is no such thing as absolute identity, according to Geach. His doctrine of relative identity would then allow us to say that although two objects are different numbers, they are the same formula. The author does not

share Geach's views on this point, but it is useful to think in terms of relative identity in our context. Geach has applied his doctrine in other contexts.

The Generated Sets Theorem.

We shall now use our coding of finite sequences to show that some intuitively computable functions which are not obviously recursive are in fact recursive. Let's start with the factorial function y = x!. Note that 0! = 1 and (n+1)! = (n+1)·n! for all n, and that this is an inductive definition that specifies the function uniquely. The sequence <0!, ..., n!> is therefore the unique sequence $<x_1, ..., x_{n+1}>$ such that $x_1 = 0$ and for all $k \leq n$, $x_{k+1} = (k+1)\cdot x_k$. Thus, y = x! just in case y is the x+1st member of some such sequence. So the following formula of RE defines the graph of the factorial function:

$$(\exists s)(\text{Seql}\ (s, x') \wedge [\mathbf{0}', [\mathbf{0}, \mathbf{0}']] \in s \wedge (z \leq s)(i \leq x')([i'', [i', z]] \in s \supset (\exists z_1 \leq s)\ ([i',[i,z_1]] \in s \wedge z = z_1 \cdot i')) \wedge [x', [x, y]] \in s).$$

(Note that we could have written $\mathbf{0}' \in s$ instead of $[\mathbf{0}', [\mathbf{0}, \mathbf{0}']] \in s$, since $[1, [0, 1]] = (1+((0+1)^2+0))^2 + 1 = 5$. Note also that, while $\supset$ is not definable in RE, its use in this formula is permissible, since its antecedent, $[i'', [i', z]] \in s$, expresses a relation whose complement is r.e. Also, the part of the formula following the initial unbounded quantifier $(\exists s)$ is a formula of Lim^+ (in which $\supset$ is definable), and is therefore equivalent to a formula of RE, and so the entire formula is a formula of RE.)

The above definition of y = x! is an example of a definition by primitive recursion; we have a *base clause*

$$0! = 1$$

in which the function's value at zero is specified, and an *induction clause*

$$(n+1)! = (n+1)(n!)$$

in which the value at n+1 is specified in terms of its value at n. Another example of this kind of definition is that of the exponentiation function $z = x^y$: we stipulate that $x^0 = 1$ and $x^{y+1} = x^y \cdot x$. Here, the induction is carried out on the variable y; however the value of the function also depends on x, which is kept fixed while y varies. x is called a *parameter*; the primitive recursive definition of exponentiation is called a primitive recursive definition *with parameters*, and that of the factorial function is said to be *parameter free*. We can show that the exponentiation function is recursive, using a similar argument to the above.

In general, if h is an n-1-place function and g is an n+1-place function, then the n-place

function f is said to come from g and h by *primitive recursion* if f is the unique function such that

$$f(0, x_2, ..., x_n) = h(x_2, ... x_n)$$

and

$$f(x_1+1, x_2, ..., x_n) = g(x_2, ..., x_n, x_1, f(x_1, x_2, ..., x_n))$$

for all $x_1, ..., x_n$. (Here we take 0-place functions to be constants, i.e. when n = 1, we let h be a number and let f(0) = h.) We define the class of primitive recursive functions inductively, as follows. (i) The basic primitive recursive functions are the zero function z(x) = 0, the successor function s(x) = x+1, and the identity functions $id^n_i(x_1, ..., x_n) = x_i$ (where $i \leq n$). (ii) The composition of primitive recursive functions is primitive recursive (that is, if $\psi(m_1,...,m_k)$ is a primitive recursive function in k variables, and $\phi_1(q_{1,1},...,q_{1,n_1}),...,$ $\phi_k(q_{k,1},...,q_{k,n_k})$ are k primitive recursive functions in $n_1,...,n_k$ variables, respectively, then so is the function in $n_1+...+n_k$ variables $\psi(\phi_1(q_{1,1},...,q_{1,n_1}),..., \phi_k(q_{k,1},...,q_{k,n_k}))$. (iii) A function that comes from primitive recursive functions by primitive recursion is primitive recursive. (iv) And the primitive recursive functions are only those things required to be so by the preceding. Using the same sort of argument given in the case of the exponentiation function, we can show that all primitive recursive functions are recursive. (That the recursive functions are closed under primitive recursion is called the *primitive recursion theorem*.)

The converse, however, does not hold. Consider the sequence of functions

$\psi_1(x, y) = x + y$
$\psi_2(x, y) = x \cdot y$
$\psi_3(x, y) = x^y$

This sequence can be extended in a natural way. Just as multiplication is iterated addition and exponentiation is iterated multiplication, we can iterate exponentiation: let $\psi_4(x, 0) = x$, $\psi_4(x, 1) = x^x$, $\psi_4(x, 2) = x^{x^x}$, etc. This function is called *superexponentiation*. We can also iterate superexponentiation, giving us a super-superexponentiation function, and so on. In general, for n > 2, we define

$\psi_{n+1}(x, 0) = x$
$\psi_{n+1}(x, y+1) = \psi_n(x, \psi_{n+1}(x, y))$

We can turn this sequence of 2-place functions into a single 3-place function by letting $\chi(n, x, y) = \psi_n(x, y)$; χ is called the *Ackermann function*. Ackermann showed that this function is not primitive recursive, though it is clearly computable. (This is the function that we

referred to earlier.) In fact, it can be shown that for any 1-place primitive recursive function ϕ, $\phi(x) < \chi(x, x, x)$ for all but finitely many x.

We shall next prove a theorem from which it follows that a wide range of functions, including both the primitive recursive functions and the Ackermann function, are recursive. This theorem will also be useful in showing that various interesting sets and relations are r.e. The theorem will further provide a way of making rigorous the extremal clauses in our earlier inductive definitions of term and formula of the different languages that we have introduced.

The basic idea that motivates the theorem is best illustrated by means of a definition formally similar to those of formula or term, that of a *theorem* of a *formal system.* In a formal system, certain strings of formulae are called axioms, and from them the theorems of the formal system are generated by means of certain rules of inference (for example, *modus ponens*, according to which if formulae A and $(A \supset B)$ are theorems, then B is a theorem). The notion of a theorem is defined inductively, specifying that all the axioms are theorems (basis clauses), that if a formula A follows from theorems $B_1, ..., B_n$ by one of the inference rules, then A is also a theorem (closure conditions, or generating clauses), and that the theorems are only those things generated in this way (extremal clause).

In a formal system a formula is a theorem if it has a proof. And a proof is a finite sequence of formulae each of which is either an axiom or a formula which comes from previous formulae in the sequence via one of the generating clauses (the inference rules). Sequences which are proofs are called proof sequences. We can generalize the notion of a proof sequence so as to apply it to the case of terms or formulae. Something is a formula if it occurs on a sequence each element of which is either an atomic formula or comes from previous formulae in the sequence via one of the generating clauses (the rules for the formation of complex formulae out of simpler ones). One such sequence can be seen as a proof that a string of symbols is a formula, which justifies using the phrase 'proof sequence' in this case as well. (Similar remarks could be made about the notion of a term).

Generalizing this, we introduce the following

Definition: A *proof sequence* for a set B, and relations $R_1, ..., R_k$ (n_1+1-place,..., n_k+1-place, respectively) is a finite sequence $<x_1, ..., x_p>$ such that, for all $i = 1, ..., p$, either $x_i \in B$ or there exist $j \leq k$ and $m_1, ..., m_{n_j} < i$ such that $R_j(x_{m_1}, ..., x_{m_{n_j}}, x_i)$.

Our extremal clauses will be understood as formulated with the help of the notion of a proof sequence determined by the appropriate sets and relations. And our proofs by induction on the complexity of terms or formulae would proceed rigorously speaking by induction on the length of the appropriate proof sequences.

If we have a set B and some relations $R_1, ..., R_k$, where each R_i is an n_i+1-place relation, the set *generated* by B and $R_1, ..., R_k$ is the set of those objects which occur in some proof sequence for B and $R_1, ..., R_k$. If S is the set generated by B and $R_1, ..., R_k$, we call B the

basis set for S and R_1, ..., R_k the *generating relations* for S.

Generated Sets Theorem: If B is an r.e. set and R_1, ..., R_k are r.e. relations, then the set generated by B and R_1, ..., R_k is itself r.e.
Proof. Let C be a formula of RE that defines the set B, and let F_1, ..., F_k be formulae of RE that define R_1, ..., R_k. We first define

$$\text{PfSeq}(s) =_{\text{df.}} \text{Seq}(s) \wedge (m \leq s)(x < s)([m,x] \in s \supset C(x) \vee \text{(clause 1)} \vee \ldots \vee \text{(clause k)}),$$

where (clause j) is the formula

$$(\exists x_1 \leq s)\ldots(\exists x_{n_j} \leq s)(\exists i_1 < i)\ldots(\exists i_{n_j} < i)([i_1, x_1] \in s \wedge \ldots \wedge [i_{n_j}, x_{n_j}] \in s \wedge F_j(x_1, \ldots, x_{n_j}, y_1).$$

PfSeq(s) thus defines the set {s: s codes a proof sequence for B and R_1, ..., R_k}. We can therefore define the set G generated by B and R_1, ..., R_k by means of the formula of RE

$$(\exists s)(\text{PfSeq}(s) \wedge (\exists m \leq s)([m, x] \in s).$$

This completes the proof.

The generated sets theorem applies in the first instance to sets of numbers; but it also applies derivatively to things that can be coded up as sets of numbers, e.g. sets of formulae. Suppose some set G of formulae is the set generated by a basis set B of formulae and generating rules R_1, ..., R_k among formulae. To show that the set G' of Gödel numbers of elements of G is r.e., simply show that the set B' of Gödel numbers of elements of B is r.e. and that the relations R_i' among Gödel numbers for formulae related by the relations R_i are r.e. (Of course, whether G' is in fact r.e. will depend on what the relations B and R_1, ..., R_k are.) In this way, it is easy to show that the set of formulae of RE is itself r.e.

The Generated Sets Theorem is known to all logicians, although it is rarely stated explicitly. It provides a simpler method of proving that some sets or relations are r.e. (and hence that some total functions are recursive) than primitive recursion. Of course, it does not provide a general method of proving recursiveness, but it is infrequent in mathematical arguments to have the need to show that a set or relation is recursive besides being recursively enumerable. It is usually emphasized as a basic requirement of logic that the set of formulae of a given language must be decidable, but it is not clear what the theoretical importance of such a requirement is. Chomsky's approach to natural language, for example, does not presuppose such a requirement. In Chomsky's view, a grammar for a language is specified by some set of rules for generating the grammatically correct sentences of a

language, rather than by a decision procedure for grammatical correctness.

However, we will eventually state a theorem an application of which will be to show that the set of codes of formulae or terms of a language is recursive.

We can use the generated sets theorem to show that a function is recursive. For example, the function y = x! is recursive iff the set {[x, x!] : x ∈ **N**} is r.e., and this set can be generated as follows: let the basis set be {[0, 1]}, and let the generating relation be {<[x, y], [x+1, y·(x+1)]>: x, y ∈ **N**}. It is easy to see that the basis set and generating relation are r.e (and indeed recursive), and that they generate the desired set. In fact, the result that all primitive recursive functions are recursive follows directly from the generated sets theorem in this way. Moreover, the generated sets theorem can be used to show that the Ackermann function is recursive. This is the virtue of the generated sets theorem: it is more powerful than the theorem about primitive recursiveness, and indeed it is easier to prove that theorem via the generated sets theorem than directly.

We may sometimes want to know that a set G is recursive, or even limited, in addition to being r.e. While the generated sets theorem only shows that G is r.e., in particular cases we can sometimes sharpen the result. For one thing, if the basis set and generating relations are recursive (or limited), then the formula PfSeq(s) defines a recursive (limited) relation. This does not itself show that G is recursive (limited), since the formula used to define G in the proof of the Generated Sets Theorem begins with the unbounded quantifier (∃s). If we can find some way of bounding this quantifier, then we can show that G is recursive (or limited). However, it is not always possible to bound this quantifier, for not all sets generated from a recursive basis set via recursive generating relations are recursive. For example, the set of Gödel numbers of valid sentences of the first-order language of arithmetic is r.e., but not recursive; and yet that set is clearly generated from a recursive basis set (the axioms) and recursive generating relations (the inference rules).

Exercises

1. a) Prove that every k-place constant function is recursive. Prove that the successor function is recursive.
b) Prove that if a function $\phi(m_1,...,m_k)$ in k variables is recursive (partial recursive), so is any k-1 place function obtained from ϕ by identifying two variables.

2. a) Prove that the composition of two 1-place total (partial) recursive functions is total (partial) recursive.
b) More generally, prove that if $\psi(m_1,...,m_k)$ is a total (partial) recursive function in k variables, and $\phi_1(q_{1,1},...,q_{1,n_1}),..., \phi_k(q_{k,1},...,q_{k,n_k})$ are k total (partial) recursive functions in $n_1,...,n_k$ variables, respectively, then so is the function in $n_1+...+n_k$ variables

$\psi(\phi 1(q_{1,1},...,q_{1,n1}),..., \phi k(q_{k,1},...,q_{k,nk}))$.

3. Show that if ϕ is a recursive pairing function whose range is recursive, then a binary relation R is recursive iff the set $\{\phi(m,n): R(m,n)\}$ is recursive. Prove that a sufficient condition for the range of a recursive pairing function ϕ to be recursive is that $m,n \leq \phi(m,n)$. (This condition is satisfied by the pairing function we have been using and by nearly all the pairing functions used in practice). Where does the argument go wrong if we do not assume that the range is recursive? (a counterexample will be given later.)

4. For arbitrary $n > 1$, define an n-tupling function, verifying that it is indeed an n-tupling function. Generalize exercise 3 to arbitrary n-place relations accordingly.

Lecture V

Truth and Satisfaction in RE.

Remember that the satisfaction relation is a relation in two variables, S(A,s), which holds between a formula A and an assignment s sufficient for A just in case s satisfies A (in the case of RE, s assigns non-negative integers to the variables, since the intended interpretation of RE is the arithmetical interpretation). Since truth can be defined in terms of satisfaction, if RE could define its own satisfaction relation, RE would have its own truth predicate.

Some assignments related to formulae by the satisfaction relation are sequences of infinite length: the sequence $\{<x_1,0>, <x_2,1>, ...\}$ is an assignment of the value i-1 to the variable x_i; this assignment is naturally sufficient for any formula, and satisfies, e.g., all formulae of the form $x_i=x_i$. However, as Cantor showed, we could not code all infinite sequences of numbers by means of numbers, so the satisfaction relation for formulae of RE cannot be represented as a relation between numbers. However, for our purposes it is really unnecessary to contemplate the full satisfaction relation. It will be enough to be able to define within RE the satisfaction relation restricted to finite assignments, or even a relation Sat(a,s), which holds between a (code of a) formula A and a (code of a) finite function s which assigns non-negative integers to all the (codes of) variables appearing free in A and satisfies A in the obvious sense (thus, if Sat(a,s), s need not be a sequence, for its domain need not be an initial segment of the non-negative integers -nor an initial segment of the codes of variables-, and s need not be an assignment, for it can assign values to things other than codes of variables). Sat(a,s) will be the relation that we will show how to define in RE. In fact, we shall show, equivalently, that the set of Gödel numbers of pairs in Sat is r.e., using the Generated Sets Theorem. One way in which we can begin to see that this will be enough for our purposes is to note that if Sat can be defined in RE, then the truth predicate for RE can be defined in RE, since a sentence of RE is true just in case it is satisfied by some finite function.

We shall now undertake the proof of the following

Theorem: The satisfaction relation Sat(a,s) for formulae of RE is definable in RE, or, in other words, RE has its own satisfaction predicate.

We shall devote to this proof this lecture and the next one.

As we just said, in showing that the satisfaction relation for RE is r.e., we shall use the Generated Sets Theorem. What we shall show is that the *set* of (numbers coding) pairs G={[a, s]: s codes a function which is sufficient for and satisfies the formula whose Gödel number is a} is generated from an r.e. set by means of r.e. relations.

In order to prove our theorem it would be perhaps most natural to generate separately the set of formulae, then define in some way the notion of a function being sufficient for a formula, and finally generate the set of pairs <A,s> where s is sufficient for and satisfies the formula A, going through the clauses in the inductive definition of satisfaction for RE. However, we will generate this set in one fell swoop, so to speak, without having first to define the set of formulae and the relation of being a function sufficient for a formula.

We will now begin our specification of the basis set, and later we will define the generating relations. In order to show that the set that we will take as basis set is r.e., we will show first that the set of terms and the relation of denotation are r.e.

It is important to stress at this stage a delicate point in our coding scheme. Remember that in our coding scheme, in order to code formulae we code terms first, by coding the sequence of numbers that code the individual symbols appearing in a term (in the same order). Thus, a term $f_1^1 f_1^1 x_i$ will be coded by any code of the sequence {[1,[4,[1,1]]], [2,[4,[1,1]]], [3,[1,i]]}, and, *as a term*, x_i will be coded by any code of the sequence {[1,[1,i]]}. Then we code formulae using the same procedure, but now taking each term as if it was an individual symbol, a code for it being a code of the appropriate sequence. Thus, a formula $P_1^2 x_i x_i$ will be coded by any code of the sequence {[1,[5,[2,1]]], [2,[1,[1,i]]], [3,[1,[1,i]]]}.

We now exhibit a formula Funct(s) which is true of a number if it codes a finite function:

$$\text{Funct}(s) =_{\text{df.}} (x \in s)(\exists m_1 \leq x)(\exists m_2 \leq x)(x = [m_1, m_2]) \wedge (n_1 \leq x)(n_2 \leq x)(m \leq s)(([m, n_1] \in s \wedge [m, n_2] \in s) \supset n_1 = n_2).$$

With the help of Funct(s), we can give an alternative formula that shows that the relation holding between a sequence and its length is r.e.:

$$\text{Seql}(s,n) =_{\text{df.}} \text{Funct}(s) \wedge (i \leq n)(\mathbf{0}' \leq i \supset (\exists j)([i,j] \in s)).$$

We now specify a formula Num(m,n) which is true of a pair of numbers p,q if p is a code of a numeral that denotes number q:

$$\text{Num}(m,n) =_{\text{df.}} \text{Seql}(m,n') \wedge [n', [\mathbf{0}^{(2)}, \mathbf{0}^{(1)}]] \in m \wedge (i \leq n)(\mathbf{0}' \leq i \supset [i, [\mathbf{0}^{(4)}, [\mathbf{0}^{(1)}, \mathbf{0}^{(1)}]]] \in m).$$

The first conjunct “says” that m is a sequence of length n+1; the second that the last pair of the sequence has as second element the code of the constant **0**, which, remember, is [2,1]; and the third conjunct “says” that all the other second elements of the sequence are codes of the symbol for successor, which is [4,[1,1]]. We can now give a formula Numeral(m) that defines the set of codes of numerals:

$$\text{Numeral}(m) =_{\text{df.}} (\exists n \leq m)(\text{Num}(m,n)).$$

The formula Vblt(m,v) will be true of two numbers p,q if p is a code of a term of the form $f_1^1 \ldots f_1^1 x_i$, for a certain i, and v is the code of x_i (in this case we say that p is (a code of) a variable term ending in variable q):

$$\text{Vblt}(m,v) =_{\text{df.}} (\exists n \leq m)(\text{Seql}(m,n') \wedge (\exists i \leq m)(v = [\mathbf{0}^{(1)}, i]) \wedge [n', v] \in m \wedge (j \leq n)(\mathbf{0}' \leq j \supset [j, [\mathbf{0}^{(4)}, [\mathbf{0}^{(1)}, \mathbf{0}^{(1)}]]] \in m)).$$

The first conjunct "says" that m is a sequence of length n+1; the second that v is the code of a variable; the third that the last value of the sequence is v; the fourth that all the values preceding v are codes of the symbol for successor. A formula similar to Numeral(m) then defines the set of codes of terms of the form $f_1^1 \ldots f_1^1 x_i$:

$$\text{Vblterm}(m) =_{\text{df.}} (\exists v \leq m)(\text{Vblt}(m,v)).$$

It will be useful to introduce a formula Vbl(v) which is true of a number if it is the code of a variable:

$$\text{Vbl}(v) =_{\text{df.}} (\exists i \leq v)(v = [\mathbf{0}^{(1)}, i]).$$

Finally, we can give a formula that defines the set of codes of terms:

$$\text{Term}(m) =_{\text{df.}} \text{Numeral}(m) \vee \text{Vblterm}(m)$$

(remember that in RE the only terms are numerals (in official notation) and variables preceded by a number of occurrences of the function letter f_1^1. If we had taken + and · as primitive function letters, there would have been more complicated terms. As it is, since ' is our only function symbol, things are much simpler).

We are now ready to define denotation. The formula Den(m,n,s) is true of a triple of numbers p,q,r if p is a term that denotes q with respect to assignment r:

$$\text{Den}(m,n,s) =_{\text{df.}} \text{Funct}(s) \wedge (\text{Num}(m,n) \vee (\exists v \leq m)(\text{Vblt}(m,v) \wedge (\exists p \leq m)(\text{Seql}(m,p')) \wedge (\exists q \leq s)([v,q] \in s \wedge q + p = n)).$$

The second disjunct of the second conjunct "says" that there is a variable v such that m is a variable term ending in v, m has length p+1 for a certain p and s assigns to v a number q such that, if you add to it 1 p times, you get n.

The formula $\text{Atf}^{=}(s)$ is true of a number if it codes an atomic formula of the form $P_1^2 t_1 t_2$, where t_1 and t_2 are terms of RE:

$$\mathrm{Atf}^{=}(s) =_{\mathrm{df.}} (\exists m_1 \leq s)(\exists m_2 \leq s)(\mathrm{Seql}(s, \mathbf{0}^{(3)}) \wedge \mathrm{Term}(m_1) \wedge \mathrm{Term}(m_2) \wedge [\mathbf{0}^{(1)}, [\mathbf{0}^{(5)}, [\mathbf{0}^{(2)}, \mathbf{0}^{(1)}]]] \in s \wedge [\mathbf{0}^{(2)}, m_1] \in s \wedge [\mathbf{0}^{(3)}, m_2] \in s).$$

The formula $\mathrm{Atf}^{A}(s)$ is true of a number if it codes an atomic formula of the form $P_1^3 t_1 t_2 t_3$, where t_1, t_2 and t_3 are terms of RE:

$$\mathrm{Atf}^{A}(s) =_{\mathrm{df.}} (\exists m_1 \leq s)(\exists m_2 \leq s)(\exists m_3 \leq s)(\mathrm{Seql}(s, \mathbf{0}^{(4)}) \wedge \mathrm{Term}(m_1) \wedge \mathrm{Term}(m_2) \wedge \mathrm{Term}(m_3) \wedge [\mathbf{0}^{(1)}, [\mathbf{0}^{(5)}, [\mathbf{0}^{(3)}, \mathbf{0}^{(1)}]]] \in s \wedge [\mathbf{0}^{(2)}, m_1] \in s \wedge [\mathbf{0}^{(3)}, m_2] \in s \wedge [\mathbf{0}^{(4)}, m_3] \in s).$$

The formula $\mathrm{Atf}^{M}(s)$ is true of a number if it codes an atomic formula of the form $P_2^3 t_1 t_2 t_3$, where t_1, t_2 and t_3 are terms of RE:

$$\mathrm{Atf}^{M}(s) =_{\mathrm{df.}} (\exists m_1 \leq s)(\exists m_2 \leq s)(\exists m_3 \leq s)(\mathrm{Seql}(s, \mathbf{0}^{(4)}) \wedge \mathrm{Term}(m_1) \wedge \mathrm{Term}(m_2) \wedge \mathrm{Term}(m_3) \wedge [\mathbf{0}^{(1)}, [\mathbf{0}^{(5)}, [\mathbf{0}^{(3)}, \mathbf{0}^{(2)}]]] \in s \wedge [\mathbf{0}^{(2)}, m_1] \in s \wedge [\mathbf{0}^{(3)}, m_2] \in s \wedge [\mathbf{0}^{(4)}, m_3] \in s).$$

Then the formula Atfmla(s) is true of a number if it codes an atomic formula:

$$\mathrm{Atfmla}(s) =_{\mathrm{df.}} \mathrm{Atf}^{=}(s) \vee \mathrm{Atf}^{A}(s) \vee \mathrm{Atf}^{M}(s).$$

Lecture VI

Truth and Satisfaction in RE (Continued).

We are getting closer to specifying the basis set. This set (let's call it B) will include the set of numbers [a,s] where a is a code of an atomic formula A and s is a function that is sufficient for A and satisfies A. B will include other (numbers coding) pairs of formulae and functions as well, as we will see later, but we can now start the construction of the formula that defines B by exhibiting the disjuncts of that formula that "correspond" to the cases in which the number a in [a,s] codes an atomic formula.

The first disjunct will be a formula true of two numbers a,s if a is an atomic formula of the form $P^2_1t_1t_2$ and s is sufficient for and satisfies a:

$$D_1(a,s)=_{df.} (\exists m_1\leq a)(\exists m_2\leq a)(Seql(a,\mathbf{0}^{(3)}) \wedge Term(m_1) \wedge Term(m_2) \wedge [\mathbf{0}^{(1)},[\mathbf{0}^{(5)},[\mathbf{0}^{(2)},\mathbf{0}^{(1)}]]]\in a \wedge [\mathbf{0}^{(2)},m_1]\in a \wedge [\mathbf{0}^{(3)},m_2]\in a \wedge (\exists y_1\leq a)(\exists y_2\leq a)(Den(m_1,y_1,s) \wedge Den(m_2,y_2,s) \wedge y_1=y_2).$$

Notice that we use the identity predicate of RE to define the relation of satisfaction restricted to codes of equalities and functions that satisfy them. Below, the predicates of addition and multiplication of RE are used analogously, and so will be the connectives and quantifiers in our definitions of the generating relations for complex formulae. This procedure for defining satisfaction, and hence truth, was first used by Tarski. In the case of a sentence, like **0**=**0**, Tarski's definition of truth comes down to the biconditional: **0**=**0** is true iff 0=0. Tarski's definition appeared when some logical positivists had expressed doubts about the possibility of a scientifically acceptable definition or theory of truth. Tarski showed that this way of defining satisfaction and hence truth existed, and that it had important uses in logic and mathematics.

The second disjunct will be a formula true of two numbers a,s if a is an atomic formula of the form $P^3_1t_1t_2t_3$, where t_1, t_2 and t_3 are terms of RE, and s is sufficient for and satisfies a:

$$D_2(a,s)=_{df.} (\exists m_1\leq a)(\exists m_2\leq a)(\exists m_3\leq a)(Seql(s,\mathbf{0}^{(4)}) \wedge Term(m_1) \wedge Term(m_2) \wedge Term(m_3) \wedge [\mathbf{0}^{(1)},[\mathbf{0}^{(5)},[\mathbf{0}^{(3)},\mathbf{0}^{(1)}]]]\in s \wedge [\mathbf{0}^{(2)},m_1]\in s \wedge [\mathbf{0}^{(3)},m_2]\in s \wedge [\mathbf{0}^{(4)},m_3]\in s \wedge (\exists y_1\leq a)(\exists y_2\leq a)(\exists y_3\leq a)(Den(m_1,y_1,s) \wedge Den(m_2,y_2,s) \wedge Den(m_3,y_3,s) \wedge A(y_1,y_2,y_3)).$$

The third disjunct will be a formula true of two numbers a,s if a is an atomic formula of the form $P^3_2t_1t_2t_3$, where t_1, t_2 and t_3 are terms of RE, and s is sufficient for and satisfies a:

$$D_3(a,s)=_{df.} (\exists m_1\leq s)(\exists m_2\leq s)(\exists m_3\leq s)(Seql(s,\mathbf{0}^{(4)}) \wedge Term(m_1) \wedge Term(m_2) \wedge$$

$$\text{Term}(m_3) \wedge [\mathbf{0}^{(1)},[\mathbf{0}^{(5)},[\mathbf{0}^{(3)},\mathbf{0}^{(2)}]]] \in s \wedge [\mathbf{0}^{(2)},m_1] \in s \wedge [\mathbf{0}^{(3)},m_2] \in s \wedge [\mathbf{0}^{(4)},m_3] \in s \wedge (\exists y_1 \leq a)(\exists y_2 \leq a)(\exists y_3 \leq a)(\text{Den}(m_1,y_1,s) \wedge \text{Den}(m_2,y_2,s) \wedge \text{Den}(m_3,y_3,s) \wedge M(y_1,y_2,y_3)).$$

Besides these three kinds of pairs [a,s], the basis set B will contain a fourth kind of pairs. The reason for this is that we are trying to avoid having to define the set of formulae of RE and the relation of a function being sufficient for a formula of RE. We are going to give generating relations that will generate the set {[a,s]: Sat(a,s)} in one fell swoop, as we said. But in order to do this, we need some way of dealing with the natural generating rule for disjunctions: if s satisfies A, B is a formula and s is sufficient for B, then s satisfies (A∨B). How are we going to define the relation corresponding to this rule without having defined the notions of being a formula and of being a function sufficient for a formula? We can appeal only to the notion of satisfaction as holding between less complex formulae and functions sufficient for them. Clearly, defining the relation corresponding to the following rule will not do: if s satisfies A and s satisfies B, then s satisfies (A∨B). It will not do because B may be unsatisfiable, in which case (A∨B) will not be generated.

To get around this we will use the following observation. All formulae of the form $(x_i<\mathbf{0})$B are satisfied by all functions sufficient for them, since there is no number less than 0, so all pairs [a,s] where a is a formula of that form and s is sufficient for B must be in our final set. But if we have already generated the pairs consisting of a formula of the form $(x_i<\mathbf{0})$B and all the functions s that satisfy it, (which we have to do in any case), then it must be the case that both B is a formula and s is sufficient for it. So the rule: if s satisfies A and s satisfies $(x_i<\mathbf{0})$B then s satisfies (A∨B), will be appropriate to generate all the pairs of disjunctions and sequences that satisfy them, provided we have taken care of generating all the pairs [a,s] where a is a formula of the form $(x_i<\mathbf{0})$B and s is sufficient for B. In fact we will take care of generating all the pairs [a,s] where a is a formula of the form $(x_i<t)$B, t is a term of RE not containing x_i, s is sufficient for $(x_i<t)$B and the denotation of t with respect to s is 0 (for the same reason as above, all these pairs are in the relation of satisfaction).This is the reason for having a fourth kind of pairs in the basis set B: they are the pairs [a,s] where a is a formula of the form $(x_i<t)$B, B is atomic, s is sufficient for $(x_i<t)$B and the denotation of t with respect to s is 0.

In order to give a formula that defines this relation, let's introduce the following formulae:

$$s(i)=m =_{df.} [i,m] \in s;$$

this is simply a convenient abbreviation. The formula OcAtfmla(v,a) is true of v,a if v codes a variable, a is an atomic formula and the variable coded by v appears in the formula coded by a (notice that, in this case, the variable coded by v must appear free, since a has no quantifiers):

$$\mathrm{OcAtfmla}(v,a) =_{df.} \mathrm{Atfmla}(a) \wedge \mathrm{Vbl}(v) \wedge (\exists m \leq a)(\exists i \leq a)(\mathrm{Term}(m) \wedge [i,m] \in a \wedge (\exists j \leq m)([j,v] \in m)).$$

We now give a formula that defines the useful relation of concatenation between sequences: Cxyz will be true of m,n,p if they code sequences and p codes the sequence that results from concatenating the sequence coded by m and the sequence coded by n, in this order:

$$Cxyz =_{df.} (\exists m \leq x)(\exists n \leq y)(\mathrm{Seql}(x,m) \wedge \mathrm{Seql}(y,n) \wedge \mathrm{Seql}(z, m+n) \wedge x \subseteq z \wedge (j \leq n)(w \leq y)([j,w] \in y \supset [j+n,w] \in z)).$$

We can naturally define iterations of this relation, e.g.,

$$C^3wxyz =_{df.} (\exists m \leq z)(Cwxm \wedge Cmyz).$$

Now we give a formula $D4(a,s)$ which is true of a,s if a is a formula of the form $(x_i<t)B$ (where, therefore, x_i does not occur in t), B is atomic, s is sufficient for $(x_i<t)B$ and the denotation of t with respect to s is 0:

$$\begin{aligned} D4(a,s) =_{df.}\ & (\exists m_1 \leq a)(\exists m \leq a)(\exists v \leq a)[\mathrm{Term}(m) \wedge \mathrm{Term}(m_1) \wedge \mathrm{Vbl}(v) \wedge [\mathbf{0}^{(1)},v] \in m_1 \wedge \\ & (j \leq m)(w \leq m)([j,w] \in m \supset w \neq v) \wedge \\ & (\exists b_1 \leq a)(\exists b_2 \leq a)(Cb_1b_2a \wedge \mathrm{Atfmla}(b_2) \wedge \mathrm{Seql}(b_1,\mathbf{0}^{(5)}) \wedge b_1(\mathbf{0}^{(1)}) = [\mathbf{0},\mathbf{0}] \wedge b_1(\mathbf{0}^{(2)}) = m_1 \wedge \\ & b_1(\mathbf{0}^{(3)}) = [\mathbf{0},\mathbf{0}^{(3)}] \wedge b_1(\mathbf{0}^{(4)}) = m \wedge b_1(\mathbf{0}^{(5)}) = [\mathbf{0},\mathbf{0}^{(1)}]) \wedge \\ & (v_1 \leq a)((\mathrm{Vbl}(v_1) \wedge v \neq v_1 \wedge (\exists j \leq a)([j,v_1] \in m)) \vee \mathrm{OcAtfmla}(v_1,a)) \supset (\exists k \leq a)([v_1,k] \in s)) \wedge \\ & \mathrm{Den}(m,0,s)]. \end{aligned}$$

Finally, our basis set B is defined by the following formula of Re:

$$\mathrm{Basis}(x) =_{df.} (\exists a \leq x)(\exists s \leq x)(x = [a,s] \wedge (D1(a,s) \vee D2(a,s) \vee D3(a,s) \vee D4(a,s))).$$

Hence, B is r.e.

We turn now to defining the relations that will generate the set G from the basis set B. These will correspond fairly closely to the clauses in the inductive definition of satisfaction for assignments and formulae of RE. But we also have to take care of generating the more and more complex formulae of the form $(x_i<t)B$ where s is sufficient for $(x_i<t)B$ and the denotation of t with respect to s is 0.

We will define first the relations corresponding to the clauses in the inductive definition of satisfaction. First we consider the rule corresponding to the clause for conjunction: if s satisfies A and s satisfies B, then s satisfies $(A \wedge B)$, or schematically:

$$\frac{\text{Sat } (A, s), \text{Sat } (B,s)}{\text{Sat } ((A \wedge B), s).}$$

All we need to define an appropriate generating relation is a formula that defines the relation {<<A, s>, <B, s>, <(A ∧ B),s>>: A, B are sequences}; or, to be exact, a formula that defines the relation R_C={<[m,s], [n,s], [p,s]>: p codes the conjunction of the sequences that m and n code} (notice that if two pairs [a,s], [b,s] have been already generated, there is no further need to require that a and b be formulae and s be a finite function sufficient for a and b). Now, the formula Conj(x,y,z) is true of m,n,p if p is the conjunction of m and n:

$$\text{Conj}(x,y,z) =_{\text{df.}} (\exists l \leq z)(\exists c \leq z)(\exists r \leq z)(\text{Seql}(w,\mathbf{0}^{(1)}) \wedge \text{Seql}(c,\mathbf{0}^{(1)}) \wedge \text{Seql}(r,\mathbf{0}^{(1)}) \wedge l(\mathbf{0}^{(1)})=[\mathbf{0},\mathbf{0}] \wedge c(\mathbf{0}^{(1)})=[\mathbf{0},\mathbf{0}^{(6)}] \wedge r(\mathbf{0}^{(1)})=[\mathbf{0},\mathbf{0}^{(1)}] \wedge C^5 lxcyrz).$$

Given this, the RE formula with free variables x,y,z

$$(\exists s \leq x)(\exists m \leq x)(\exists n \leq y)(\exists p \leq z)(x=[m,s] \wedge y=[n,s] \wedge z=[p,s] \wedge \text{Conj } (m,n,p))$$

defines the generating relation we need.

For disjunction, we have two rules; schematically:

$$\frac{\text{Sat } (A,s), \text{ B is a formula of RE and s is sufficient for B}}{\text{Sat } ((A \vee B),s)}$$

and

$$\frac{\text{Sat } (B,s), \text{ A is a formula of RE and s is sufficient for A}}{\text{Sat } ((A \vee B),s).}$$

We could define the corresponding relation in a way analogous to the case of disjunction if we had defined the notion of formula of RE and the notion of being a function sufficient for a formula, but this is what we set ourselves to avoid. It is here that we will appeal to the trick explained above. For example, the first rule becomes:

$$\frac{\text{Sat } (A,s),\ \text{Sat}((x_i<\mathbf{0})B,s)}{\text{Sat } ((A \vee B),s).}$$

Now we need a formula that defines the relation {<<A, s>, <(x_i<**0**)B, s>, <(A ∨ B),s>>: A, B are sequences}; or, to be exact, a formula that defines the relation R_{D1}={<[r,s], [n,s], [p,s]>: n is a concatenation of (x_i<**0**) (for some i) and some sequence q, and p codes the disjunction of the sequences that r and q code}. Now, a formula Disj(x,y,z) true of r,n,p if p is the disjunction of m and n is easily definable as in the case of conjunction. Assuming this, the RE formula with free variables x,y,z

$$\begin{gathered}(\exists s\le x)(\exists r\le x)(\exists n\le y)(\exists q\le y)(\exists p\le z)(x=[r,s] \wedge y=[n,s] \wedge\ z=[p,s] \wedge \\ (\exists m_1\le n)(\exists v\le n)[\text{Term}(m_1) \wedge \text{Vbl}(v) \wedge [\mathbf{0}^{(1)},v]\in m_1 \wedge \\ (\exists b\le n)(\text{Cbqn} \wedge \text{Seql}(b,\mathbf{0}^{(5)}) \wedge b(\mathbf{0}^{(1)})=[\mathbf{0},\mathbf{0}] \wedge b(\mathbf{0}^{(2)})=m_1 \wedge b(\mathbf{0}^{(3)})=[\mathbf{0},\mathbf{0}^{(3)}] \wedge \\ b(\mathbf{0}^{(4)})=[\mathbf{0}^{(2)},\mathbf{0}^{(1)}] \wedge b(\mathbf{0}^{(5)})=[\mathbf{0},\mathbf{0}^{(1)}]] \wedge \\ \text{Disj}(r,q,p))\end{gathered}$$

defines the generating relation we need. The second rule for disjunction corresponds to another generating relation, R_{D2}, that can be defined analogously.

The rule for existential quantification is a bit more complicated:

$$\frac{\text{Sat } (A,s),\ s_1 \text{ differs from s at most in what it assigns to } x_i}{\text{Sat } ((\exists x_i)A,s_1)}$$

In order to define the appropriate generating relation we need a formula that defines the relation {<<A, s>, <($\exists x_i$)A, s_1>>: A is a sequence and s_1 differs from s (if at all) in what it assigns to x_i}; or, to be exact, a formula that defines the relation R_{EQ}={<[m,s], [n,s_1]>: n codes the concatenation of ($\exists x_i$) (for some i) and m, and s_1 differs from s (if at all) in what it assigns to x_i}. Now, a formula ExQu(m,y,p) true of x,i,z if p is a Gödel number of the concatenation of ($\exists x_i$) and m, is easily definable as in the above cases (remembering our special way of coding terms when they appear in formulae). We also need to have an RE formula Diff(s,s_1,v) that says that s and s_1 assign the same to variables other than v. This can be done as follows:

$$\begin{gathered}\text{Diff}(s,s_1,v) =_{\text{df.}} \text{Funct}(s) \wedge \text{Funct}(s_1) \wedge \text{Vbl}(v) \wedge (w\le s)(p\le s)(q\le s_1)((\text{Vbl}(w) \wedge w\ne v \wedge \\ [w,p]\in s \wedge [w,q]\in s_1)\supset p=q).\end{gathered}$$

Then R_{EQ} is definable by the following formula of RE with free variables x,y:

$$(\exists s\leq x)(\exists s_1\leq y)(\exists i\leq y)(\exists m\leq y)(\exists p\leq y)(x=[m,s]\wedge ExQu(m,i,p)\wedge y=[p,s_1]\wedge Diff(s,s_1,i)).$$

Finally, we come to bounded universal quantification, which is a bit subtler than the preceding cases. We have the rule

$$\frac{\text{Sat }((x_i<\mathbf{0}^{(n)})A,s),\ \text{Sat }(A,s_1),\ s_1 \text{ differs from s only in that } s_1 \text{ assigns n to } x_i,\ t \text{ denotes } n+1 \text{ with respect to s, } x_i \text{ does not occur in t}}{\text{Sat }((x_i < t)A,s)}$$

This says, roughly, that if s satisfies both $(x_i<\mathbf{0}^{(n)})A$ and $A(\mathbf{0}^{(n)})$, then s satisfies any formula of the form $(x_i<t)A$ where t denotes n+1 with respect to s. The corresponding relation is R_{UQ}={<[m,s], [n,s_1], [p,s]>: for some term t and some variable x_i not occurring in t, p codes the concatenation of $(x_i < t)$ and n, s_1 differs from s at most in that it assigns a number q to x_i, and m is the concatenation of $(x_i<\mathbf{0}^{(q)})$ and n}. Now, a formula UnQu(m,i,t,p) which "says" that p is the Gödel number of $(x_i<t)A$, where A is the formula whose Gödel number is m, is easily definable as in the above cases (taking care of remembering that we are not using the simple-minded coding scheme). Then we can define R_{UQ} by means of the following formula with free variables x,y,z:

$$(\exists s\leq x)(\exists s_1\leq y)(\exists m\leq x)(\exists n\leq y)(\exists p\leq z)(\exists i\leq x)(\exists t\leq z)(\exists q\leq x)(\exists r\leq x)(Term(t)\wedge Vbl(i)\wedge (j\leq t)(w\leq t)([j,w]\in t\supset w\neq i)\wedge UnQu(n,i,t,p)\wedge Diff\ (s,s_1,i)\wedge Num(q,r)\wedge Den(i,r,s_1)\wedge UnQu(n,i,q,m)\wedge x=[m,s]\wedge y=[n,s_1]\wedge z=[p,s]).$$

We are now done in our job of defining the generating relations corresponding to the clauses in the inductive definition of satisfaction for RE. We now have to make sure that we can define the relations that generate more and more complex formulae of the form $(x_i<t)A$.

For conjunction, we have the rule

$$\frac{\text{Sat }((x_i<t)A, s),\ \text{Sat }((x_i<t)B,s),\ t \text{ denotes 0 with respect to s}}{\text{Sat }((x_i<t)(A\wedge B), s).}$$

The corresponding relation is R_{C*}={<[m,s], [n,s], [p,s]>: for some q,r,x_i,t such that x_i is a variable, t is a term, m codes the concatenation of $(x_i<t)$ and q, n codes the concatenation of $(x_i<t)$ and r, the denotation of t with respect to s is 0 and p is the concatenation of $(x_i<t)$ and the conjunction of q and r}. R_{C*} is definable by the following formula with free variables x,y,z:

$$(\exists s\leq x)(\exists m\leq x)(\exists n\leq y)(\exists p\leq z)(\exists q\leq x)(\exists r\leq y)(\exists i\leq m)(\exists t\leq m)(x=[m,s] \wedge y=[n,s] \wedge z=[p,s] \wedge \mathrm{Vbl}(i) \wedge \mathrm{Term}(t) \wedge \mathrm{UnQu}(q,i,t,m) \wedge \mathrm{UnQu}(r,i,t,n) \wedge \mathrm{Den}(t,\mathbf{0},s) \wedge (\exists w\leq p)(\mathrm{Conj}(q,r,w) \wedge \mathrm{UnQu}(w,i,t,p))).$$

For disjunction, we have the rule

$$\frac{\mathrm{Sat}((x_i<t)A, s),\ \mathrm{Sat}((x_i<t)B,s),\ t \text{ denotes } 0 \text{ with respect to } s}{\mathrm{Sat}((x_i<t)(A \vee B), s),}$$

whose corresponding generating relation R_{D*} is definable similarly.

For existential quantification we have the rule

$$\frac{\mathrm{Sat}((x_i<t)A, s_1),\ t \text{ denotes } 0 \text{ with respect to } s, \text{ and } s_1 \text{ differs from } s \text{ at most in what it assigns to } x_j}{\mathrm{Sat}((x_i<t)(\exists x_j)A, s),}$$

whose corresponding relation R_{EQ*} is easily definable by means of the formulae that we already have.

The same is true for the relation R_{UQ*} corresponding to the relevant rule for bounded universal quantification:

$$\frac{\mathrm{Sat}((x_i<t)A, s_1),\ t \text{ denotes } 0 \text{ with respect to } s, \text{ and } s_1 \text{ differs from } s \text{ at most in what it assigns to } x_j,\ t_1 \text{ is a term and } x_j \text{ is a variable not appearing in } t_1}{\mathrm{Sat}((x_i<t)(x_j<t_1)A, s).}$$

Thus we finish our specification of the generating relations. It can be proved (by induction on the complexity of the formulae) that for every formula A and function s, if a codes A and s is a function that satisfies A, then [a,s] is generated from B by the relations R_C, R_{D1}, R_{D2}, R_{EQ}, R_{UQ}, R_{C*}, R_{D*}, R_{EQ*}, R_{UQ*}. So G is indeed the set generated from B, which is r.e., by these relations, which are r.e. Using the Generated Sets Theorem, we then reach the result that G is r.e., which is what we had set out to prove.

Given that RE contains its own satisfaction predicate, it also contains its own truth predicate, that is, there is a formula of RE Tr(m) with one free variable which is true of a (Gödel number of a) formula if and only if the formula is true in the intended interpretation of RE. Noting that a formula is true iff it is satisfied by a Gödel number of the empty sequence, we can define

$$Tr(m) =_{df.} Sat(m,\mathbf{0}^{(3)}).$$

(3 codes the empty sequence, for it is not in the range of our pairing function.)

If we had shown that RE contains its own truth predicate Tr(x), we could then have defined satisfaction with its help, using a remark due to Tarski: a number m satisfies a formula $A(x_1)$ with one free variable just in case the sentence $(\exists x_1)(x_1 = \mathbf{0}^{(m)} \wedge A(x_1))$ is true, so easily, since the concatenation function is definable in RE, we can define satisfaction in terms of truth within RE.

Exercises

1. Define the *characteristic function* of a set S to be the function which, for every natural number x, takes value 1 if $x \in S$, and value 0 if $x \notin S$. Similarly, the characteristic function of an n-place relation R is the function that for every n-tuple $<x_1,...,x_n>$, takes value 1 if $<x_1,...,x_n> \in R$, and value 0 if $<x_1,...,x_n> \notin R$. The *weak characteristic function* of a set S (or n-place relation R) takes value 1 on a number x (or n-tuple $<x_1,...,x_n>$) if x ($<x_1,...,x_n>$) belongs to S (R). (a) Show that a set or relation is recursive iff its characteristic function is. (b) Show that a set or relation is r.e. iff its weak characteristic function is partial recursive. (c) Show that the range and domain of any recursive function in one variable is r.e.

2. Show that the relation $z=x^y$ is r.e., and therefore that the exponentiation function is recursive by both the method of primitive recursion and the method of generated sets. How do the two defining formulae in RE differ from each other?

3. Use the Generated Sets Theorem to show that the Ackermann function is recursive. Where would an argument that the Ackermann function is primitive recursive break down?

4. Show that the set of Gödel numbers of formulae of the first order language of arithmetic is r.e., using the Generated Sets Theorem. Do the problem explicitly as follows. Write the basis clauses and generating clauses for the formulae themselves, and also show how they are then translated into basis clauses and generating clauses for codes. Then indicate what formula of RE results if we apply the argument in the proof of the Generated Sets Theorem to this case. Do the same for the set of formulae obeying the nested quantifier restriction.

Lecture VII

The Enumeration Theorem. A Recursively Enumerable Set which is Not Recursive

Now that we have shown how to define satisfaction for RE within RE itself, we are ready to prove what may be considered the fundamental theorem of recursion theory. First, let us define a variant on the notion of satisfaction we defined above. If B is a formula whose only free variable is x_1, then we say the number n satisfies$_1$ B just in case the unit sequence <n> satisfies B. We may thus define

$$\mathrm{Sat}_1\,(m, n) =_{df.} (\exists s)\,(\mathrm{Sat}\,(m, s) \wedge (y \in s)(y=[[\mathbf{0}^{(1)},\mathbf{0}^{(1)}],n]) \wedge [[\mathbf{0}^{(1)},\mathbf{0}^{(1)}],n] \in s).$$

One could in general define Sat_k, satisfaction for formulae with k free variables, either directly in a similar manner or by using the k-tupling function to reduce it to the case k=1.

We also use the notation W(e,n) for the relation that holds just in case Sat_1(e,n) holds. We now have the

Enumeration Theorem: There is a 2-place r.e. relation W such that for every r.e. set S, there is an e such that S = {n: W(e,n)}.

Proof: Let W_e={x: W(e,x)}. Each set W_e is defined by Sat_1 ($\mathbf{0}^{(e)}$,x) and is therefore r.e. If, on the other hand, S is an r.e. set, then it is defined by some RE formula B with one free variable. We may assume that B's free variable is x_1; letting e be a Gödel number of B, we see that S = W_e.

The Enumeration Theorem is so called because W enumerates the r.e. sets. This theorem is a standard theorem of recursion theory, though our presentation of it is not standard. When recursion theory is presented in terms of Turing machines, for example, W(e, x) is usually the relation *e codes a Turing machine which gives output "yes" on input x* for some fixed method of coding up Turing machines, and is shown to be r.e. by constructing a Turing machine which decodes the instructions given in e and applies them to the input x. In each formalism for developing recursion theory, the relation W(e, x) will be a different relation. (The notation 'W' originates in Kleene.)

In general, we can define a k+1-place relation W^{k+1} which holds of a k+1-tuple <e,n_1,...,n_k> if W(e, [n_1,...,n_k]) holds. This can be used to prove that W^{k+1} enumerates the k-place r.e. relations.

A very famous corollary of the Enumeration Theorem is the following:

Theorem. There is an r.e. set whose complement is not r.e. (thus, an r.e. set which is not recursive).

Proof: Let K be the set $\{x: W(x, x)\}$. K is clearly r.e., since it is defined by the formula $Sat_1(x_1, x_1)$. However, -K is not r.e., and so K is not recursive. To see this, suppose -K were r.e. Then we would have $-K = W_e$ for some e. By the definition of K, we see that $x \in -K$ iff $x \notin W_x$, so in particular $e \in -K$ iff $e \notin W_e$. But $W_e = -K$, so $e \in -K$ iff $e \notin -K$, contradiction.

(This shows that negation is not definable in RE: if it were, the complement of any r.e. set would be definable in RE, so in particular -K would be definable in RE.) This proof uses an idea due to Cantor. As we will see, once we have this theorem, Gödel's first incompleteness theorem is just around the corner.

The fact (if it is one) about the intuitive notion of computability corresponding to the enumeration theorem is that there is a semi-computable relation that enumerates all the semi-computable sets. It follows from this that there is a semi-computable set that is not computable. However, to prove the enumeration theorem for semi-computability, and thus to prove that not all semi-computable sets are computable, it seems necessary to use Church's Thesis. If there were a single language in which all computation procedures could be written out, then the enumeration theorem would follow: simply find some way of coding up this language numerically, and some effective way of decoding coded instructions and applying them to arguments. However, if we do not assume Church's Thesis, then it is by no means obvious that there is such a language. At first glance it might appear that the lesson of Gödel's work is that there is no single language in which all computation can be represented, just as there is no single fully classical language in which everything can be expressed. Every language will have some sort of Gödel-type limitation; it is a peculiarity of the language RE that the limitation is not that it cannot express its own semantic notions (as is the case with full first-order languages), but that it cannot express negation. But if it turns out that there are some semi-computable sets and relations that are not expressible in RE, then it is quite conceivable that all semi-computable sets and relations are computable and that the enumeration theorem for semi-computability fails.

The fact that the enumeration theorem is so fundamental to recursion theory, and that its proof for semi-computability requires Church's Thesis, indicates a limitation to how much recursion theory can be developed for the informal notion of computability by starting with intuitively true axioms about computability. Shoenfield tries this approach in his book on recursion theory; he winds up assuming the enumeration theorem, and does not give a fully convincing intuitive justification for it, since he in effect assumes that there is a single language in which all computation procedures can be represented.

The Road from the Inconsistency of the Unrestricted Comprehension Principle to the Gödel-Tarski Theorems

The result that RE contains its own truth and satisfaction predicates may seem surprising, since it is commonly said that Gödel and Tarski showed that no language can contain its own truth or satisfaction predicates. This is true for a wide range of languages, but the language RE is not among them. We shall now look at their result and at is historical roots.

Early on in the 20th century, it was discovered, to the surprise of many, that a certain set-theoretic principle is self-contradictory. The principle is called the *unrestricted comprehension scheme*:

$$(z_1)...(z_n)(\exists y)(x)(x \in y \equiv A(x, z_1, ..., z_n))$$

where A is any formula in the language of set theory whose free variables are among x and $z_1, ..., z_n$ (in particular, A does not contain y free). $z_1, ..., z_n$ are called *parameters*. In the case n = 0, we have the *parameter-free* unrestricted comprehension scheme:

$$(\exists y)(x)(x \in y \equiv A(x))$$

It is important to note that A may itself contain the predicate '$\in$'.

Russell showed that the unrestricted comprehension scheme, even in its parameter-free version, is self-contradictory: simply take A(x) to be the formula $\sim x \in x$. We then have

$$(x)(x \in y \equiv \sim x \in x)$$

for some y, from which it follows that

$$y \in y \equiv \sim y \in y$$

which is directly self-contradictory. This observation is called *Russell's paradox*.

Russell got the idea of his paradox by analyzing Cantor's proof via diagonalization that there is no function mapping a set onto its powerset, and applying it to a more complicated paradox that embedded his. The Russell paradox is not the only set-theoretic paradox. Other paradoxes were discovered at the very time of the formation of set theory itself. For example, there is the Burali-Forti paradox, and the paradox of the greatest cardinal. In general, these paradoxes, like the Russell paradox, can be used to show that the unrestricted comprehension scheme is inconsistent, or at least, that it leads to an inconsistency in conjunction with the axiom of extensionality.

If the unrestricted comprehension scheme is logically self-contradictory, this cannot depend in any way on the interpretation of '$\in$'. From a purely formal point of view, it doesn't

matter whether '$\in$' means 'is a member of' or 'is satisfied by'. In fact, if you knew somehow that the unrestricted comprehension scheme is inconsistent but didn't know why, you would be able to see immediately that no first-order language can contain its own satisfaction predicate. Consider the following interpretation of a first-order language with '$\in$' as its only predicate: let the variables range over the formulae with one free variable of the language, and assume that the language contains its own expressions in its own domain. Suppose that we interpret '$\in$' by means of the 2-place relation S(y,x) which holds just in case y is a formula with one free variable and x satisfies y. Then suppose that the language could define its own satisfaction for formulae of one free variable. Then we would have a model that would make the comprehension scheme true, because a formula A(x) can be taken to be the object y, and then the principle says that x satisfies y if and only if A(x), which is of course true. So any proof of the inconsistency of the comprehension principle proves that a first order language cannot contain its own satisfaction predicate for formulae with one free variable. Often, the results of Gödel and Tarski are presented as if they involved all kinds of sophisticated ideas very different from these, but the inconsistency of the unrestricted comprehension scheme is essentially what proves those results.

(Now, suppose that in the derivation of a paradox we used the unrestricted comprehension axiom with n parameters. How are we going to interpret 'x $\in$ y' in this case? One way will be analogous to the reduction of satisfaction to truth that we saw for the case of RE. We take y to range over the formulae with one free variable, or their codes, and 'x $\in$ y' to be defined by "x satisfies y", where y will be, or code, the formula with one free variable

$$(\exists z_1)...(\exists z_n)(z_1=\mathbf{0}^{(m_1)} \wedge ... \wedge z_n=\mathbf{0}^{(m_n)} \wedge A(x, z_1, ..., z_n)),$$

where $m_1,...,m_n$ are the parameters. We could also use a relation of substitution of terms for free variables in formulae to specify how y will be, in a way indicated below. Or we could define 'x $\in$ y' with the help of the pairing function, as holding between a number and a pair composed of (the code of) the formula $A(x, z_1, ..., z_n)$ and a finite function assigning values to all of the variables $z_1, ..., z_n$.)

We can use what we have learned about satisfaction to state some very general results about the indefinability of truth. As long as the language L contains a name 'a' for each object a in its domain, we can (in the metalanguage) define satisfaction for L in terms of truth for L: an object a satisfies a formula $A(x_1)$ just in case the sentence A('a') is true (where A('a') is got from $A(x_1)$ by replacing all free occurrences of x_1 with 'a'). We can turn this into a definition in L of satisfaction in terms of truth as long as L possesses certain syntactic notions. Suppose, for example, L has function symbols Q and S denoting functions q and s, where q(a) = 'a' for all a in L's domain, and $s(A(x_1), t) = A(t)$ for all formulae $A(x_1)$ and terms t. Then $s(A(x_1), q(a)) = A('a')$ for all $A(x_1)$ and a, and so the formula Tr(S(y,Q(x))) of L will define satisfaction in L if Tr defines truth in L. Since

satisfaction for L is not definable in L, it follows that truth is not definable either. So in general we see that, for fully classical L, the following conditions cannot jointly obtain:

1. Truth for L is definable in L.
2. The relation of substitution of terms for free variables is definable in L.
3. Every object in L's domain has a name, and moreover the function from an object to its name is definable in L.

In fact, somewhat weaker conditions than 2 and 3 are also incompatible with 1. For example, 3 may be replaced with: every object in L's domain is denoted by some term of L and the relation *t denotes x* is definable in L. For suppose Den(t, x) defined that relation; then the formula $(\exists z)(Den(z, x) \wedge Tr(S(y, z)))$ would define satisfaction in L in terms of truth. (We could further weaken 3 by assuming only that every object a of L's domain has a definite description D in L, and that we can specify D in terms of a within L.) Also, we can use the trick remarked on by Tarski to avoid using the substitution function. An object a will satisfy $A(x_1)$ just in case the sentence $(\exists x_1)(x_1 = 'a' \wedge A(x_1))$ is true (or $(\exists x_1)(x_1 = t \wedge A(x_1))$ is true for some term t denoting a, or $(\exists x_1)(D(x_1) \wedge A(x_1)$ is true for some definite description D of a), so as long as the function $F(t, A(x_1)) = (\exists x_1)(x_1 = t \wedge A(x_1))$ is definable in L, and 3 obtains, we can define satisfaction in terms of truth within L. Moreover, F is itself easily definable in terms of concatenation (as long as terms for the primitive symbols of L exist in L), and is anyway simpler than the substitution function, which has to distinguish between free and bound occurrences of variables. To put it in a succinct form, we see that a language cannot both define all the devices that can be used to reduce truth to satisfaction and contain its own truth predicate.

So far, we have been concentrating on interpreted languages whose domains include all the expressions of the language itself. However, we can generalize the discussion by considering languages that can talk about their own expressions indirectly, via coding. Let L be a language with a countable vocabulary and with an infinite domain D. Suppose we had a function f mapping the elements of some subset D_1 of D onto the formulae, or at least onto the formulae with one free variable. Call this a coding function. Given any coding function f for L, the relation {<y,x>: x is in D_1 and f(y) is satisfied by x} of "coded satisfaction" between elements of L's domain is not itself definable in L: if S(y,x) defined it, then the unrestricted comprehension scheme, with '$x \in y$' replaced by 'S(y,x)', would be true in L, which we know to be impossible. Note that none of this depends on any particular facts about f; any coding function will do, and we know that such a mapping will exist whenever L has an infinite domain.

As before, this is related to the question of the definability of truth in L. Let us say that a formula Tr(x) of L defines truth in L (relative to f) if Tr(x) defines {y: f(y) is true in L}. We can always find a function f such that some formula of L defines truth in L relative to f (for practically any L). For example, let L be the language of arithmetic, and let f "assign"

Gödel numbers to sentences of L so that the Gödel numbers of the true sentences of L are 0, 2, 4, ... and the Gödel numbers of false sentences of L are 1, 3, 5, ... (Or if arbitrary formulae are to be coded rather than just sentences, we can let 0, 3, 6, ... code the true sentences, let 1, 4, 7, ... code the false sentences, and let 2, 5, 8, ... code the rest of the formulae.) Relative to such a Gödel numbering, truth in L is obviously definable in L. However, for any coding f, conditions 1-3 above will still not be jointly satisfiable, and so for this particular Gödel numbering, the basic syntactic notions will not be definable in L. For any ordinary Gödel numbering (e.g. our own numbering), the syntactic notions will be definable, and so truth will be undefinable. But again, satisfaction will be undefinable for any f, and for any L.

Now, as we have seen, if we drop the requirement that L be fully classical, these indefinability results will no longer hold, since the language RE has its own satisfaction predicate. However, RE does not have its own *unsatisfaction* predicate, i.e. there is no formula U(y,x) of RE that obtains just in case x fails to satisfy y. (If there were, then we could define -K in RE by $U(x_1, x_1)$.) More generally, if L is a language with classical semantics, but not necessarily with all classical connectives, then unsatisfaction is not definable in L. To see this, suppose we had an unsatisfaction predicate U(y,x). Then letting u be the formula U(y,y), we would have that U(u,y) obtains iff y does not satisfy U(y,y), and so U(u, u) obtains iff u does not satisfy U(y,y); but to say that U(u, u) obtains is just to say that u satisfies U(y, y), so this is impossible. Similarly, given suitable restrictions on L, a language cannot have its own untruth predicate. Similar remarks apply when we allow languages to talk about their formulae indirectly via codes.

The enumeration theorem is really a form of the naive comprehension scheme for RE, since the content of the theorem is that for every RE formula $A(x_1)$ there is an e such that

$$(x_1)(A(x_1) \equiv W(e, x_1))$$

We cannot derive a contradiction by letting A be the formula $\sim W(x_1, x_1)$, since negation is not definable in RE. This shows that it is essential to use either negation or unbounded universal quantification in showing that scheme to be inconsistent for classical languages, since RE lacks both negation and unbounded universal quantification. In fact, however, there are languages which have unbounded universal quantification, as well as the other logical symbols of RE, but which lack negation, and which have their own satisfaction predicates; so only the use of negation is really essential.

From all of this it follows that, given our Gödel numbering, the language of arithmetic does not have its own truth predicate. This was originally shown by Tarski as an application of the work of Gödel. However, there it was presented in a more complicated way as an application of the liar paradox. First, Gödel's self-reference theorem was used to obtain, for any formula $T(x_1)$ of the language of arithmetic, a sentence A such that

$$A \equiv \sim T(\mathbf{0}^{(n)})$$

is true, where n is the Gödel number of A. If $T(x_1)$ defined the set of truths of arithmetic, then we would also have

$$A \equiv T(\mathbf{0}^{(n)})$$

from which $T(\mathbf{0}^{(n)}) \equiv \sim T(\mathbf{0}^{(n)})$ follows, which is self-contradictory. Intuitively, if $T(x_1)$ means "x_1 is true", then the sentence A says of itself that it is not true; so the Tarski-Gödel proof can be seen as an application of the liar paradox. However, the construction of the sentence A is rather tricky, and leaves one with the impression that something much more subtle is going on here than actually is. In fact, when one takes the Tarski-Gödel proof apart, one sees that it really boils down to the observation that arithmetic lacks its own satisfaction predicate, which in turn is a direct consequence of the Russell paradox, as we saw above.

Under the interpretation of '$\in$' as the relation of satisfaction, Russell's paradox is known as 'the paradox of 'heterological''. Grelling was the discoverer of this paradox, and he obtained it by reflecting on Russell's paradox. Call a predicate of English *autological* if it is true of itself, and *heterological* otherwise. (For example, 'polysyllabic' is autological and 'monosyllabic' is heterological.) A problem arises when we ask whether 'heterological' is itself heterological. 'heterological' is heterological iff 'heterological' is not true of 'heterological', iff 'heterological' is not heterological—a contradiction. If we interpret '$x \in y$' to mean 'y is true of x', then the formula '$x \notin x$' means that x is heterological, and the derivation of the Grelling paradox is formally identical to the above derivation of the Russell paradox.

Gödel mentions the liar paradox (and the paradox of Richard) as sources for the reasoning leading to his theorems. He does not mention Russell's paradox, or the paradox of 'heterological', although the Tarski-Gödel results are more naturally motivated by appeal to them, as we have seen. That Russell's paradox and Grelling's paradox can be most naturally put to this use is perhaps a fact known to some logicians, but to the author's knowledge, it is not mentioned in the printed literature.

In fact, some logicians have probably misunderstood the relation between the liar paradox on the one hand and Russell's and Grelling's paradoxes on the other. That the Gödel-Tarski results can be motivated in the two ways is no surprise, for, in fact, on one way of stating the liar paradox it just is the Grelling paradox. The liar paradox is traditionally stated in terms of sentences like 'This sentence is false'. One way to get a liar sentence without using locutions like 'this sentence' is via the sentence "Yields a falsehood when appended to its own quotation' yields a falsehood when appended to its own quotation'. Since the result of appending the phrase mentioned in the sentence to its own quotation is the sentence itself, the sentence says of itself that it is false. (A briefer way of

writing the sentence is as follows: "Is not true of itself' is not true of itself'.) Quine gives this version of the liar paradox in "The Ways of Paradox". But he goes on to say that this antinomy is "on a par with the one about 'heterological'" ("The Ways of Paradox", in *The Ways of Paradox,* New York, Random House, 1966; p. 9). This is at best misleading, especially in this context, for the paradox simply *is* the Grelling paradox, since 'is heterological' means the same as 'yields a falsehood when appended to its own quotation'.

Lecture VIII

Many-one and One-one Reducibility.

Given that not all sets are recursive, and indeed that some r.e. sets are not recursive, we may want to ask of some nonrecursive set A whether the problem of deciding membership in it can be reduced to that of deciding membership in some other set B. This idea gives rise to a number of reducibility notions in recursion theory; in this section, we shall discuss two of the simplest such notions.

We say that a set A is *many-one reducible* to B (in symbols, $A \leq_m B$) if there is a total recursive function ϕ such that for all m, $m \in A$ iff $\phi(m) \in B$. (We also say *m-reducible* for *many-one reducible*.) We also write $\phi: A \leq_m B$ when ϕ is a total recursive function satisfying this condition. If $\phi: A \leq_m B$ and ϕ is 1-1, then we say that A is *1-1 reducible* (or *1-reducible*) to B (in symbols, $A \leq_1 B$).

One way to think of this informally is in terms of *oracles*. Suppose there were an oracle that could tell you, for an arbitrary number, whether it is an element of B. Then if $A \leq_m B$, you will have a way to use the oracle to find out whether an arbitrary number is an element of A: to see whether $m \in A$, simple compute $\phi(m)$ and consult the oracle. If the oracle tells you that $\phi(m) \in B$, then you know that $m \in A$, and if it tells you that $\phi(m) \notin B$, then you know that $m \notin A$.

Here's a simple example of 1-1 reducibility. Let A be an arbitrary set, and let $B = \{2m: m \in A\}$. Then we see that $A \leq_1 B$ by letting $\phi(m) = 2m$. And intuitively, we can effectively determine whether $m \in A$ by consulting the oracle about whether $2m \in B$.

It can readily be shown that the relations $\leq_1$ and $\leq_m$ are reflexive and transitive. Let us write $A \equiv_m B$ for $A \leq_m B$ & $B \leq_m A$, and similarly for $\equiv_1$; it then follows that $\equiv_m$ and $\equiv_1$ are equivalence relations. The $\equiv_m$-equivalence classes are called many-one degrees or m-degrees; similarly, the $\equiv_1$-equivalence classes are called 1-1 degrees or 1-degrees.

The relations $\leq_m$ and $\leq_1$ do not coincide. To see this, let A = {even numbers}, and let $\phi(m) = 0$ if m is even, 1 if m is odd. Then we see that $\phi: A \leq_m \{0\}$. However, there is clearly no 1-1 function ϕ such that $x \in A$ iff $\phi(x) \in \{0\}$, so A is not 1-reducible to {0}.

A set is *many-one complete r.e.* (*1-1 complete r.e.*) iff it is r.e. and every r.e. set is many-one reducible (1-1 reducible) to it. If S is many-one complete r.e., then every r.e. set is of the form $\{x: \phi(x) \in S\}$ for some total recursive ϕ. One example of a many-one complete set (which is also 1-1 complete) is the set $S = \{[e, x]: x \in W_e\}$. To see that S is 1-1 complete, let S_1 be any r.e. set. S_1 is W_e for some e, so let $\phi(x) = [e, x]$; $x \in S_1$ iff $x \in W_e$ iff $\phi(x) \in S$. ϕ is clearly as required; in particular, ϕ is recursive, since its graph is defined by the formula $y = [\mathbf{0}^{(e)}, x]$. Another 1-1 complete set is the set T of Gödel numbers of true RE sentences. To see this, note that if $A(x_1)$ defines a set S_1, then $n \in S_1$ iff $A(\mathbf{0}^{(n)})$ is true; so if $\phi(n)$ is a recursive function that picks a Gödel number of $A(\mathbf{0}^{(n)})$ (for

example, the smallest one), then $x \in S_1$ iff $\phi(x) \in T$. The set K can also be shown to be 1-1 complete, but the proof is a bit trickier.

If $S_1 \leq_m S_2$ and S_2 is r.e. (recursive), then S_1 is also r.e. (recursive). An r.e. m-complete set cannot be recursive: if S were an m-complete recursive set, then $K \leq_m S$, and this would imply that K is recursive, which it is not.

Some questions about reducibility naturally arise. First, is there an r.e. set which is neither recursive nor many-one complete? Emil Post answered this question in the affirmative, and we shall prove his result later on in the course. Second, are there any many-one complete sets that are not 1-1 complete? The answer is no; this result is surprising, and the proof is nontrivial; we shall give the proof later on. (The notion of being 1-complete and of being m-complete are also equivalent to the notion of being *creative*, which we shall define later on.)

Despite Post's result, all (or practically all) naturally arising r.e. sets are either 1-1 complete or recursive. That is, while r.e. sets that are neither 1-1 complete nor recursive exist in great abundance, they tend to arise as cooked-up counterexamples rather than sets which are interesting for separate reasons. A common way to prove that an r.e. set is nonrecursive is to show that some 1-1 complete set reduces to it, which implies that it is 1-1 complete.

Another way to put this is in terms of degrees. Among the r.e. 1-1 degrees (i.e. $\equiv_1$-equivalence classes containing r.e. sets), there is a degree on top (the degree of 1-1 complete sets), and, excluding the degrees containing finite and cofinite sets, a degree on the bottom (the degree of recursive sets with infinite complements), and many degrees in between. However, all the naturally occurring r.e. sets are to be found on top or on the bottom.

Besides $\leq_m$ and $\leq_1$, there are coarser-grained reducibility relations, all of which give an intuitive notion of the idea that given an oracle for a set B, we can decide A. Post, who originally formulated the notions of many-one and 1-1 reducibilities, gave a variety of reducibility notions, still studied today. One of his notions, the broadest of all, was supposed to capture the intuitive notion of being able to decide A given an oracle that will answer all questions about set membership in B. He called this notion 'Turing-reducibility'; it has also been called 'relative recursiveness' and 'recursiveness in'. As we said above, Post found an r.e. set that was not many-one complete (and therefore not 1-1 complete). However, he was able to define another reducibility relation with respect to which this set was still complete. In general, he found broader and broader reducibility relations, and more complicated r.e. but not recursive sets that failed to be complete with respect to them. However, he could not solve this problem for the basic notion of Turing-reducibility, and it was a long-standing question whether there are any r.e. sets which are neither recursive nor Turing-complete. This was answered in the affirmative in 1956 by Friedberg and Mucnik.

Two sets A and B are called *recursively isomorphic* (in symbols, $A \equiv B$) if there is a 1-1 total recursive function ϕ which maps **N** *onto* **N**, and such that $B = \{\phi(x): x \in A\}$. (It follows that ϕ^{-1} is also such a function and that $A = \{\phi^{-1}(x): x \in B\}$.) If $A \equiv B$, then it is

easy to see that $A \equiv_1 B$, since ϕ: $A \leq_1 B$ and ϕ^{-1}: $B \leq_1 A$. The converse is also true, and is highly nontrivial. It was once proposed that recursion theory be regarded as the study of those properties of sets of natural numbers which are invariant under recursive isomorphism. In fact, nearly all the properties studied by recursion theory are of this nature; however, there are some exceptions.

The Relation of Substitution

In several occasions we have mentioned the relation of substitution of a term for all the free occurrences of a variable in a formula, noting that we could use it to give alternative proofs of some results. For example, we could have given an alternative proof that RE defines its own satisfaction using definitions of the notion of formula and of the relation of substitution. Also, we could have defined truth in RE by means of satisfaction in RE using substitution, instead of Tarski's trick. We will now show how a certain notion of "naive substitution" is definable within RE, leaving as an exercise showing how to define the notion of proper substitution.

First, we will define a relation that we will call "term substitution": specifically, we shall define the relation $\{<t_1, t_2, v, t>$: the term t_2 comes from the term t_1 by replacing all occurrences of the variable v, if any, by the term t; if v does not occur in t_1, then $t_1=t_2\}$. This is defined by the following formula of RE:

$$\begin{aligned}\mathrm{TSubst}(t_1, t_2, v, t) =_{df.} \mathrm{Vbl}(v) \wedge \mathrm{Term}(t_1) \wedge \mathrm{Term}(t_2) \wedge \mathrm{Term}(t) \wedge [((j \leq t_1) \sim([j,v] \in t_1) \wedge t_1 = t_2) \\ \vee ((\exists j \leq t_1)([j+1,v] \in t_1 \wedge (\exists l \leq t)(\mathrm{Seql}(t,l+1) \wedge \mathrm{Seql}(t_2,j+l+1) \wedge (y \leq t)([l+1,y] \in t \supset [j+l,y] \in t_2) \\ \wedge (i \leq j+l)([i, [\mathbf{0}^{(4)}, [\mathbf{0}^{(1)},\mathbf{0}^{(1)}]]] \in t_2))))].\end{aligned}$$

(Recall that all terms of RE are either of the form $f_1^1 \ldots f_1^1 \mathbf{0}$ or of the form $f_1^1 \ldots f_1^1 x_i$, for some i.)

We now show how to define the notion of "naive substitution", that is the relation $\{<m_1, m_2, v, m>$: the sequence m_2 comes from the sequence m_1 by replacing all occurrences of the variable v by the term m$\}$. We call this "naive" substitution because the result of a substitution of this kind may not be a formula, even if the expression operated upon was one (note, for example, that even occurrences of the variable to be replaced within quantifiers will be replaced, so if the replacing term is not a variable, substitution may transform a quantifier into an expression that cannot form part of a formula.) Naive substitution is definable by the following formula of RE:

$$\begin{aligned}\mathrm{NSubst}(m_1,m_2,v,m) =_{df.} (\exists l \leq m_1)(\mathrm{Seql}(m_1,l) \wedge \mathrm{Seql}(m_2,l) \wedge (i \leq l)(y \leq l)[(([i,y] \in m_1 \wedge \\ \sim(\exists j \leq y)([j,v] \in y)) \supset [i,y] \in m_2) \wedge (([i,y] \in m_1 \wedge (\exists j \leq y)([j,v] \in y)) \supset (z \leq m_2)(\mathrm{TSubst}(y,z,v,m) \\ \supset [i,z] \in m_2))]).\end{aligned}$$

This simply "says" that if y is a part of m_1 in which v does not occur, it's left untouched in m_2, and if it is a term in which v occurs, v is replaced by m in y to obtain a term z which is then part of the result m_2.

The result of a naive substitution will not in general be a formula. It will be a formula, however, if the variable to be replaced never occurs bound in the initial formula (and also if the term replacing the variable in a formula is another variable). The utility of naive substitution can be seen from the fact that it is sufficient for showing that the set of logical axioms (of a given language) in the deductive system of the next section is r.e. Notice, in particular, that axiom schema 4 only invokes naive substitution. In standard systems, in place of axiom schemata 4 and 5 we find schemata 4' and 5', and the natural way of coding these is more complicated. It involves proper substitution, that is, the relation {$<m_1, m_2, v, m>$: the sequence m_2 comes from the sequence m_1 by replacing all *free* occurrences of the variable v by the term t; and no variable occurring in m becomes bound in m_2}. The definition of proper substitution in RE is left as an exercise.

Deductive Systems.

We want, for a given language L, a deductive system in which all and only the valid sentences of L are provable. When L does not contain function symbols, the following is such a system.

The axioms are all of the instances (in L) of the following schemata:

1. $A \supset (B \supset A)$;

2. $(A \supset (B \supset C)) \supset ((A \supset B) \supset (A \supset C))$;

3. $(\sim A \supset \sim B) \supset (B \supset A)$;

4. $(x_i)A \supset A'$, where A' is got from A by substituting all occurrences of x_i in A by a fixed term t, and neither (x_i) nor (x_j) occurs in A, where x_j is any variable occurring in t;

5. $A \supset (x_i)A$, where x_i does not occur in A;

6. $(x_i)(A \supset B) \supset ((x_i)A \supset (x_i)B)$.

There are also two inference rules:

modus ponens (MP): $A \supset B, A$ universal generalization (UG): A

$$\frac{\quad}{B} \qquad\qquad \frac{}{(x_i)A}$$

A *proof* in L is a finite sequence of formulae of L such that each formula is either an axiom or follows from earlier formulae by one of the inference rules. A sentence is a theorem just in case it occurs in a proof. More generally, a proof from Γ, where Γ is a set of *sentences* of L, is a finite sequence of formulae in which each formula is either an axiom *or an element of Γ*, or follows from earlier formulae by the inference rules. A sentence is a theorem of Γ just in case it occurs in a proof from Γ. We write fi A to mean that A is a theorem, and Γ fi A to mean that A is a theorem of Γ.

Remarks. The axiom schemata 1-3, along with MP, are sufficient to prove all tautologies. We could have simply taken all tautologies (of L) as axioms, but the present approach will prove more convenient later.

The present system differs from standard systems in that axiom schemata 4 and 5 are usually formulated as follows:

4'. $(x_i)A \supset A'$, where A' is got from A by replacing all *free* occurrences of x_i in A by a fixed term t, where either t is a constant, or t is a variable x_j and no free occurrence of x_i in A falls within the scope of a quantifier (x_j).

5'. $A \supset (x_i)A$, where x_i does not occur *free* in A.

All instances of 4' and 5' that are not also instances of 4 and 5 prove to be derivable and hence redundant. As we said, 4 and 5 are simpler to represent in our coding of syntax than 4' and 5'.

(In any deductive system which contains a version of universal instantiation, some restriction like that in 4 or 4' must be made, for the *prima facie* more natural scheme

$(x_i)A \supset A'$, where A' comes from A by replacing all free occurrences of x_i in A by a fixed term t

is invalid. To see this, consider the instance $(x)(\exists y)\ x \neq y \supset (\exists y)\ y \neq y$. In any interpretation whose domain has more than one element and in which = is interpreted as identity, this sentence is false. The problem is that the instantial term, y, becomes bound once it is substituted for x; as long as we prevent this sort of thing, the restricted scheme will be valid.)

We say that a system is *sound* just in case every theorem is valid, and every theorem of Γ is a consequence of Γ, for any Γ. A system is *complete* if every valid sentence is a theorem, and *strongly complete* if for all Γ, every consequence of Γ is a theorem of Γ. The

proof that our system is sound is fairly easy by induction on the length of a proof of any given theorem. We will sketch later the argument for completeness (and strong completeness), with more discussion of variant formulations, and in particular of the comprehensive virtues and drawbacks of 4 and 5 as opposed to 4' and 5'. In particular, how to derive 4' and 5' from 4 and 5 will be sketched later.

The Narrow and Broad Languages of Arithmetic.

By the *narrow* language of arithmetic, we mean the language as given in lecture I. Recall that the primitive connectives of that language are $\supset$ and $\sim$ and that (x_i) is the only primitive quantifier; so, perversely, the logical vocabulary of the narrow language is disjoint from that of RE. By the *broad* language of arithmetic we mean the language which has all the vocabulary of the narrow language, and in addition the primitive connectives $\wedge$ and $\vee$ and the quantifiers $(\exists x_i)$, $(x_i < t)$, and $(\exists x_i < t)$ (with the usual restrictions on t). So the broad language of arithmetic literally contains the languages RE and Lim (but not Lim^+). We shall use L ambiguously to refer to both the broad and the narrow language of arithmetic; when we wish to refer to them unambiguously, we shall use L^- to refer to the narrow language and L^+ to refer to the broad language. Note that L^- and L^+ are equal in expressive power, since the extra connectives of L^+ are already definable in L^-.

The language L^+ has redundancies, as its extra connectives and quantifiers are already definable in L^-. L^- also has redundancies. For example, the negation sign is superfluous in L^-, as $\sim A$ is equivalent to $A \supset \mathbf{0} = \mathbf{0}'$. If we had included the function symbols + and $\cdot$, then all the connectives would have been superfluous, since they could be eliminated in the manner indicated in an exercise. Even in L^+, we can eliminate all of the connectives except for $\wedge$. To see this, let A be any formula of L^+, and let A* be an equivalent formula in which A and M are replaced by + and $\cdot$. Let A** be an equivalent formula in which no connectives appear, constructed in the way indicated in lecture II. Finally, let A*** be a formula of L+ got from A** by Russell's trick. Since the only connective that Russell's trick introduces is $\wedge$, $\wedge$ is the only connective A*** contains; and A*** is equivalent to A.

Note that our deductive system only has axioms and rules for the connectives $\sim$ and $\supset$ and the quantifier (x_i); when we are considering the broad language of arithmetic, we want our system to prove all the valid formulae that contain the new logical vocabulary as well as the old. This can be achieved by adding the following equivalence axiom schemes when we are working in L^+:

$(A \vee B) \equiv (\sim A \supset B)$
$(A \wedge B) \equiv \sim(A \supset \sim B)$
$(\exists x_i)A \equiv \sim(x_i)\sim A$
$(x_i < t)A \equiv (x_i)(\text{Less}(x_i, t) \supset A)$

$(\exists x_i < t)A \equiv (\exists x_i)(\text{Less}(x_i, t) \wedge A)$

(Note that $\equiv$ is a defined rather than a primitive symbol, even in L^+.) It can be shown (though we will not show it here) that when these schemes are taken as axioms, every formula of L^+ is provably equivalent to a formula of L^-, i.e. for every formula A of L^+ there is a formula A' of L^- such that $A \equiv A'$ is provable. Thus, the completeness of the system of L^+ follows from that for L^-.

It is easy to show that the set of logical axioms for L (either L^+ or L^-) is r.e. We can then prove, using the Generated Sets Theorem, that the set of provable formulae is r.e., and that if Γ is an r.e. set of sentences of L, then the set of theorems of Γ is r.e. We can generalize this to languages other than L. If K is a first order language such that the set of formulae of K is r.e., then the set of provable formulae in K is r.e., and if Γ is an r.e. set of sentences of K, then the set of theorems of Γ is r.e. For the set of formulae of K to be r.e., it is necessary and sufficient that the set $\{i: a_i \in K\}$ and the relations $\{<n, i>: P_i^n \in K\}$ and $\{<n, i>: f_i^n \in K\}$ be r.e.

The Theories Q and PA.

There are two theories in the language L that are traditionally given special attention. One is the theory Q, also called *Robinson's Arithmetic* (after its inventor Raphael Robinson). Q is usually given in the language with + and ·, so our version of it is slightly nonstandard. In the usual version, the axioms of Q are

1. $(x_1)\ \mathbf{0} \neq x_1'$
2. $(x_1)(x_2)\ (x_1' = x_2' \supset x_1 = x_2)$
3. $(x_1)\ (x_1 = \mathbf{0} \vee (\exists x_2)\ x_1 = x_2')$
4. $(x_1)\ x_1 + \mathbf{0} = x_1$
5. $(x_1)(x_2)\ x_1 + x_2' = (x_1 + x_2)'$
6. $(x_1)\ x_1 \cdot \mathbf{0} = \mathbf{0}$
7. $(x_1)(x_2)\ x_1 \cdot (x_2') = x_1 \cdot x_2 + x_1$

(Axioms 4-7 are usually called the *recursion axioms*.) To adapt this to our language, we rewrite the axioms as follows:

1. $(x_1)\ \mathbf{0} \neq x_1'$
2. $(x_1)(x_2)\ (x_1' = x_2' \supset x_1 = x_2)$
3. $(x_1)\ (x_1 = \mathbf{0} \vee (\exists x_2)\ x_1 = x_2')$
4. $(x_1)\ A(x_1, \mathbf{0}, x_1)$
5. $(x_1)(x_2)(x_3)\ (A(x_1, x_2, x_3) \supset A(x_1, x_2', x_3'))$

6. $(x_1)\, M(x_1, \mathbf{0}, \mathbf{0})$
7. $(x_1)(x_2)(x_3)(x_4)\, ((M(x_1, x_2, x_3) \wedge A(x_3, x_1, x_4) \supset M(x_1, x_2', x_4))$

(Note that axioms 1-3 are unchanged.) In addition, we need existence and uniqueness axioms for addition and multiplication:

$(x_1)(x_2)(\exists x_3)\, (A(x_1, x_2, x_3) \wedge (x_4)(A(x_1, x_2, x_4) \supset x_4 = x_3))$
$(x_1)(x_2)(\exists x_3)\, (M(x_1, x_2, x_3) \wedge (x_4)(M(x_1, x_2, x_4) \supset x_4 = x_3))$

(These are unnecessary for Q as it is usually stated, because their analogs in the language with function symbols + and · rather than predicates A and M are logical truths.) Finally, we did not include axioms for identity in our deductive system for the predicate calculus, so we must include them here. The usual identity axioms are the reflexivity axiom $(x_1)\, x_1=x_1$ and the axiom scheme $(x_1)(x_2)\, (x_1 = x_2 \supset A \equiv A')$, where A is any formula with at most x_1 and x_2 free and A' comes from A by replacing one or more free occurrences of x_1 by x_2. In fact, we can get away with taking only finitely many instances of this scheme as axioms, and the rest will be deducible. Specifically, we can take as our identity axioms the reflexivity axiom and those instances of the above scheme in which A is an atomic formula not containing the function symbol '. Since there are only finitely many predicates in L, there are only finitely many such instances. Q, then, is the theory in L whose axioms are 1-7 above along with the existence and uniqueness clauses and the identity axioms just specified.

PA, or *Peano Arithmetic*, comes from Q by deleting axiom 3 and adding all those sentences which are instances in L of the *induction scheme*:

$$[A(\mathbf{0}) \wedge (x_1)(A(x_1) \supset A(x_1'))] \supset (x_1)A(x_1).$$

(Axiom 3 is a theorem of the resulting system, so we need not take it as an axiom.)

The intuitive idea behind the induction scheme is that if zero has a property, and if whenever a number n has that property n' does too, then every number has that property. This was the intuition that Peano, and Dedekind before him, intended to capture, through an induction *axiom,* that we could formalize

$$(P)([P(\mathbf{0}) \wedge (x_1)(P(x_1) \supset P(x_1'))] \supset (x_1)P(x_1)).$$

However, since in our languages we do not have quantification over arbitrary sets of natural numbers, the induction axiom cannot be formalized in them. Unlike Dedekind's and Peano's axiom, the induction scheme of the system we call 'Peano Arithmetic' only guarantees that when zero has a property definable in L and when a number n has it n' does too, then every number has it. So the induction scheme is really weaker than the intuitive idea behind it, that

Dedekind and Peano had in mind. In this respect, the name 'Peano Arithmetic' is somewhat misleading.

The theory PA is adequate for elementary number theory, in the sense that all of elementary number theory can be carried out within PA. This is not obvious, however, and requires proof. Notice, for example, that before the work of Gödel, it was not obvious that such simple functions as exponentiation could even be defined in the language in which PA is given, and exponentiation should certainly be regarded as a part of elementary number theory. This illustrates another respect in which the name 'Peano Arithmetic' is misleading, since it suggests that elementary arithmetic can be developed in that system in a straightforward, obvious manner.

Exercises

1. Consider a language RE^{exp} which has the same terms, connectives and quantifiers as RE but has only one predicate letter P_1^3. P_1^3xyz is interpreted as $x^y=z$ (it doesn't matter what to say about the case 0^0; you can call P_1^3xyz always false in that case, or give it the value 0 or 1). Prove that RE^{exp} defines the same sets and relations as RE. Prove also that in RE^{exp} (for the same reason as in RE) disjunction is superfluous. (Remark: half of this exercise has been done. To do the other direction, that is, defining the notions of RE in RE^{exp}, it is best to proceed in the opposite order from what appears to be natural.)

2. Cantor proved that there can be no function ϕ mapping a set onto the set of all of its subsets. Show directly that if there were such a mapping, then we would have an interpretation of '$\in$' which makes true the unrestricted comprehension schema, including the version with parameters. Remark: hence, any set-theoretical paradox that proves the inconsistency of the schema also proves the theorem of Cantor in question. The case Cantor actually used was once again the analog of Russell's paradox. Historically, this went the other way around, since Russell discovered his paradox by analyzing Cantor's proof.

3. Show that the relations $\leq_m$ and $\leq_1$ are reflexive and transitive.

4. Show that if $A \leq_m B$ and B is r.e. (recursive), then A is r.e. (recursive).

5. For the language of arithmetic, prove using the Generated Sets Theorem that if a set of axioms is r.e., then the set of theorems logically provable from it is r.e.

Lecture IX

Cantor's Diagonal Principle

A relation is called *arithmetical* if it is definable in L, the language of arithmetic. Since L contains RE, it follows that all r.e. relations are arithmetical. Also, since L contains negation, it follows that all complements of r.e. relations are arithmetical. That L contains negation also implies that the enumeration theorem fails for arithmetical sets, i.e. there is no arithmetical relation that enumerates all the arithmetical relations; similarly, there is no recursive relation that enumerates all the recursive relations.

The best way to see this is by proving a general theorem. As in the enumeration theorem for r.e. sets, if R is a two-place relation, we write R_x for $\{y: R(x, y)\}$. We give the following

Definition: Let X be a set, F be a family of subsets of X, and R a two place relation defined on X. R is said to *supernumerate* F iff for any $S \in F$, there is an $x \in X$ such that $S = R_x$. R is said to *enumerate* F iff R supernumerates F and for all $x \in X$, $R_x \in F$.

The content of the enumeration theorem is thus that there is an r.e. relation which enumerates the r.e. sets. Next we have

Cantor's Diagonal Principle: The following two conditions are incompatible:

(i) R supernumerates F
(ii) The complement of the*Diagonal Set* is in F (the Diagonal Set is $\{x \in X: R(x, x)\}$).

Proof: Suppose (i)-(ii) hold. Then by (ii) $X-\{x \in X: R(x, x)\} = \{x \in X: \sim R(x, x)\} \in F$. By (i), $\{x \in X: \sim R(x, x)\} = R_y$ for some y. But then R(y, x) iff ~R(x, x) for all $x \in X$, so in particular R(y, y) iff ~R(y, y), contradiction.

Cantor applied this lemma in the case F = power set of X to show that a set is never in 1-1 correspondence with its own power set. We can apply it to formal languages by letting F be the family of sets definable in a given language and letting R be a relation definable in the language. Unless the language is very strange indeed, (ii) will be satisfied, so (i) will be false. In the case of RE, we know from the enumeration theorem that (i) is satisfied, so it follows that (ii) fails, and therefore that negation is not definable in RE. In the case of L, on the other hand, (ii) holds, so (i) must fail. The same applies to the language Lim. Finally, if we let F be the family of recursive sets and R be an arbitrary recursive relation, (ii) clearly holds, so (i) fails and no recursive relation enumerates the recursive relations. (To see that

(ii) holds in this case, let R be a recursive relation, and let A(x, y) and B(x, y) define R and -R, respectively, in RE. Then the diagonal set is defined in RE by A(x, x), and its complement is defined by B(x, x).)

A First Version of Gödel's Theorem.

We now know enough to prove a version of Gödel's first incompleteness theorem. A sentence is said to be *undecidable* in an axiom system Γ if neither it nor its negation is a theorem of Γ, and Γ is said to be *incomplete* if some sentence is undecidable in it. We normally use the same letter to denote a set of axioms and its set of theorems. If a set of axioms is r.e., so is its set of theorems (by the Generated Sets Theorem). Similarly, if a set of axioms is arithmetical, so is its set of theorems. We have the following

Theorem: Every arithmetical set Γ of true sentences of the language L is incomplete.
Proof: Since Γ consists of true sentences, if Γ were complete, then the true sentences of L would be precisely the theorems of Γ. But as Γ is arithmetical, the set of theorems of Γ is also arithmetical, i.e. definable in L. And as we have seen earlier, the set of true sentences of L is not definable in L.

The theorem implies that every r.e. set of true sentences of L is incomplete.

In Gödel's original result the assumption that Γ is a set of true sentences was weakened, and hence Gödel's original result is stronger. An axiom system Γ in the language L is said to be *ω-consistent* if there is no formula A(x) such that Γ fi $(\exists x)A(x)$ but Γ fi $\sim A(\mathbf{0}^{(n)})$ for all n. Obviously, an axiom system consisting of true sentences of L is ω-consistent. An axiom system can be consistent without being ω-inconsistent, however. Gödel showed (in effect) that if Γ is an r.e. ω-consistent extension of Q, then Γ is incomplete. We shall not prove the full result this time, though we shall prove some related results. Rosser later showed that the assumption of ω-consistency can be weakened still further, and that no *consistent* r.e. extension of Q is complete.

One of Gödel's main intents was to prove the theorem we just gave. The reason he gave a stronger result must be understood in the light of the fact that, in the discovery and presentation of his results, he was oriented by Hilbert's program. In a nutshell, Hilbert's program demanded a proof of the consistency of the formal systems that codified the theories of classical mathematics, a proof in which, roughly, no appeal to notions or principles involving infinities was made: only so called 'finitistic' principles and methods of proof were to be employed in proofs about the properties of formal systems. The notion of truth in the standard interpretation of the language of arithmetic is a typically non-finitistic one, and hence not usable within the context of Hilbert's program. However, the notions of ω-consistency and consistency are finitistic.

More Versions of Gödel's Theorem

If $B(x_i)$ is a formula of L that defines a set S, let us say that a system Γ is *correct* for B if Γ fi $B(\mathbf{0}^{(n)})$ implies that $n \in S$, and *complete* for B if $n \in S$ implies that Γ fi $B(\mathbf{0}^{(n)})$.

Theorem: If Γ is r.e. and B defines a set which is not r.e., then Γ is not both correct and complete for B. That is, the set $S' = \{n: \Gamma \text{ fi } B(\mathbf{0}^{(n)})\}$ is different from S.
Proof: This is simply because S' is r.e., since it is defined by the formula $(\exists x)(\exists m)(Num(m,n) \wedge NSubst(\mathbf{0}^{(k)},x,[\mathbf{0}^{(1)},\mathbf{0}^{(i)}],m) \wedge Th(x))$ where Th(x) is an RE formula defining the set of theorems of Γ and k is the Gödel number of $B(x_i)$ (we can use naive substitution because we may assume that B does not contain bound occurrences of x_i).

So when S is not recursively enumerable there's a difference between being an element of S and being provably an element of S. If Γ is a true set of axioms, and thus correct, there will be an instance $B(\mathbf{0}^{(n)})$ that is true but unprovable from Γ.

This is a slight generalization of a result due to Kleene. Kleene's result was that no r.e. axiom system can be complete and correct for any formula that defines -K, and thus in particular for the formula $\sim W(x, x)$. In fact, this holds for formulae defining -S whenever S is a nonrecursive r.e. set.

Thus the interest of the theorem depends on the previous proof that there *are* r.e. nonrecursive sets (which in turn depends on the Enumeration Theorem). We can, however, state a theorem which does not depend on this fact (or on any important fact of recursion theory), and which says that any formal system must be incomplete for any formula defining the complement of *some* r.e. set:

Theorem: If Γ is an r.e. set of true axioms, then there is an r.e. set S such that if $A(x_1)$ defines -S, some instance $A(\mathbf{0}^{(n)})$ is true but unprovable.
Proof: Suppose, for a contradiction, that for every r.e. set S at least one formula $A(x_1)$ defining -S is such that Γ is complete for A. Then the following relation would be a supernumeration of the complements of the r.e. sets: $R(m,n)=\{<m,n>: m \text{ is a Gödel number of a formula } A(x_1) \text{ and m is provable of n}\}$; this relation is clearly r.e., using the same reasoning as in the proof above. But now we can use Cantor's Diagonal Principle, and conclude that the complement of the diagonal set $\{n:R(n,n)\}$ cannot be the complement of an r.e. set. But this is absurd, since $\{n:R(n,n)\}$ is an r.e. set (if B(x,y) is an RE formula that defines R, then B(x,x) defines $\{n:R(n,n)\}$).

Q is RE-Complete

Call a set Γ *RE-complete* if every true sentence of RE is a theorem of Γ, and *RE-correct* if every theorem of Γ which is a sentence of RE is true. Whenever Γ fi $A(\mathbf{0}^{(n)})$ iff $n \in S$ for all n, A(x) is said to *weakly represent* S in Γ, and S is said to be *weakly representable* in Γ if some formula weakly represents it in Γ. (We also say that S is *numerable* in Γ.) Thus, any r.e. set is weakly representable in any RE-complete and correct axiom system. Moreover, if Γ is an r.e. set which is RE-complete and correct, then the sets weakly representable in Γ are *precisely* the r.e. sets, since any set weakly representable in an r.e. axiom system Γ is r.e. (To see this, recall that if $A(x_i)$ weakly represents S in Γ, k is a Gödel number of $A(x_i)$, and Th(x) is an RE formula that defines the set of theorems of Γ, then the RE formula $(\exists x)(\exists m)(\mathrm{Num}(m,n) \wedge \mathrm{NSubst}(\mathbf{0}^{(k)},x,[\mathbf{0}^{(1)},\mathbf{0}^{(i)}],m) \wedge \mathrm{Th}(x))$ defines S.)

It turns out that Q is RE-complete and correct. Q is obviously RE-correct, because all of its axioms are true; it takes a bit more work to show that Q is RE-complete. The main fact we need to show this is

(Fact 1) Q fi $(x_1)(x_1 < \mathbf{0}^{(n)} \equiv (x_1 = \mathbf{0} \vee ... \vee x_1 = \mathbf{0}^{(n-1)}))$ for all n > 0, and
Q fi $(x_1) \sim(x_1 < \mathbf{0})$

Another useful fact is

(Fact 2) For all n, Q fi $(x_1)(x_1 = \mathbf{0}^{(n)} \vee x_1 < \mathbf{0}^{(n)} \vee \mathbf{0}^{(n)} < x_1)$

Fact 2 is not necessary to prove that Q is RE-complete, however. We shall not prove either fact, but we shall give a proof that Q is RE-complete.

It is also worth noting that a natural strengthening of Fact 2, namely that Q fi $(x_1)(x_2)$ $(x_1 = x_2 \vee x_1 < x_2 \vee x_2 < x_1)$, is false. We can show this by constructing an interpretation in which the axioms of Q are true but the statement $(x_1)(x_2)$ $(x_1 = x_2 \vee x_1 < x_2 \vee x_2 < x_1)$ is false. The domain of this interpretation is $\mathbf{N} \cup \{\alpha, \beta\}$, where α and β are two new elements not in **N**. The constant **0** still denotes 0, and successor, addition and multiplication have the same interpretations as before when restricted to the natural numbers. When the arguments include α or β, we make the following stipulations:

Successor: $\alpha' = \alpha$, $\beta' = \beta$
Addition: $n + \alpha = \alpha + n = \alpha$; $n + \beta = \beta + n = \beta$; $\alpha + \alpha = \alpha + \beta = \alpha$; $\beta + \beta = \beta + \alpha = \beta$
Multiplication: $\alpha \cdot 0 = \beta \cdot 0 = 0$; $\alpha \cdot n = \alpha$, $\beta \cdot n = \beta$ $(n > 0)$; $n \cdot \alpha = \alpha$, $n \cdot \beta = \beta$ (all n); $\alpha \cdot \alpha = \beta \cdot \alpha = \alpha$; $\beta \cdot \beta = \alpha \cdot \beta = \beta$

(where n ranges over the natural numbers) We leave it to the reader to verify that the axioms of Q are true in this interpretation, but that neither $\alpha < \beta$ nor $\beta < \alpha$ holds.

We are now ready to prove our theorem about Q.

Theorem: Q is RE-complete.
Proof: We show by induction on the complexity of sentences that every true sentence of RE is a theorem of Q.

(1) Atomic sentences. First, note that every true atomic sentence involving = is provable, since any such sentence is of the form $\mathbf{0}^{(n)} = \mathbf{0}^{(n)}$ and therefore follows from the identity axioms. Next, we show by induction on n that $A(\mathbf{0}^{(m)}, \mathbf{0}^{(n)}, \mathbf{0}^{(p)})$ is provable for all m, where p = m + n. $A(\mathbf{0}^{(m)}, \mathbf{0}, \mathbf{0}^{(m)})$ follows from axiom 4 of Q and is therefore provable. If Q fi $A(\mathbf{0}^{(m)}, \mathbf{0}^{(n)}, \mathbf{0}^{(p)})$, then by axiom 5 we see that Q fi $A(\mathbf{0}^{(m)}, \mathbf{0}^{(n+1)}, \mathbf{0}^{(p+1)})$. So Q proves all the true atomic sentences involving A; that Q proves all the true atomic sentences involving M follows similarly from the recursion axioms for multiplication.

(2) Conjunctions. Suppose A and B are theorems of Q if true. If their conjunction is true, both of them are true, so both are provable, and so is their conjunction.

(3) Disjunctions. Similar to the preceding case.

(4) Existential quantification. Suppose any statement less complex than $(\exists x)A(x)$ is a theorem of Q if true. If $(\exists x)A(x)$ is true, so must be one of its instances $A(\mathbf{0}^{(n)})$, which is then provable. But then so is $(\exists x)A(x)$.

(5) Bounded universal quantification. Suppose $(x_i < \mathbf{0}^{(n)})A$ is a true RE sentence. Then all of $A(\mathbf{0})$, ... $A(\mathbf{0}^{(n-1)})$ are true, and hence provable by the inductive hypothesis. Therefore Q fi $(x_i)((x_i = \mathbf{0} \vee ... \vee x_i = \mathbf{0}^{(n-1)}) \supset A)$, and so by Fact 1, Q fi $(x_i)(x_i < \mathbf{n} \supset A)$.

It follows, as we have seen, that the sets representable in Q are precisely the r.e. ones. We have a related result about Lim:

Theorem: Q proves, among the sentences of Lim, exactly the true ones.
Proof: This time, we show by induction on the complexity of sentences that for all sentences A of Lim, Q fi A if A is true and Q fi ~A if A is false.

(1) Atomic sentences. We have already proved half our result; we only need to show that all false atomic sentences are refutable in Q. Moreover, if we can show this for sentences involving =, the result will follow for those involving A and M: if $p \neq m + n$, then Q fi $A(\mathbf{0}^{(m)}, \mathbf{0}^{(n)}, \mathbf{0}^{(k)})$ (where k = m + n) and Q fi $\mathbf{0}^{(k)} \neq \mathbf{0}^{(q)}$, so by the uniqueness axiom for A, Q fi $\sim A(\mathbf{0}^{(m)}, \mathbf{0}^{(n)}, \mathbf{0}^{(p)})$; and similarly for multiplication.

First, observe that by axiom 1 of Q, Q fi $\mathbf{0} \neq \mathbf{0}^{(n)}$ when n > 0 (since then $\mathbf{0}^{(n)}$ is a term of the form t'). Next, note that axiom 2 is equivalent to $(x_1)(x_2)$ $(x_1 \neq x_2 \supset x_1' \neq x_2')$, so we can show by induction on k that Q fi $\mathbf{0}^{(n)} \neq \mathbf{0}^{(p)}$ where p = n + k. It follows that whenever n < m, Q fi $\mathbf{0}^{(n)} \neq \mathbf{0}^{(m)}$. Finally, by the identity axioms we have that Q fi $\mathbf{0}^{(m)} \neq \mathbf{0}^{(n)}$.

(2) Negation. Suppose A is the sentence ~B. If A is true, then B is false and by the inductive hypothesis Q fi ~B, i.e. Q fi A. If A is false, then B is true, so by the inductive hypothesis Q fi B, so Q fi ~~B (= ~A).

(3) Conjunction and disjunction. These are straightforward, and we shall only do the case of conjunction. Suppose A = (B ∧ C). If A is true, then so are B and C, so Q fi B, Q fi C, and so Q fi (B ∧ C). If A is false, then either B or C is false; suppose B is. Then Q fi ~B, so Q fi (~B ∨ ~C), and so Q fi ~(B ∧ C).

(4) Bounded universal and existential quantification. Again, we only do universal quantification, as the other case is similar. If $(x_i < \mathbf{0}^{(n)})$A is true, then A(**0**), ..., A($\mathbf{0}^{(n-1)}$) are true, so Q fi A(**0**), ..., Q fi A($\mathbf{0}^{(n-1)}$), and by Fact 1, Q fi $(x_i < \mathbf{0}^{(n)})$A. If $(x_i < \mathbf{0}^{(n)})$A is false, then A($\mathbf{0}^{(k)}$) is false for some k < n, so Q fi ~A($\mathbf{0}^{(k)}$); Less($\mathbf{0}^{(k)}$, $\mathbf{0}^{(n)}$) is a true sentence of RE, so Q fi Less($\mathbf{0}^{(k)}$, $\mathbf{0}^{(n)}$), and so Q fi $(\exists x_i)$(Less(x_i, $\mathbf{0}^{(n)}$) ∧ ~A) and Q fi ~$(x_i < \mathbf{0}^{(n)})$A.

A formula A(x) is said to *binumerate* a set S in a system Γ iff for all n, n ∈ S iff Γ fi A($\mathbf{0}^{(n)}$) and n ∉ S iff Γ fi ~A(n). If some formula binumerates S in Γ, then we say that S is *binumerable* in Γ (or *numeralwise expressible*, or *strongly representable*, or even simply *representable*). Clearly, if a set is binumerable in Γ then both it and its complement are numerable, so in particular if Γ is r.e., then any set binumerable in Γ is recursive. So not all r.e. sets are binumerable in Q. The converse, that all recursive sets are binumerable in Q, is true but not evident at this point: if S is recursive, then we have some formula A which numerates S in Q and some formula B which numerates -S, but we don't yet have a *single* formula which numerates both S and -S. The theorem we just proved shows that all sets definable in Lim are binumerable in Q, since if A(x) is a formula of Lim that defines S, then A(x) binumerates S.

The facts about weak representability in Q just given also hold for arbitrary r.e. extensions of Q that have true axioms. However, they do not hold for *arbitrary* extensions of Q, or even arbitrary r.e. extensions. For example, let Γ be an inconsistent set. Then Γ clearly extends Q, but only one set is weakly representable in Γ, namely **N** itself (since for any A and any n, A($\mathbf{0}^{(n)}$) is a theorem of Γ). Also, no set is strongly representable in Γ (since we will always have Γ fi A($\mathbf{0}^{(n)}$) and Γ fi ~A($\mathbf{0}^{(n)}$)). However, they do hold for arbitrary *consistent* r.e. extensions of Q. That is, if Γ is a consistent r.e. extension of Q, then the sets weakly representable in Γ are precisely the r.e. ones. (Again, it is easy to show that all sets weakly representable in Γ are r.e.; the hard part is showing that all r.e. sets are representable in Γ.) Moreover, as Shepherdson has shown, every r.e. set is weakly represented in Γ by some formula that actually defines it, though it is not necessarily the case that every formula that defines it weakly represents it in Γ. The proof of this result is tricky, however. It is easier to prove if we only require Γ to be ω-consistent; we will prove this later.

Let Γ be any consistent extension of Q whatsoever, and let A(x) be a formula of RE that defines a set S. Then whenever Γ fi ~A($\mathbf{0}^{(n)}$), n ∉ S. To see this, suppose Γ fi ~A($\mathbf{0}^{(n)}$) and n ∈ S. Then A($\mathbf{0}^{(n)}$) is a true sentence of RE, and so Q fi A($\mathbf{0}^{(n)}$); since Γ extends Q, Γ fi A($\mathbf{0}^{(n)}$). But then both A($\mathbf{0}^{(n)}$) and ~A($\mathbf{0}^{(n)}$) are theorems of Γ, contradicting our assumption that Γ is consistent. We thus have the following

Theorem: Any consistent extension of Q is correct for negations of formulae of RE.

Lecture X

True Theories are 1-1 Complete.

Theorem: If Γ is an r.e. set which is RE-complete and correct, then the theorems of Γ form a 1-1 complete set.
Proof: Suppose Γ is such a set. Let S be any r.e. set, and let $A(x_i)$ be a formula of RE that defines it; we can assume A to contain no bound occurrences of x_i. We define $\phi(n)$ to be the least among the Gödel numbers of $A(\mathbf{0}^{(n)})$. ϕ is recursive: its graph $\{<n,y>: \phi(n)=y\}$ is defined in RE by the formula $(\exists m \leq y)(\mathrm{Num}(m,n) \wedge \mathrm{NSubst}(\mathbf{0}^{(k)},y,[\mathbf{0}^{(1)},\mathbf{0}^{(i)}],m) \wedge \mathrm{Th}(x) \wedge (w<y)(\sim \mathrm{NSubst}(\mathbf{0}^{(k)},w,[\mathbf{0}^{(1)},\mathbf{0}^{(i)}],m)))$ (where Th(x) is an RE formula defining the set of theorems of Γ and k is the Gödel number of $B(x_i)$); notice that the use of negation in the last conjunct is legitimate, since the formula it affects is equivalent to a formula of Lim^+. Clearly ϕ is 1-1, and for any n, $n \in S$ iff $A(\mathbf{0}^{(n)})$ is true, iff Γ fi $A(\mathbf{0}^{(n)})$, iff $\phi(n)$ belongs to the set of Gödel numbers of theorems of Γ. So ϕ: $S \leq_1$ {theorems of Γ}. Finally, the set of theorems of Γ is r.e., and therefore is 1-1 complete.

It follows that the theorems of Q form a 1-1 complete set, and hence a nonrecursive set. In fact, we can prove the stronger result that if Γ is any r.e. set of true axioms, the set of theorems of Γ is 1-1 complete, as we will see shortly.

Let us say that a formula A(x) of the language of arithmetic *nicely* weakly represents a set S in a theory Γ if it weakly represents S in Γ and also defines S. We may similarly define "nicely *strongly* represents". Similarly, a formula $A(x_1, ..., x_n)$ nicely weakly (strongly) represents an n-place relation R in Γ if it both weakly (strongly) represents R in Γ and also defines R.

It follows from our results of the last lecture that any r.e. set is nicely weakly representable in Γ whenever Γ is true and extends Q. We shall now see that the latter requirement, that Γ extend Q, is unnecessary: any r.e. set is nicely weakly representable in any set Γ of true axioms of the language of arithmetic. Before proving this, we shall need the following theorem:

Deduction Theorem: For any set Γ and any sentences A and B (of any first-order language), if Γ, A fi B then Γ fi A ⊃ B. (Here, Γ, A fi B means $\Gamma \cup \{A\}$ fi B.)
Proof: Suppose Γ, A fi B, and let M be a model of Γ (i.e. an interpretation in which every element of Γ is true). If A is true in M, then M is a model of $\Gamma \cup \{A\}$, and so by the soundness of the predicate calculus B is true in M, so A ⊃ B is true in M. If A is false in M, then again A ⊃ B is true in M. So A ⊃ B is true in all models of Γ, and therefore by the completeness theorem Γ fi A ⊃ B.

The proof we just gave is model-theoretic; however, it is possible to establish the deduction theorem proof-theoretically, by showing how to transform any proof of B from $\Gamma \cup \{A\}$ into a proof of $A \supset B$ from Γ. Such a proof-theoretic argument might be more satisfying, since the model-theoretic argument merely shows that whenever a proof of A from $\Gamma \cup \{A\}$ exists, then a proof of $A \supset B$ from Γ exists, and leaves it an open question whether there is any direct way to transform the former into the latter.

Now let $A(x_1)$ be any sentence of RE that defines a set S; we claim that the formula $Q \supset A(x_1)$ nicely weakly represents S in any system Γ with true axioms. (By Q we mean here some conjunction of the axioms of Q; such a conjunction exists because Q's axioms are finite in number.) Clearly, $Q \supset A(x_1)$ defines S; we must show that $Q \supset A(\mathbf{0}^{(n)})$ is a theorem of Γ iff $n \in S$. First, suppose that $n \in S$. Since Q is RE-complete, $Q \vdash A(\mathbf{0}^{(n)})$. Clearly, $\Gamma, Q \vdash A(\mathbf{0}^{(n)})$. By the deduction theorem, $\Gamma \vdash Q \supset A(\mathbf{0}^{(n)})$. Conversely, suppose $\Gamma \vdash Q \supset A(\mathbf{0}^{(n)})$. Then since Γ is true, $Q \supset A(\mathbf{0}^{(n)})$ is also true. But Q is true, so $A(\mathbf{0}^{(n)})$ is true. But $A(\mathbf{0}^{(n)})$ is a sentence of RE, and is true iff $n \in S$. So $n \in S$, and we are done. Therefore we have established this

Theorem: If Γ is a set of true sentences of L, then every r.e. set is nicely weakly representable in Γ.

Corollary: For such a Γ, the set of all theorems of Γ is a set to which all r.e. sets are 1-1 reducible. If Γ is r.e., then Γ's theorems form a 1-1 complete set.

Note that, while every r.e. set is nicely weakly representable in such a Γ, we have not shown that every r.e. set is nicely representable by every formula that defines it, or even every RE formula that defines it. If we require Γ to extend Q, on the other hand, then every RE formula that defines S represents it in Γ, because any such Γ is RE-complete and correct.

Church's Theorem

Note that the empty set Ø is trivially a set of true axioms; it follows from our theorem that every r.e. set is nicely weakly representable in Ø, and therefore that Ø's theorems, i.e. the valid formulae of L, form a 1-1 complete set (since Ø is r.e.). So the set of valid formulae of L is not recursive (when the set of theorems of a theory is not recursive, the theory is called *undecidable*; this use of the term 'undecidable' must not be confused with the use we are familiar with, in which the term applies to sentences). This is called *Church's Theorem*. Note that whether a formula is valid does not depend on the interpretation of the nonlogical vocabulary, and therefore that Church's theorem does not depend on the interpretation of the predicates and function symbols of L: for any language with two 3-place predicates, one 2-

place predicate, a constant, and a 1-place function symbol, the set of valid formulae of that language is undecidable, and indeed 1-1 complete.

(Actually, there are two versions of Church's theorem, depending on whether the identity predicate is regarded as a logical symbol. We have been regarding it as nonlogical; when it is regarded as logical, so that the identity axioms are taken as logical axioms, Church's Theorem states that the set of valid formulae (in the present sense of "valid") of a language with two 3-place predicates, a constant and a 1-place function symbol is an undecidable set. The proof is exactly the same.)

Clearly, this result also applies to first-order languages extending L. In fact, we can use a few tricks to show that it also applies to some languages smaller than L. We already know that the constant **0** is redundant in L, since the formula $x = \mathbf{0}$ is equivalent to $(y)A(x, y, x)$. We can also eliminate the successor function sign, since the graph of the successor function is defined by $(\exists z)[(w)M(w, z, w) \wedge A(x, z, y)]$. Reasoning in this way, we can show that Church's theorem applies to any language with two 3-place predicates. Using a trick that we saw in an exercise, we can eliminate these predicates in favor of a single 3-place predicate defining the graph of the exponentiation function plus a constant for 0 and a 1-place function letter for successor. Using still more devious tricks, we can show that Church's theorem applies to a language which contains only a single 2-place predicate. However, we cannot go any further: the set of valid formulae of a language with only 1-place predicates (with or without identity) is recursive.

(The reasoning we have given is not wholly rigorous. For one thing, while we can find a language K which is properly included in L and which has the same expressive power, we must also show that the above remarks about Q hold for some translation of Q into K. We shall not enter into these considerations here; they will be addressed when we prove the Tarski-Mostowski-Robinson theorem.)

The name "Church's Theorem", though traditional, does not make full justice to its discoverers, since Turing proved the same theorem in his famous original paper on computability; "the Church-Turing theorem" would be a more appropriate name. Also Gödel, in his paper on the incompleteness theorems, stated a very closely related result which, from our vantage point, establishes Church's theorem; but Gödel may not have realized that this was a consequence of his result. Gödel's result is that for any formal system with a primitive recursive set of axioms we can always find a sentence which is not quantificationally valid, but such that the statement that it is not quantificationally valid is not provable in the system.

Complete Theories are Decidable

Theorem: Consider a language in which the set of all sentences is recursive, and let Γ be a set of axioms in this language. If Γ is r.e. and the set of closed theorems of Γ is not

recursive (i.e., undecidable), Γ is incomplete.
Proof: We first give an informal proof using Church's Thesis. We can assume that Γ is consistent, for otherwise the set of theorems of Γ is simply the set of all sentences, which by hypothesis is recursive. Suppose Γ is complete, and let A be any expression. We shall show that either A or ~A is a theorem of Γ. Since the set of sentences of the language of Γ is recursive, we can tell effectively whether A is a sentence. If A is a sentence, then since Γ is complete, either A or ~A is a theorem of Γ. So to see whether A is a theorem of Γ or not, simply run through all the proofs from Γ. If you encounter a proof of A, then A is a theorem; if you encounter a proof of ~A, then A is not a theorem; and since Γ is complete, you will eventually encounter a proof of A or of ~A, so this procedure will eventually terminate. So we can effectively tell whether an arbitrary expression is a theorem of Γ, and so the set of theorems of Γ is recursive.

We now reason more formally. Suppose Γ is complete; again, we may suppose that Γ is consistent. Since Γ is r.e., the set of theorems of Γ is defined in RE by some formula Th(x). Since Γ is complete, an expression A is a nontheorem of Γ just in case either A is a nonsentence or ~A is a theorem. Thus the set of nontheorems of Γ is defined by an RE formula $N(x) \vee (\exists y)(Neg(x, y) \wedge Th(y))$, where N(x) defines the set of nonsentences of the language of Γ and Neg(x, y) defines the relation *y is the negation of x*. We know that such an RE formula N(x) exists because by hypothesis the set of sentences of the language of Γ is recursive, and Neg(x, y) is easily defined using concatenation. It follows that the set of theorems of Γ is recursive.

It was a while before logicians realized this fact, despite the simplicity of its proof. This may be because the decision procedure given is not intuitively a "direct" one, i.e. a procedure which determines whether A is a theorem of Γ or not by examining A itself.

Note that the requirement that Γ be r.e. is essential here, as is seen by letting Γ equal the set of true sentences of the language of arithmetic: the set of theorems of Γ, i.e. Γ itself, is certainly not recursive, but it is complete, and the set of sentences of L is recursive.

Replacing Truth by ω-Consistency

Let us now state some incompleteness results about ω-consistent, but not necessarily true formal systems. *School* is the set of (consequences of) true atomic sentences of the language of arithmetic and of true negations of atomic sentences.

Theorem: If Γ is arithmetical, ω-consistent and contains School, Γ is incomplete.
Proof: Suppose Γ was complete. Since Γ is arithmetical, it does not coincide with the set of truths, so there must be a sentence A which is either true but unprovable or false but provable from Γ. If A is false but provable, since the system is complete, ~A must be true

but unprovable (ω-consistency implies consistency), and so is any prenex version of it. Take the shortest example B of a true but unprovable sentence: among the prenex true unprovable sentences, take one with the shortest number of quantifiers. This sentence must have some quantifiers, since if Γ contains School, all sentences made up from atomic sentences by use of connectives are decidable in Γ. The first quantifier in B will not be existential, because if it were, some instance of it would be true, and thus a shorter true but unprovable statement (unprovable, because if it were provable so would be its existential generalization). So the first quantifier in B must be universal, which means that all its instances must be true and provable, since they are shorter. Since Γ is complete, ~B must also be provable. But this contradicts the hypothesis that Γ is ω-consistent.

The theorem does not hold if we replace ω-consistency by consistency. There are consistent arithmetical, even recursive, sets of sentences containing School (and extensions of School) which are complete. An example is the set of truths in the structure of the real numbers with a constant for 0 and function letters for successor, addition and multiplication. This is naturally a complete set, which, by a celebrated result of Tarski, is recursive.

On the other hand, if Γ is r.e. and contains Q (not just School), the hypothesis of ω-consistency in the theorem can be weakened to consistency. We will prove this later. Now we can establish the following

Theorem: If Γ is r.e., ω-consistent and contains Q, then Γ is incomplete.
Proof: We know that if Γ contains Q, it is RE-complete and hence that it is correct for negations of RE sentences. Let $A(x_1)$ be an RE-formula that defines K. Then $\sim A(x_1)$ defines -K. Since Γ is r.e., the set $\{n: \sim A(\mathbf{0}^{(n)})$ is a theorem of Γ$\}$ is r.e. and thus it does not coincide with -K. Since Γ is correct for negations of RE sentences, there is no false provable statement of the form $\sim A(\mathbf{0}^{(n)})$, so there must be a statement of that form which is true but unprovable. This does not yet tell us that $A(\mathbf{0}^{(n)})$ is also unprovable, since Γ need not be a set of true sentences. Let's take again, from among the prenex true unprovable sentences of the form $\sim A(\mathbf{0}^{(n)})$, one with the shortest number of quantifiers. This sentence must have some quantifiers, for the same reason as before. And it cannot begin with an existential, again for the same reason. So the sentence must be of the form (x)C(x), and such that all of its instances are provable. Now, $A(\mathbf{0}^{(n)})$ cannot be provable, since it is equivalent to ~(x)C(x) and that would contradict ω-consistency.

The Normal Form Theorem for RE.

Although our incompleteness results are quite powerful, it is a bit unsatisfying that we have not been able to construct effectively examples of undecidable sentences. One way to do this uses a result that we will prove now.

Normal Form Theorem for RE: Every r.e. relation is definable by a formula of the form $(\exists y)B$, where B is a formula of Lim in which there are no occurrences of negation.

This is a version of a theorem of Kleene, though what he showed was something weaker, namely that every r.e. formula is defined by $(\exists y)B$ for some formula B that defines a *primitive recursive* relation.

To prove it, we prove by induction on the complexity of formulae that every RE formula A is equivalent to a formula $(\exists y)B$, with B a formula of Lim without negation.

(1) A is atomic. Then A is already in Lim without negation, and is equivalent to $(\exists y)A$, where y is a variable that does not occur in A. (That is, we get a formula of the required form by adding a vacuous existential quantifier. Note that A is also equivalent to $(\exists y)(y = y \wedge A)$, so the use of vacuous quantifiers is not really necessary.)

(2) A is $A_1 \vee A_2$. By the inductive hypothesis, A_1 and A_2 are equivalent to formulae $(\exists y)B_1$ and $(\exists y)B_2$, where B_1 and B_2 are formulae of Lim without negation; so A is equivalent to $(\exists y)B_1 \vee (\exists y)B_2$, which is equivalent to $(\exists y)(B_1 \vee B_2)$, which is of the required form.

(3) A is $A_1 \wedge A_2$. Again, A_1 and A_2 are equivalent to $(\exists y)B_1$ and $(\exists y)B_2$, with B_1 and B_2 formulae of Lim without negation, so A is equivalent to $(\exists y)B_1 \wedge (\exists y)B_2$. By rewriting bound variables, we see that A is equivalent to $(\exists z)B_1' \wedge (\exists w)B_2'$, where B_1' and B_2' come from B_1 and B_2 by changing bound occurrences of y to z (or w) throughout. This is in turn equivalent to $(\exists z)(\exists w)(B_1' \wedge B_2')$. This is not yet in the required form, since we have two unbounded quantifiers. However, this is equivalent to $(\exists y)(\exists z < y)(\exists w < y)(B_1' \wedge B_2')$, which is of the required form.

[The usual way to reduce this pair of unbounded quantifiers to a single quantifier uses the pairing function; however, the present approach is simpler.]

(4) A is $(\exists z)A_1$. Then A_1 is equivalent to $(\exists w)B_1$ for some formula B_1 of Lim without negation, and so A is itself equivalent to $(\exists z)(\exists w)B_1$. As in (3), this is equivalent to $(\exists y)(\exists z < y)(\exists w < y)B_1$.

(5) A is $(z < t)A_1$. This is the trickiest case. Let A_1 be equivalent to $(\exists w)B_1$, with B_1 in Lim without negation. A is equivalent to $(z < t)(\exists w)B_1$. We claim that this is equivalent to $(\exists y)(z < t)(\exists w < y)B_1$. To see this, first fix an assignment of values to the free variables. Suppose $(\exists y)(z < t)(\exists w < y)B_1$ holds; then $(z < t)(\exists w < n)B_1$ holds for some n, so *a fortiori* $(z < t)(\exists w)B_1$ holds. Conversely, suppose $(z < t)(\exists w)B_1$ holds, and let n be the denotation of t. Then for each $m < n$, $(\exists w)B_1(w, m)$ holds, so $B_1(k, m)$ holds for some particular k. For each $m < n$, pick a k_m such that $B(k_m, m)$ holds. Since there are only finitely many k_m's, there is a number p such that $p > k_m$ for all $m < n$. So for all $m < n$, there is a $k < p$ (namely k_m) such that $B_1(k, m)$ holds. Therefore, $(z < t)(\exists w < p)B_1$ holds, and so $(\exists y)(z < t)(\exists w < y)B_1$ holds. This completes the proof.

The normal form theorem yields very strong results when combined with the

enumeration theorem. From the normal form theorem, we see that the relation W(e, x) is defined by some formula $(\exists y)T(e, x, y)$, where T is a formula of Lim without negation. T is a particular formula, and therefore has a *fixed* number of bounded quantifiers and *no* unbounded quantifiers. It follows that there is a single fixed n such that every r.e. set is defined by some formula with at most n bounded quantifiers and only one unbounded quantifier. This leaves open the possibility that n must be very large; in fact, however, it is known that n can be made rather small.

Whenever we have a normal form theorem, we can combine it with the enumeration theorem to get an analogous result. The most spectacular enumeration theorem we have mentioned is that proved by Matijasevic (building on earlier work by Davis, Putnam and Julia Robinson), that every r.e. set or relation is definable by a formula of the form $(\exists x_1)...(\exists x_n)\ t_1 = t_2$, where t_1 and t_2 are terms involving only the function symbols ', + and · (and the constant **0** and the variables). Note that the formula $t_1 = t_2$ is simply a polynomial equation. Thus the decision problem for any r.e. set is equivalent to the decision problem for some polynomial equation with integer coefficients. The Matijasevic theorem alone does not give us an indication of how large the degree of such equations can be, or of how many variables they may contain. If we apply the enumeration theorem, however, we see that the relation W(e, x) is defined by some *particular* formula $(\exists x_1)...(\exists x_n)\ t_1 = t_2$, whose free variables are e and x. Let us indicate the free variable e by writing this formula as $(\exists x_1)...(\exists x_n)\ t_1(e) = t_2(e)$. Every r.e. set is therefore defined by the formula $(\exists x_1)...(\exists x_n)\ t_1(\mathbf{0}^{(e)}) = t_2(\mathbf{0}^{(e)})$, for some particular e. So not only is the decision problem for every r.e. set equivalent to the problem of solving some polynomial equation; we can also simultaneously bound the number of variables and the degree of the polynomial.

An immediate application of the normal form theorem is in the proof of the following result:

Theorem: If Γ is r.e., ω-consistent and contains Q, then Γ is complete and correct for the set of all formulae of the form $(\exists y)B$, where B is a formula of Lim without negation.
Proof: Completeness: If $(\exists y)B$ is true, then $B(\mathbf{0}^{(n)})$ is true, for some n. So $B(\mathbf{0}^{(n)})$ will be provable, since Q proves all the true sentences of Lim, and Γ contains Q. Therefore, $(\exists y)B$ will be provable too. Correctness: Suppose $(\exists y)B$ is provable but false. Then all the negations of its instances will be true: ~A(**0**), ~A(**0'**)... So these are all provable, again because Q is complete for the true sentences of Lim. But this contradicts ω-consistency.

Corollary: If Γ is r.e., ω-consistent and contains Q, then every r.e. set is nicely weakly representable in Γ.

Corollary: If Γ is r.e., ω-consistent and contains Q, then the theorems of Γ form a 1-1 complete set.

The hypothesis of ω-consistency in these results could have been replaced by that of consistency, but the corresponding proof is much trickier.

Exercises

1. Prove that if an r.e. set S_1 is 1-1 complete, there is an infinite r.e. set S_2 disjoint from S_1.

2. Prove that a nonempty set is r.e. iff it is the range of a total recursive function.
Note: this result is the origin of the term 'recursively enumerable'. That is, a nonempty set S is r.e. iff there is a recursive function ϕ that enumerates it, i.e. $S=\{\phi(0), \phi(1),...\}$.

3. The Goldbach conjecture is the statement that every even integer greater than 2 is the sum of two primes. Show that this conjecture can be written in the form (x)A, where A is in Lim. Suppose that the conjecture, written in this form, is undecidable in the system we have called Peano Arithmetic. What, if anything, would follow regarding the truth of the Goldbach conjecture itself? (Explain your answer; if nothing follows, explain why, or if something does follow, explain what follows and why.)

4. *Proper* substitution, as opposed to what we have called 'naive substitution', is the substitution of a term for a variable, subject to the following restrictions. Only *free* occurrences of the variable x_i are to be replaced by the term t; and the substitution is improper if any variable occurring in t becomes bound in the result. Define proper substitution in RE, that is, PSubst(m_1,m_2,v,m). where m_2 is the result of a proper substitution of the term m for free occurrences of the variable v in m_1. Use the following fact: an occurrence of a variable x_i within a term is bound in a formula iff the formula is a concatenation of three sequences m, n and p, where the occurrence in question is in the part corresponding to n, and n is (the Gödel number of) a formula beginning with (x_i). m and/or p are allowed to be empty. (This is a form of the usual definition.) Another treatment of proper substitution, which is perhaps more elegant, will be sketched later. It should be clear from the preceding why naive substitution is simpler, at least if this is the treatment adopted.

Lecture XI

An Effective Form of Gödel's Theorem

Recall that Γ is ω-consistent if we never have Γ fi (∃x)A(x) and Γ fi ~A(**n**) for all n. Γ is said to be *ω-complete* if whenever Γ fi A(**n**) for all n, Γ fi (x)A(x). Γ is *ω-inconsistent* iff it is not ω-consistent, and similarly for *ω-incomplete*.

Let us call a formula A Σ_1 if A is of the form (∃y)B, where B is a formula of Lim, and Π_1 if it is of the form (y)B for B a formula of Lim. Note that a negation ~(∃y)B of a Σ_1 formula is equivalent to (y)~B, which is Π_1, and that each Π_1 formula (y)B is equivalent to a negation ~(∃y)~B of a Σ_1 formula (and these equivalences are provable). We sometimes use the terms Σ_1 and Π_1 loosely to refer to formulae that are equivalent to formulae that are Σ_1 or Π_1 in the strict sense; we also refer to a *set* or *relation* as Σ_1 (or Π_1) if it is defined by some Σ_1 (or Π_1) formula. It follows from the normal form theorem for RE that the r.e. sets are precisely the Σ_1 sets and the complements of r.e. sets are precisely the Π_1 sets.

We sometimes write Σ_1^0 for Σ_1 and Π_1^0 for Π_1. The superscript zero indicates that the unbounded quantifier ranges over numbers. Other superscripts are possible; in general, when we talk about a Σ_n^m or Π_n^m formula, m indicates the type of the variables in the unbounded quantifiers, and the n indicates the number of alterations between unbounded universal and unbounded existential quantifiers. This will be made more precise later on in the course.

Suppose Γ extends Q. If B is a sentence of Lim, then as we saw in Lecture IX, if B is true, then B is a theorem of G. So let (∃y)B(y) be a true Σ_1 sentence. Since it is true, B($\mathbf{0}^{(n)}$) is true for some n and therefore is a theorem of Γ; but B($\mathbf{0}^{(n)}$) logically implies (∃y)B(y), so (∃y)B(y) is also a theorem of Γ. So every true Σ_1 sentence is a theorem of Γ; in short, Γ is Σ_1-complete. If Γ is also consistent, then it is Π_1-correct, i.e. every Π_1 sentence provable in Γ is true. To see this, let A be a Π_1 sentence provable in Γ. If A is false, then ~A is true; but ~A, being the negation of a Π_1 sentence, is provably equivalent to a Σ_1 sentence, and is therefore provable in Γ, since Γ is Σ_1-complete. But then both A and ~A are theorems of Γ, and so Γ is incomplete. So a consistent extension of Q is both Σ_1-complete and Π_1-correct.

Moreover, as we saw in the last lecture, every ω-consistent system extending Q is Σ_1-correct. Recall the argument: suppose Γ is such a system, and suppose it proves a false Σ_1 sentence (∃y)B(y). Since that sentence is false, B($\mathbf{0}^{(n)}$) is false for all n, and therefore, since Γ extends Q, Γ fi ~B($\mathbf{0}^{(n)}$) for all n, contradicting Γ's ω-consistency. So any ω-consistent extension of Q is Σ_1-complete and correct.

We can now prove an effective version of Gödel's theorem.

Effective Form of Gödel's Theorem: Let Γ be an r.e. extension of Q. Then we can find

effectively a Π_1 formula A such that

(1) If Γ is consistent, then A is true but unprovable in Γ

and

(2) If Γ is ω-consistent, then ~A is also unprovable in Γ.

Proof: Since W(e,x) is r.e., it is definable by a Σ_1 formula $(\exists y)L(e,x,y)$ which can be effectively found from the original RE formula and through the (effective) proof of the normal form theorem for RE. So K is r.e., and it is definable by $(\exists y)L(x,x,y)$, which is Σ_1. -K is then defined by the Π_1 formula $(y)\sim L(x,x,y)$. The set $\{n: (y)\sim L(\mathbf{0}^{(n)},\mathbf{0}^{(n)},y)$ is provable from $\Gamma\}$ is r.e. (for the known reasons), and an RE formula that defines it can be found effectively; therefore also its Gödel number e can be found effectively. Then for all n, $(\exists y)L(\mathbf{0}^{(e)},\mathbf{0}^{(n)},y)$ is true iff $(y)\sim L(\mathbf{0}^{(n)},\mathbf{0}^{(n)},y)$ is provable. So $(\exists y)L(\mathbf{0}^{(e)},\mathbf{0}^{(e)},y)$ is true iff $(y)\sim L(\mathbf{0}^{(e)},\mathbf{0}^{(e)},y)$ is provable. Then the Π_1 formula $(y)\sim L(\mathbf{0}^{(e)},\mathbf{0}^{(e)},y)$ cannot be provable from Γ, because given that Γ is a consistent extension of Q, Γ is Π_1-correct, so $(y)\sim L(\mathbf{0}^{(e)},\mathbf{0}^{(e)},y)$ would be true and so would be the equivalent formula $\sim(\exists y)L(\mathbf{0}^{(e)},\mathbf{0}^{(e)},y)$, and on the other hand if $(y)\sim L(\mathbf{0}^{(e)},\mathbf{0}^{(e)},y)$ were provable $(\exists y)L(\mathbf{0}^{(e)},\mathbf{0}^{(e)},y)$ would be true. We therefore may take A to be $(y)\sim L(\mathbf{0}^{(e)},\mathbf{0}^{(e)},y)$. A is not provable, and therefore $\sim(\exists y)L(\mathbf{0}^{(e)},\mathbf{0}^{(e)},y)$ and A itself are true.

Now suppose that Γ is ω-consistent. Then Γ is Σ_1-correct. ~A is logically equivalent to a false Σ_1 sentence, and is therefore not a theorem of Γ.

This is an informal argument in the sense that it appeals to the intuitive notion of computability or effectiveness. We could now give a more formal proof without making this appeal. It will be easier to give such a proof later, once we have some more results. We can note here that the effectiveness of the construction of A depends on the fact that we use K, for which the number e with appropriate properties can be effectively found from every Γ. Not every r.e. nonrecursive set would have served the purpose of effectiveness, since as we will show later, for some such sets the corresponding Gödel sentences cannot be effectively found..

The hypothesis of our effective form of Gödel's theorem is already quite weak; in fact, we can weaken it a bit more. In particular, the condition that Γ extends Q can be weakened. The only fact about Q needed in proving that Q is RE-complete and correct for negations of RE sentences is Fact 1, along with the fact that all sentences of School are provable in Q. So these are the only facts needed to show, using the normal form theorem, that Q is Σ_1-complete and Π_1-correct. So the theorem will still hold if Q is replaced by any theory containing School for which Fact 1 holds.

Gödel's Original Proof.

The following is, nearly enough, Gödel's own presentation of the first incompleteness

theorem. Let Γ be an r.e. system containing Q. Consider the relation Prov(x, y) which holds if y is provable of x, i.e. if the result of replacing all (free) occurrences of x_1 in the formula (coded by) y by the numeral for x is provable. We can take Prov to be Σ_1, since it can be written out in RE; so it is of the form $(\exists z)L(x, y, z)$ for some formula L(x, y, z) of Lim. Consider the formula $\sim Prov(x_1, x_1)$; it has some Gödel number m. Let G be the sentence $\sim Prov(\mathbf{0}^{(m)}, \mathbf{0}^{(m)})$, i.e. $\sim(\exists z)L(\mathbf{0}^{(m)}, \mathbf{0}^{(m)}, z)$. Suppose G is provable; that is, suppose the formula $\sim Prov(x_1, x_1)$ is provable of m. Then $L(\mathbf{0}^{(m)}, \mathbf{0}^{(m)}, \mathbf{0}^{(k)})$ holds for some k; since this is a true sentence of Lim, it is provable, and so $(\exists z)L(\mathbf{0}^{(m)}, \mathbf{0}^{(m)}, z)$ is provable. But G, which we are supposing to be provable, is just the sentence $\sim(\exists z)L(\mathbf{0}^{(m)}, \mathbf{0}^{(m)}, z)$. So if our system is consistent, G is not provable after all, i.e. $\sim Prov(x_1, x_1)$ is not provable of m. But what G says is that the formula with Gödel number m, namely $\sim Prov(x_1, x_1)$, is not provable of m; so G is true. Therefore G is true but unprovable. As long as the system is Σ_1-correct, its negation is not provable either, and as we have seen, it suffices for this that the system be ω-consistent.

Presented in this way, the proof seems rather tricky, and the undecidable sentence is produced in a very devious way. As we have previously presented it, Gödel's theorem should seem more like the inevitable outcome of the Russell paradox. In fact, there is a way of viewing Gödel's original proof which makes it look this way.

Recall the proof that any fully classical language lacks its own satisfaction predicate. If L, for example, has a predicate Sat(y,x) which defines the relation {<x, y>: y codes a formula which x satisfies}, then L has a predicate Het(x) = ~Sat(x, x), which defines the set of (Gödel numbers of) heterological formulae. But then if we ask whether Het(x) is itself heterological, we can derive a contradiction. (Indeed, that there is no formula defining the set of heterological formulae follows directly from the inconsistency of the instance $(\exists y)(x)(x \in y \equiv x \notin x)$ of the unrestricted comprehension scheme, as we saw before.) It follows from the indefinability of satisfaction that the formula Prov does not define satisfaction.

We can show directly that the Gödel sentence G is true but unprovable, in a way that imitates the reasoning of the last paragraph. Call a formula *Gödel heterological* if is not provable of itself; the formula $\sim Prov(x_1, x_1)$ defines the set of Gödel heterological formulae. Let us write this formula as $GHet(x_1)$. Now we ask, is "Gödel heterological" Gödel heterological? The statement that "Gödel heterological" is Gödel heterological is simply the statement $GHet(\mathbf{0}^{(m)})$, where m is the Gödel number of $GHet(x_1)$. Rather than leading to a contradiction, our question has a definite answer "yes". Suppose "Gödel heterological" were not Gödel heterological, i.e. that $GHet(x_1)$ were provable of m. If $GHet(x_1)$ is provable of m, then $Prov(\mathbf{0}^{(m)}, \mathbf{0}^{(m)})$ is a theorem of Q and therefore of any system extending Q; note that $GHet(\mathbf{0}^{(m)})$ is simply the negation of $Prov(\mathbf{0}^{(m)}, \mathbf{0}^{(m)})$. So if $GHet(x_1)$ is provable of itself, then our system is inconsistent, since both $Prov(\mathbf{0}^{(m)}, \mathbf{0}^{(m)})$ and its negation are provable; so if our system is consistent, then $GHet(x_1)$ is not provable of itself, i.e. is Gödel heterological. This is simply to say that $GHet(\mathbf{0}^{(m)})$ is true but not

provable. A similar argument shows that ~GHet($\mathbf{0}^{(m)}$) is also unprovable, provided that the system is ω-consistent. Finally, note that GHet($\mathbf{0}^{(m)}$) is simply the sentence G of the last paragraph. So we have really presented Gödel's own proof, but with a different exposition than is usual.

An analogy is often drawn between the unprovability of the Gödel sentence and the liar paradox. From the present exposition, we see that the analogy with the heterological paradox is even closer. In fact, all we really need in order to see that ~Prov($\mathbf{0}^{(m)}$, $\mathbf{0}^{(m)}$) is true but unprovable is to notice that ~Prov($\mathbf{0}^{(m)}$,$\mathbf{0}^{(m)}$) says "'Is not provable of itself" is not provable of itself', i.e. "'Gödel heterological" is Gödel heterological': it is not essential to our proof (though we may observe this afterwards) that it says "I am not provable".

That there is an analogy both to the heterological paradox and to the liar paradox is no accident, since the heterological paradox is really a special case of the liar paradox. The heterological paradox involves the sentence "'Is not true of itself" is not true of itself'. To say that "is not true of itself" is not true of itself is simply to say that the sentence "'Is not true of itself" is not true of itself' is not true, so this sentence says of itself that it is not true — that is, it is a liar sentence.

The Uniformization Theorem for r.e. Relations.

Definition: A *uniformization* of a binary relation R is a relation S such that:

(i) $S \subseteq R$
(ii) S and R have the same domain, i.e. for any x, there is a y such that R(x, y) iff there is a y such that S(x, y)
(iii) S is single valued, i.e. every x bears S to at most one y (i.e. S is the graph of a partial function).

We can think of a relation R(x, y) as a *many valued* function, with x as the argument and any y such that R(x, y) as one of the values for the argument x. Then, for example, an r.e. relation is a partial recursive many-valued function. A uniformization of a many valued function is a single valued function with the same domain.

S can be regarded as a choice function, i.e. S chooses, for each x, something that x bears R to. This definition can be extended to n+1-place relations in an obvious way.

A uniformization theorem in general says that any relation in a particular class C can be uniformized by a relation in C. If this is so, then C is said to have the *uniformization property*. Note that the class of all relations on the natural numbers has the uniformization property. Taking the 2-place case for simplicity, any relation R(x, y) is uniformized by the relation $R(x, y) \wedge (z < y)\sim R(x, z)$ (i.e. the relation $y = \mu z R(x, z)$). This also shows that the class of recursive relations has the uniformization property, since the relation $y = \mu z R(x, z)$

is recursive if R is; the same applies to the class of relations definable in Lim and to the arithmetical relations. And this argument can easily be generalized to relations of more than two places.

A trickier case is the class of r.e. relations. The above argument will not show that this class has the uniformization property, since $y = \mu z R(x, z)$ will not in general be r.e. when R is. To see this, let X be any r.e. set which is not recursive, and let R be the r.e. relation $\{<x, y>: (y = 0 \wedge x \in X) \vee y = 1\}$. Let S be the relation $y = \mu z R(x, z)$. If $x \in X$, then $S(x) = 0$, but if $x \notin X$, then $S(x) = 1$. So if S is r.e., then -X can be defined in RE by $S(x, \mathbf{0}')$. But by hypothesis X is nonrecursive, so S is not r.e.

However, we can use a somewhat trickier proof to show that uniformization holds for the r.e. relations. Let R be any 2-place r.e. relation, and let F(x, y) be some formula of RE that defines it. By the normal form theorem, we can take F to be $(\exists z)L(x, y, z)$ for some formula L of Lim. $(\exists z)L(x, y, z)$ is equivalent to $(\exists w)(y = K_1(w) \wedge L(x, K_1(w), K_2(w)))$. We can now define a uniformizing relation S by $(\exists w)(y = K_1(w) \wedge L(x, K_1(w), K_2(w)) \wedge (u < w) \sim L(x, K_1(u), K_2(u)))$ (intuitively, w is the smallest code of a pair [y,z] for which L(x,y,z) holds). Since L is a formula of Lim, the formula defining the uniformizing relation is a Σ_1 formula and so S is an r.e. relation. This can be generalized to n+1-place r.e. relations in a fairly obvious way. So we have proved the

Uniformization Theorem for r.e. Relations: The class of r.e. relations has the uniformization property.

Corollary: Every r.e. relation can be uniformized by a partial recursive function with the same domain.

The Normal Form Theorem for Partial Recursive Functions.

An application of the proof of the uniformization theorem for r.e. relations is a normal form theorem for partial recursive functions, due to Kleene. Let ϕ be any partial recursive function, and let R be its graph. R is defined by some Σ_1 formula $(\exists z)L(x, y, z)$. As in the proof of the uniformization theorem, we see that R is defined by $y = K_1(\mu w L(x, K_1(w), K_2(w)))$. Since $L(x, K_1(w), K_2(w))$ is a formula of Lim, and K_1 is a function whose graph is definable in Lim, we have the following:

Normal Form Theorem for Partial Recursive Functions: Every n-place partial recursive function is of the form $U(\mu w R(x_1, ..., x_n, w))$ for some relation R definable in Lim and some function U whose graph is definable in Lim.
Proof: The case n = 1 was just proved, and the general case is proved similarly.

This is not exactly what Kleene originally proved; he only required R and U to be primitive recursive.

It is important not to forget the U; it is not true that every partial recursive function is of the form $\mu w R(x_1, ..., x_n, w)$ for some R definable in Lim. Even if we allow R to be an arbitrary recursive relation, this is still wrong. If, on the other hand, we require the function ϕ to be total, then $\phi(x_1, ..., x_n)$ is $\mu y(\phi(x_1, ..., x_n) = y)$, so we can drop the U by taking R to be ϕ's graph; but then we cannot require R to be definable in Lim.

Lecture XII

An Enumeration Theorem for Partial Recursive Functions

We can use the uniformization theorem for r.e. relations to prove an enumeration theorem for partial recursive functions. First we recall that we have a general version of the Enumeration Theorem for n-place r.e. relations. That is, there is an n+1-place relation $W^{n+1}(e, m_1, \ldots, m_n)$ that enumerates the n-place r.e. relations. (So $W^2(e, m_1)$ is just our previous relation W(e, y). We will usually omit the superscript when the context makes it clear which one is intended.) The easiest way to prove this is by defining $W^{n+1}(e,m_1,...,m_n)$ as $W^2(e, [m_1, \ldots, m_n])$. It is clear that this enumerates the n-place r.e. relations. We now have:

Theorem: For all n, there is an n+1-place partial recursive function Φ^{n+1} which enumerates the n-place partial recursive functions, i.e. for each n-place partial recursive function ϕ there is a number e such that $\Phi^{n+1}(e, x_1, ..., x_n) = \phi(x_1, ..., x_n)$ for all $x_1, ..., x_n$ for which ϕ is defined, and is undefined on e, $x_1, ..., x_n$ when ϕ is undefined on $x_1, ..., x_n$.
Proof: We only prove the theorem in the case n = 1. (The general case can be proved either by imitation of this case, or via the pairing function.) Consider the relation W^3 which enumerates the 2-place r.e. relations. Being an r.e. relation itself, it is uniformized by some 2-place partial recursive function Φ. Now let ϕ be any 1-place partial recursive function, and let R be ϕ's graph. R is W^3_e for some e, i.e. for some e, $W^3(e, x, y)$ holds iff $\phi(x) = y$. Since Φ uniformizes W^3, $\Phi(e, x) = y$ iff $\phi(x) = y$; moreover, if $\phi(x)$ is undefined, then $W^3(e, x, y)$ does not hold for any y, and so $\Phi(e, x)$ is undefined.

The number e is called an *index* of the function ϕ. Kleene's notation for $\Phi(e, x)$ is $\{e\}(x)$; so $\{e\}$ denotes the partial recursive function with index e.

Just as no recursive relation enumerates the recursive sets, no total recursive function enumerates the total recursive functions. To see this, suppose Ψ did enumerate the total recursive functions. Let ϕ be the total recursive function $\Psi(x, x) + 1$; then there is an e such that $\phi(x) = \Psi(e, x)$ for all e. So in particular, $\phi(e) = \Psi(e, e)$. But $\phi(e) = \Psi(e, e) + 1$, so we have $\Psi(e, e) = \Psi(e, e) + 1$, which is impossible.

Why doesn't this show that an enumeration of the *partial* recursive functions is impossible? Let $\phi(x) = \Phi(x, x) + 1$. ϕ is a partial recursive function, so it has an index e; so $\Phi(e, x) = \Phi(x, x) + 1$ for all x, and in particular $\Phi(e, e) = \Phi(e, e) + 1$. But this is not a contradiction, for it only shows that $\Phi(e, e)$ is undefined. It is this fact about partial recursive functions, that they can be undefined, that allows there to be an enumeration theorem for partial recursive functions, and indeed this was the point of studying partial recursive functions (as opposed to just total recursive functions) in the first place.

(In some presentations, a new "number" u is introduced to represent an undefined value, i.e. we declare that $\phi(x) = u$ when $\phi(x)$ is undefined. Then every function we care to deal with has a value, of sorts, for every argument. The argument of the last paragraph shows that we must have $u = u + 1$: we showed that $\Phi(e, e) = \Phi(e, e) + 1$, from which it follows that $\Phi(e, e) = u$ ($n \neq n + 1$ for all n other than u), and therefore that $u = u + 1$. A similar argument shows that $\phi(u) = u$ for all partial recursive functions ϕ, i.e. u is a fixed point of all partial recursive functions. We will not use u in this course, however.)

This also provides an example of a partial recursive function which is not totally extendible, i.e. which is not extended by any total recursive function. Specifically, Φ is such a function. For suppose Ψ is a total recursive function extending Φ, and let $\psi(x) = \Psi(x, x) + 1$. ψ is a total recursive function with some index e. Then $\Phi(e, x) = \psi(x) = \Psi(x, x) + 1$ for all x on which y is defined, which is all x, since ψ is total. So $\Phi(e, e) = \psi(e) = \Psi(e, e) + 1$. Since Ψ extends Φ, and therefore agrees on Φ whenever Φ is defined, $\Psi(e, e) = \psi(e) = \Psi(e, e) + 1$, which is impossible.

Given a 2-place partial recursive function which is not totally extendible, we can find a 1-place partial recursive function which is not totally extendible via the pairing function: if $\phi(x, y)$ is such a 2-place partial function, let ψ be a partial recursive 1-place function such that $\psi([x, y]) = \phi(x, y)$ whenever $\phi(x, y)$ is defined. If ψ were totally extendible to some function ψ', then we could let $\phi'(x, y) = \psi'([x, y])$, and ϕ' would be a total recursive function extending ϕ. Alternatively, we could simply observe that the function $\phi(x) = \Phi(x, x) + 1$ is not totally extendible, using the argument of the last paragraph.

Reduction and Separation.

Let C be any class of sets. C is said to have the *separation property* if for any disjoint S_1 and $S_2 \in C$, there is an $S \in C$ such that $-S \in C$, $S_1 \subseteq S$, and $S_2 \subseteq -S$. S is said to separate S_1 and S_2.

Separation fails for the r.e. sets. A pair of r.e. sets which is not separated by any recursive set is called a *recursively inseparable pair*. The proof that there are recursively inseparable pairs of r.e. sets is due to Kleene, using Φ. Let $S_1 = \{m: \Phi(m, m) = 0\}$, and let $S_2 = \{m: \Phi(m, m)$ is defined and $> 0\}$. Clearly, S_1 and S_2 are disjoint r.e. sets. If separation held for the r.e. sets, then there would be a recursive S with $S_1 \subseteq S$ and $S_2 \subseteq -S$. But we can easily derive a contradiction by considering the characteristic function of S, ψ. If $\Phi(m, m) = 0$ then $\psi(m)=1$; if $\Phi(m, m) > 0$ then $\psi(m)=0$. Since S is recursive, ψ is recursive, and so partial recursive, and therefore, for all m, $\psi(m)=\Phi(e, m)$ for some e. Then if $\Phi(e, e) = 0$, $\psi(e)=1=\Phi(e, e)$; and if $\Phi(e, e)>0$, $\psi(e)=0=\Phi(e, e)$. Contradiction.

Let C be any class of sets. C is said to have the *reduction property* if for any $S_1, S_2 \in C$, there are disjoint sets S_1' and $S_2' \in C$ such that $S_1' \subseteq S_1$, $S_2' \subseteq S_2$, $S_1' \cup S_2' = S_1 \cup S_2$.

The class of r.e. sets has the reduction property. This can be seen by an application of

the uniformization theorem. Specifically, let S_1 and S_2 be r.e. sets, and let R be the many-valued function that takes the value 1 on S_1 and 0 on S_2 (and therefore takes both values on $S_1 \cap S_2$); apply uniformization to shrink R to a single-valued function R' with the same domain. Then we let $S_1'=\{m: R'(m,1)\}$ and $S_2'=\{m: R'(m,0)\}$. S_1' and S_2' obviously have the desired properties.

If a class C has the reduction property, the corresponding class $C_1=\{-X: X \in C\}$ has the separation property. For suppose we have two disjoint sets S_1 and S_2 in C_1. Then $-S_1 \cup -S_2=\mathbf{N}$. Applying to $-S_1$ and $-S_2$ the reduction property of C, we know that there are S_1' and S_2' in C such that $-(S_1') \subseteq -S_1$, $-(S_2') \subseteq -S_2$, and such that $-(S_1') \cup -(S_2') = -S_1 \cup -S_2 =\mathbf{N}$, and $-(S_1')$ and $-(S_2')$ are disjoint. So $S_1' \cup S_2'=\mathbf{N}$ and S_1' and S_2' are disjoint and therefore $S_2'=-(S_1')$; and moreover $S_1 \subseteq S_1'$ and $S_2 \subseteq S_2'$. So S_1' separates S_1 and S_2.

This fact can be used to prove that separation holds for the Π_1 sets (which are the co-r.e. sets). On the other hand, they cannot have the reduction property, because that would imply that the Σ_1 sets (i.e., the r.e. sets) have the separation property, and they don't. We may also note that the Π_1 relations do not have, unlike the Σ_1 relations, the uniformization propety: if they did, we could imitate the proof we gave for the r.e. sets to prove that the Π_1 sets have the reduction property.

Functional Representability.

We have said what it is for a relation to be weakly or strongly representable in a theory; we now define a notion of representability in a theory for partial functions.

Definition: A partial function ϕ is *represented* in a theory Γ by a formula $A(x_1, ..., x_n, y)$ iff whenever $\phi(a_1, ..., a_n) = b$, Γ fi $A(\mathbf{0}^{(a_1)}, ..., \mathbf{0}^{(a_n)}, \mathbf{0}^{(b)}) \wedge (y)(A(\mathbf{0}^{(a_1)}, ..., \mathbf{0}^{(a_n)}, y) \supset y = \mathbf{0}^{(b)})$. ϕ is *representable* in Γ iff some formula represents it in Γ.

Notice that in our definition we do not say what happens when $\phi(a_1, ..., a_n)$ is undefined. In particular, we do not require that $A(\mathbf{0}^{(a_1)}, ..., \mathbf{0}^{(a_n)}, \mathbf{0}^{(b)})$ not be a theorem. So whenever a formula A represents a function ϕ in Γ, A also represents each subfunction of ϕ; in particular, every formula represents the completely undefined function in every theory. Also, if A represents ϕ in Γ and ϕ has an infinite domain, then ϕ has $2^{\aleph_0}$ subfunctions, and so A represents $2^{\aleph_0}$ functions in Γ. It follows that not every function that A represents is partial recursive, since there are only $\aleph_0$ partial recursive functions. Notice also that in an inconsistent theory, every formula represents every function.

Notice that representability is different from definability: a formula can represent a function without defining it, and vice versa. Notice also that if Δ extends Γ, then every function representable in Γ is representable in Δ, since if $A(\mathbf{0}^{(a_1)}, ..., \mathbf{0}^{(a_n)}, \mathbf{0}^{(b)}) \wedge (y)(A(\mathbf{0}^{(a_1)}, ..., \mathbf{0}^{(a_n)}, y) \supset y = \mathbf{0}^{(b)})$ is a theorem of Γ, then it is also a theorem of Δ.

We now set out to prove the theorem that every partial recursive function is representable in Q. To this effect we prove the following two lemmas.

Lemma 1: If two partial 1-place functions ϕ_1 and ϕ_2 are both representable in a theory Γ so is their composition $\phi_2(\phi_1(x))$.
Proof: Let $R_1(x,y)$ and $R_2(y,z)$ represent ϕ_1 and ϕ_2 respectively. Then it is not difficult to verify that the formula $(\exists y)(R_1(x,y) \wedge R_2(y,z))$ represents their composition.

Lemma 2: Any partial function whose graph is definable in Lim is representable in Q.
Proof: Let ϕ be any 1-place partial function whose graph is defined by the formula A(x, y) of Lim. Let B(x, y) be the formula $A(x, y) \wedge (z < y) \sim A(x, z)$. We claim that B represents ϕ in Q. Suppose $\phi(a) = b$. We have to verify two things: namely, that $B(\mathbf{0}^{(a)}, \mathbf{0}^{(b)})$ is a theorem of Q, and that $(y)(B(\mathbf{0}^{(a)}, y) \supset y = \mathbf{0}^{(b)})$ is a theorem of Q. Clearly, $A(\mathbf{0}^{(a)}, \mathbf{0}^{(b)})$ is a theorem of Q, since $A(\mathbf{0}^{(a)}, \mathbf{0}^{(b)})$ is a true RE sentence. To show that Q fi $B(\mathbf{0}^{(a)}, \mathbf{0}^{(b)})$, we must also show that Q fi $(z < \mathbf{0}^{(b)}) \sim A(\mathbf{0}^{(a)}, z)$. But again, this is a true sentence of RE, and is therefore a theorem of Q.

Next, we must show that Q fi $(y)(B(\mathbf{0}^{(a)}, y) \supset y = \mathbf{0}^{(b)})$. Here we use Fact 2 about Q from Lecture IX, i.e. for all n, Q fi $(x_1)(x_1 = \mathbf{0}^{(n)} \vee x_1 < \mathbf{0}^{(n)} \vee \mathbf{0}^{(n)} < x_1)$. Using this fact, we establish $(y)(B(\mathbf{0}^{(a)}, y) \supset y = \mathbf{0}^{(b)})$ by reasoning within Q. Suppose $B(\mathbf{0}^{(a)}, y)$, i.e. $A(\mathbf{0}^{(a)}, y)$ and $(z < y) \sim A(\mathbf{0}^{(a)}, z)$. We want to show that $y = \mathbf{0}^{(b)}$. By Fact 2, there are three possibilities: $y = \mathbf{0}^{(b)}$, or $y < \mathbf{0}^{(b)}$, or $\mathbf{0}^{(b)} < y$. If $\mathbf{0}^{(b)} < y$, then $\sim A(\mathbf{0}^{(a)}, \mathbf{0}^{(b)})$, since $\sim A(\mathbf{0}^{(a)}, z)$ for all $z < \mathbf{0}^{(b)}$. So suppose $y < \mathbf{0}^{(b)}$. We know that $B(\mathbf{0}^{(a)}, \mathbf{0}^{(b)})$, and so $(z < \mathbf{0}^{(b)}) \sim A(\mathbf{0}^{(a)}, z)$. So in particular $\sim A(\mathbf{0}^{(a)}, y)$, contradiction. So neither $\mathbf{0}^{(b)} < y$ nor $y < \mathbf{0}^{(b)}$ holds, and so $y = \mathbf{0}^{(b)}$. This reasoning can be carried out formally in Q, as can easily be verified, and so Q fi $(y)B(\mathbf{0}^{(a)}, y) \supset y = \mathbf{0}^{(b)})$. This completes the proof that B represents ϕ in Q.

We can now prove the desired

Theorem: Every partial recursive function is representable in Q (and therefore in any axiom system extending Q).
Proof: For simplicity we only prove the theorem for 1-place functions. Let ϕ be a partial recursive function. Then by the normal form theorem for partial recursive functions, $\phi(x) = U(\mu y R(x, y))$ for some relation R definable in Lim and some U whose graph is definable in Lim. (In fact, of course, we can take U to be K_1.) Then the functions U and $\mu y R(x, y)$ both have graphs definable in Lim, so by Lemma 2, both are representable in Q; by Lemma 1, their composition, which is ϕ, is representable in Q.

Corollary: Every recursive set is strongly representable in every consistent extension of Q.
Proof: Let Γ be some consistent extension of Q, and let S be any recursive set. Let ϕ be S's

characteristic function, and let R(x, y) be some formula which represents ϕ in Q, and therefore in Γ. (Such an R exists by the preceding theorem.) Let B(x) be the formula R(x, **0**'). If $n \in S$, then $\phi(n) = 1$, so $\Gamma \vdash R(\mathbf{0}^{(n)}, \mathbf{0}')$. If, on the other hand, $n \notin S$, then $\phi(n) = 0$, so $\Gamma \vdash (y)(R(\mathbf{0}^{(n)}, y) \supset y = \mathbf{0})$, so $\Gamma \vdash \sim R(\mathbf{0}^{(n)}, \mathbf{0}')$ (since $\mathbf{0} \neq \mathbf{0}'$ is a theorem of Q). So R(x, **0**') strongly represents S in Γ.

This corollary extends our previous result: before, we only knew that every set definable in Lim is strongly representable in Q (and therefore in any consistent extension of Q).

We can use our results to prove Rosser's form of Gödel's theorem:

Rosser's Theorem: If Γ is a consistent r.e. extension of Q, then Γ is incomplete.
Proof: We can give two different proofs using results we have proved. The first, closer in spirit to Rosser's is this. Consider the function $\Phi(x, x)$. We know that the sets $S_1 = \{m: \Phi(m, m) = 0\}$, and $S_2 = \{m: \Phi(m, m)$ is defined and $> 0\}$ are recursively inseparable. Let A(x, y) be a formula that represents the function $\Phi(x, x)$ in Γ. So we have that if $\Phi(m, m) = 0$ then $\Gamma \vdash A(\mathbf{0}^{(m)}, \mathbf{0}) \wedge (y)(A(\mathbf{0}^{(m)}, y) \supset y = \mathbf{0})$; and if $\Phi(m, m)$ is defined and $=n> 0$ then $\Gamma \vdash A(\mathbf{0}^{(m)}, \mathbf{0}^{(n)}) \wedge (y)(A(\mathbf{0}^{(m)}, y) \supset y = \mathbf{0}^{(n)})$. By the second conjunct in the last formula, if $\Phi(m, m)$ is defined and > 0, $\Gamma \vdash \sim A(\mathbf{0}^{(m)}, \mathbf{0})$. Since Γ is consistent, it is not the case that $\Gamma \vdash A(\mathbf{0}^{(m)}, \mathbf{0})$ and $\Gamma \vdash \sim A(\mathbf{0}^{(m)}, \mathbf{0})$. Let $R_1 = \{m: \Gamma \vdash A(\mathbf{0}^{(m)}, \mathbf{0})\}$ and $R_2 = \{m: \Gamma \vdash \sim A(\mathbf{0}^{(m)}, \mathbf{0})\}$. These are disjoint (since Γ is consistent) and, if Γ were complete, they would be the complement of each other (and so exhaust **N**). They are r.e., for the usual reasons. So if they were the complement of each other, they would be recursive, and then R_1 would be a recursive set that would separate S_1 and S_2, and we prove that no set does that. So we can conclude that Γ is not complete.

A second way of proving the theorem is the following. Suppose, for a contradiction, that Γ is a consistent r.e. extension of Q that is complete. Since Γ is complete, the set of theorems of Γ is recursive. Consider the relation $R = \{\langle e, m\rangle:$ e is a Gödel number of a formula $A(x_1)$, and $A(\mathbf{0}^{(m)})$ is a theorem of $\Gamma\}$. Γ being recursive, R is a recursive relation. Moreover, R enumerates the recursive sets, in the sense that each recursive set is R_e for some e. To see this, let S be a recursive set, and let $A(x_1)$ be a formula that strongly represents it in Γ; then if e is a Gödel number of $A(x_1)$, $S = \{m: \Gamma \vdash A(\mathbf{0}^{(m)})\} = R_e$. So R is a recursive enumeration of the recursive sets. But as we saw in Lecture IX, this is impossible. Therefore, no such Γ can exist, and so any r.e. consistent Γ extending Q is incomplete.

Exercises

1. (a) Prove that an infinite set S of natural numbers is r.e. iff it is the range of a 1-1 total recursive function.

(b) Prove that an infinite set S of natural numbers is recursive iff it is the range of a 1-1 monotone increasing total recursive function.

(c) Prove that every infinite r.e. set has an infinite recursive subset.

2. *Reduction property within Q.* (a) If S_1 and S_2 are two r.e. sets, prove that there are two r.e. sets S_1' and S_2' such that $S_1' \subseteq S_1$, $S_2' \subseteq S_2$, and $S_1' \cup S_2' = S_1 \cup S_2$, such that S_1' is weakly represented by a formula A(x) and S_2' by a formula B(x) and $(x)\sim(A(x) \wedge B(x))$ is a theorem of Q.

(b) Hence, if two r.e. sets S_1 and S_2 are in fact disjoint, they can be weakly represented by two formulae A(x) and B(x) such that $(x)\sim(A(x) \wedge B(x))$ is a theorem of Q.

3. (a) Show that the following instance of the naive comprehension scheme is inconsistent: $(\exists y)(x)(x \in y \equiv \sim(\exists w)(x \in w \wedge w \in x))$.

(b) Analogous to the construction of K using Russell's paradox, use the result in (a) to obtain a corresponding r.e. set which is not recursive.

(c) Given an r.e. axiom system Γ extending Q, define a number n to be *Gödel-unreciprocated* if m is a Gödel number of a formula $A(x_1)$ and there is no n such that n is the Gödel number of a formula $B(x_1)$ with $A(\mathbf{0}^{(n)})$ and $B(\mathbf{0}^{(m)})$ both provable in Γ. (Otherwise, m is *Gödel-reciprocated*.) Show, analogously to the treatment of 'Gödel-heterological', that the sentence '"Gödel-unreciprocated" is Gödel-unreciprocated' has the properties of the Gödel statement, i.e. it is a Π_1 statement that is true but unprovable if Γ is consistent and not disprovable if Γ is ω-consistent. (Note: this is the Gödelian analog of the paradox of part (a), and is meant to illustrate the theme that set-theoretic paradoxes can be turned into proofs of Gödel's theorem.)

4. (a) Show that there is a recursive function $\psi(m,n)$ such that $\psi(m,n)$ is a code of the n-term sequence all of whose terms are m.

(b) The Upwards Generated Sets Theorem says that if G is a set generated by a recursive basis set and some recursive generating relations such that for each generating relation R, the conclusion of R is greater than or equal to all of the premises, then G is recursive. Prove this theorem. [Hint: prove that every element m of G occurs in a proof sequence for G such that all elements preceding m in the sequence are strictly less than m. Then use (a).]

(c) Use (b) to prove that the set of Gödel numbers of formulae of the narrow first order language of arithmetic is recursive.

(d) Extend this result to any first order language (in the narrow formulation) with finitely many function letters and primitive predicate letters and constants.

5. **Gödel's Theorem via a language with self-reference and extra constants.**

The following is a method of proving the Gödel theorem that directly captures the idea that the Gödel sentence says "I am not provable". It goes by adding additional constants to the narrow first order language of arithmetic; as we have formulated that language, it has only a single constant a_1 standing for zero. We now add all of the others (a_2, a_3, ...) which will denote various numbers. Call the expanded language L*. If we have a set Γ of axioms in L, once we know what we want the extra constants to denote, Γ* will be obtained by adding to Γ all axioms of the form $a_{n+1} = \mathbf{0}^{(m_n)}$, where m_n is the number we want a_{n+1} to denote. (We may not care what certain of the a_{n+1}'s denote, in which case we do not add any axiom involving a_{n+1} to Γ*.) Notice that the language L* and the axiom system Γ* are a mere variant of L and Γ, since all we've done is to add special names for various particular numbers, and nothing can be expressed or proved that couldn't be expressed or proved already.

(a) Use the last remark to prove that if Γ is expanded to an axiom set Γ* with at most one axiom of the given form for each constant, then any proof in Γ* can be transformed into a proof in Γ by replacing each constant by the corresponding numeral and using the axiom (x)(x=x).

(b) Hence, show that every theorem of Γ* becomes a theorem of Γ when constants in the theorem, if any, are replaced by the corresponding numerals. Also show that Γ* is consistent iff Γ is, and that the same holds for ω-consistency.

Now let us make a particular choice of m_n, as follows: if n is a Gödel number of a formula A of L in which x_1 does not occur bound (but in which variables other than x_1 may occur free), let m_n be the least Gödel number of the formula $A(a_{n+1})$ obtained from A by naive substitution of x_1 by a_{n+1} throughout, and include the sentence $a_{n+1} = \mathbf{0}^{(m_n)}$ in Γ*. (Notice that intuitively, if A says something $A(x_1)$, then under our interpretation of the meaning of a_{n+1}, $A(a_{n+1})$ says "I have property $A(x_1)$". Observe that what numbers are the Gödel numbers of a given formula is independent of which interpretation we give to the extra symbols.)

(c) Show that if Γ r.e., then so is Γ* and therefore so is the set of theorems of Γ*.

(d) Show that there is therefore a Π_1 formula $(x_2)B(x_1, x_2)$, where $B(x_1, x_2)$ is a formula of Lim, and which is satisfied by precisely those numbers that are not Gödel numbers of theorems of Γ. We may assume that in this formula x_1 does not occur bound. Let n be the smallest Gödel number of this formula. Assume that Γ extends Q. Prove that if Γ is consistent, then $(x_2)B(a_{n+1}, x_2)$ is true but not provable from Γ*, and therefore that $(x_2)B(\mathbf{0}^{(m_n)}, x_2)$ is also true but unprovable from Γ.

(e) Show that if Γ* is ω-consistent, then $\sim(x_2)B(a_{n+1}, x_2)$ is not provable from Γ* and that if Γ is ω-consistent, $\sim(x_2)B(\mathbf{0}^{(m_n)}, x_2)$ is not provable from Γ.

Remark: (d) and (e) prove Gödel's theorem both for Γ* and for the original system Γ. The point of this exercise is to show that the use of "self-reference" in Gödelian arguments, usually obtained by a rather indirect method, can be obtained by directly constructing a

formula of the form "a is not provable", where a is a *name* of the formula itself. Gödel himself may have been under a certain amount of misapprehension about this point. See his *Collected Works*, vol. I (Oxford, 1986), p. 151, n. 15: "Contrary to appearances, such a proposition involves no faulty circularity, for initially it [only] asserts that a certain well-defined formula (namely, the one obtained from the *q*th formula in the lexicographic order by a certain substitution) is unprovable. *Only subsequently (and so to speak by chance) does it turn out that this formula is precisely the one by which the proposition itself was expressed*." (Emphasis added) In the present construction, this is not at all "by chance". On the contrary, we have deliberately set up the denotation so that the formula refers to itself. Nonetheless, there is no "faulty circularity", because the constant a denotes the (smallest) Gödel number of a definite string of symbols, and this number is determined independently of any interpretation of a. We can then assign that number to a as denotation. There are other ways of accomplishing this type of 'direct' self-reference.

(f) In this version of the construction, why are infinitely many constants introduced? Only one constant is used in the undecidable formula.

6. Let ϕ be a uniformization of the relation defined by $W(x,y) \wedge y>2x$. Let S be the range of ϕ.

(a) Prove that S is r.e.

(b) Prove that S intersects every infinite r.e. set.

(c) Prove that the complement of S is infinite.

(d) Prove that S is neither recursive nor 1-1 complete, citing a previous exercise.

Remark: This is the promised example of an r.e. set that is neither recursive nor 1-1 complete. As I have said, such sets rarely arise in practice unless we are trying to construct them. Later it will be proved that K *is* 1-1 complete.

Lecture XIII

Languages with a Recursively Enumerable but Nonrecursive Set of Formulae.

In dealing with formal languages, it is common to require that the set of formulae of the language be recursive. In practice, however, one hardly ever needs to use more than the fact that the set of formulae is r.e. In practice, one also hardly ever encounters languages with a recursively enumerable but nonrecursive set of formulae. However, there seems to be nothing in principle wrong with such languages, especially if one thinks, as e.g. Chomsky does, that to give a grammar for a language is to give a set of rules for generating the well-formed formulae, rather than to give a procedure for determining whether a given string of symbols is well-formed or not.

We can easily cook up a language with a non-recursive but r.e. set of formulae. For example, let S be any set which is r.e. but not recursive, and let L be the first-order language which contains no function symbols or constants and whose predicates are $\{P_1^n: n \in S\}$. L will be as required.

While this language is artificial, natural examples sometimes arise as well. In a system of Hilbert and Bernays, for example, there is, in addition to the usual logical symbols, an operator (ιy), such that $(\iota y)A(x_1, ..., x_n, y)$ denotes the unique y such that $A(x_1, ..., x_n, y)$ holds. Hilbert and Bernays thought that this really only makes sense if there is a unique y such that $A(x_1, ..., x_n, y)$ holds, so they stipulated that (ιy) could be introduced only through the rule

$$\frac{(x_1) ...(x_n)(\exists!y)A(x_1, ..., x_n, y)}{(x_1) ...(x_n)A(x_1, ..., x_n, (\iota y)A(x_1, ..., x_n, y))}$$

(where $(\exists!y)A(x1, ..., xn, y)$ means that there is a unique y such that $A(x1, ..., xn, y)$ holds, and is an abbreviation of $(\exists y)(A(x1, ..., xn, y) \wedge (z)(A(x1, ..., xn, z) \pounds z = y)))$. A result of this policy is that the set of well-formed formulae of the language will in general be nonrecursive, though it will be r.e. Hilbert and Bernays were criticized on this point, though it is not clear why this is a ground for criticism.

In terms of our own formalism, we could stipulate that f_i^n be introduced when $(x_1)...(x_n)(\exists!y)A(x_1, ..., x_n, y)$ is a theorem, where i is a certain Gödel number of A, and add as a theorem $(x_1)...(x_n)A(x_1, ..., x_n, f_i^n(x_1, ..., x_n))$.

The S^m_n Theorem.

If R is a 2-place r.e. relation, then intuitively R_k should be r.e. as well; but furthermore, given k, we ought to be able to effectively find an index for R_k. This is indeed the case, and is a special case of the S^m_n theorem. More generally, let R be an m+n-place r.e. relation. In the case we have just considered, R_k is obtained from R by fixing k as a parameter; the general form of the S^m_n theorem (put informally) says that given m numbers k_1, ..., k_m, we can effectively find an index for the relation obtained from R by fixing k_1, ..., k_m as parameters. In our own formalism, the S^m_n theorem is an easy consequence of the definability in RE of substitution. We now state the S^1_1 theorem, i.e. the special case of the S^m_n theorem in which m = n = 1.

Theorem: For any 2-place r.e. relation R, there is a 1-1 recursive function ψ such that, for all k, $W_{\psi(k)} = R_k$.
Proof: Let e be an index for R. e is a Gödel number of some formula of RE $A(x_2,x_1)$ that defines R. An index of R_k, i.e. a Gödel number of a formula of RE that defines R_k, can be obtained from e via substitution. More specifically, we define the graph of ψ in RE by the formula $PS(k,y) =_{df.} (\exists p \leq y)(Num(p,k) \wedge (w<p) \sim Num(w,k) \wedge NSubst(\mathbf{0}^{(e)},y,[\mathbf{0}^{(1)},\mathbf{0}^{(2)}],p) \wedge (w<y)(\sim NSubst(\mathbf{0}^{(e)},w,[\mathbf{0}^{(1)},\mathbf{0}^{(2)}],p)))$ (the use of negation is legitimate, since the formulae it affects are equivalent to formulae of Lim). Informally, ψ assigns to k the least Gödel number of the formula obtained by substituting the least Gödel number of the numeral of k for x_2 in the formula with Gödel number e. The function thus defined is clearly 1-1, since the results of substituting different numerals for the same variable in the same formula must have different Gödel numbers.

The general form of the S^m_n theorem can be stated and proved similarly: for any m+n-place r.e. relation R, there is a 1-1 recursive function ψ such that, for all $k_1,...,k_m$, $W^{n+1}_{\psi(k1,...,km)} = R_{k1,...,km}$ (where $R_{k1,...,km}$ is $\{<y_1, ..., y_n>: R(k_1, ..., k_m, y_1, ..., y_n)\}$). As we see, the name "$S^m_n$ theorem" derives from the convention of taking m as the number of parameters and n as the number of other variables; 'S' probably stood for 'substitution' in the original conception of Kleene, to whom the theorem is due.

As a consequence of the above theorem, we have the following

Theorem: For all m and n, there is a one to one m+1-place recursive function ψ such that for all m+n-place r.e. relations R, if e is an index of R and k_1, ..., k_n are numbers, then $\psi(e, k_1, ..., k_m)$ is an index of $\{<y_1, ..., y_n>: R(k_1, ..., k_m, y_1, ..., y_n)\}$.
Proof: Apply the previous form of the S^m_n theorem to the relation W^{m+n+1}. That is, let ψ be a function such that $\psi(e, k_1, ..., k_m)$ is an index of $\{<y_1, ..., y_n>: W(e, k_1, ..., k_m, y_1, ..., y_n)\} = \{<y_1, ..., y_n>: R(k_1, ..., k_m, y_1, ..., y_n)\}$.

The second form of the S_n^m theorem can thus be seen as a special case of the first. The first form also follows directly from the second. A third form of the theorem is the standard form in most presentations of recursion theory, and the form originally proved by Kleene:

Theorem: For all m and n, there is a one to one m+1-place recursive function ψ such that if e is an index of an m+n-place partial recursive function ϕ, then $\psi(e, k_1, ..., k_m)$ is an index of the n-place function $\phi(k_1, ..., k_m, y_1, ..., y_n)$.
Proof: Apply the previous theorem to the relation W^{m+n+2*}, the graph of the m+n+1-place function Φ^{m+n+1}.

Given m,n, a function ψ with the property stated in the third version of the theorem (for m,n) is a function standardly called an S_n^m function.

The Uniform Effective Form of Gödel's Theorem.

We can use the S_n^m theorem to prove the uniform effective form of Gödel's theorem, i.e. that for any consistent r.e. extension Γ of Q, a sentence undecidable in Γ can be obtained (in a uniform way for all Γ) effectively from Γ itself. Specifically, given a formula A defining -K, we can find a recursive function ψ such that for all e, $\psi(e)$ is a number such that the statement that A is true of $\psi(e)$ is true but unprovable from W_e if W_e is a consistent extension of Q, and undecidable in W_e if W_e is also ω-consistent. (We say that a sentence is a theorem of W_e if it is a theorem of the set of sentences whose Gödel numbers are elements of W_e; so if W_e contains numbers other than the Gödel numbers of sentences, we ignore them.)

Recall the proof of Gödel's theorem. Let $\Gamma = W_e$ be any r.e. axiom system, and let A(x) be some Π_1 formula that defines -K. Then let $(-K)^* = \{m: \Gamma \vdash A(\mathbf{0}^{(m)})\}$, the set of numbers provably in -K. Since $(-K)^*$ is r.e., for the familiar reasons, $(-K)^*$ is W_f for some f. Then the proof we are familiar with shows that $A(\mathbf{0}^{(f)})$ is true but unprovable in Γ provided that Γ is a consistent extension of Q, and undecidable if Γ is ω-consistent. Intuitively, f depends effectively on e, so f should be $\psi(e)$ for some recursive function ψ. It is the proof that this is the case that uses the S_n^m theorem.

Uniform Effective Form of Gödel's Theorem: For every Π_1 formula A(x) defining -K, there is a recursive function ψ such that for all e, $A(\mathbf{0}^{(\psi(e))})$ is true but unprovable from W_e, if W_e is a consistent extension of Q, and undecidable if W_e is an ω-consistent extension of Q.
Proof: Let A(x) be a fixed Π_1 formula defining -K, and let R be the relation $\{<e, m>: A(\mathbf{0}^{(m)})$ is a theorem of $W_e\}$. If R is r.e., then by the S_1^1 theorem we can find a recursive ψ

such that $W_{\psi(e)}=R_e = \{m: R(e, m)\} = \{m: A(\mathbf{0}^{(m)})$ is a theorem of $W_e\}$ for all e. So $A(\mathbf{0}^{(\psi(e))})$ must be true but unprovable if W_e is a consistent extension of Q, and undecidable if W_e is an ω-consistent extension of Q. So we only have to prove that R is r.e., but this is clear. Let χ be a recursive function such that $\chi(m)$ is a certain Gödel number of $A(\mathbf{0}^{(m)})$. Note that $A(\mathbf{0}^{(m)})$ is a theorem of W_e iff there is a proof sequence from the sentences of W_e on which a Gödel number of $A(\mathbf{0}^{(m)})$ occurs. A proof sequence from W_e is simply a finite sequence of numbers, each of which either codes a sentence in W_e or a logical axiom, or follows from earlier terms in the sequence by a logical rule of inference. So it is clear that we can find an RE formula PS(s, e) which says that s is a proof sequence from W_e; we can then define Th(e, x) as $(\exists s)(\exists n \leq s)(PS(s, e) \wedge [n, x] \in s)$. Th(e, x) says that x is a Gödel number of a formula provable from W_e. Using the function χ above, the relation R is defined by the RE formula $Th(e, \chi(m))$.

We say that a nonrecursive r.e. set S *satisfies the uniform effective form of Gödel's theorem* just in case for some Π_1 formula A(x) defining -S, there is a recursive function ψ such that for all e, $A(\mathbf{0}^{(\psi(e))})$ is true but unprovable from W_e, if W_e is a consistent extension of Q, and undecidable if W_e is an ω-consistent extension of Q. The theorem just proved shows that the set K satisfies the uniform effective form of Gödel's theorem . However, not every nonrecursive r.e. set satisfies it. In particular, Post's simple set (defined in the exercises) does not satisfy the uniform effective form of Gödel's theorem.

The Second Incompleteness Theorem.

We shall now use the uniform effective form of Gödel's theorem to prove a version of Gödel's second incompleteness theorem, the theorem that says that a sufficiently strong r.e. axiom system cannot prove its own consistency. Our proof is based on a proof by Feferman, although it differs from that proof in an important respect. Before giving the proof, we will say a little bit about the philosophical background of Gödel's second incompleteness theorem.

In the early decades of the twentieth century, many mathematicians believed, especially because of the paradoxes, that mathematics might be in serious foundational trouble. Several leading mathematicians had then a strong interest in logic and foundations. Many of these mathematicians thought that the reason behind the trouble is that one cannot reason validly about the infinite, at least in a "natural" way, e.g., they thought that one cannot reason validly about the totality of natural numbers, as opposed to something you can reason about by reasoning about larger and larger initial segments.

Two of those leading mathematicians with strong foundational interests were Brouwer and Hilbert. Brouwer thought from the beginning that mathematics had to be radically revised, and he proposed a doctrine of what mathematical reasonings are acceptable, called

'intuitionism'. In intuitionism, infinitary constructions were not acceptable, and principles about infinite collections licensed by classical logic, like the principle that, for a given property, either all numbers have it or there is a number that is a counterexample; thus, a proof that not all numbers have a certain property does not guarantee, for the intuitionist, that there is a number without that property (this can only be shown by constructing such a number).

Some mathematicians adopted the point of view on foundations common today, i.e., the point of view that there was no problem of legitimacy with mathematics as it had been done, including set theory; in the case of the logicists, at least a certain modified logical form of set theory was legitimate. An entirely different approach to the foundational crisis was taken by Hilbert. He thought that the intuitionists were right in their worries whether mathematics as it was being done was legitimate. He further thought that the set of methods of mathematical reasoning guaranteed to be legitimate was even more restrictive than the set of methods allowed by the intuitionists. On the other hand, Hilbert did not want to change mathematics. He had the following idea. One should develop mathematics by means of formal systems, as had been done by people working in logic and foundations, and view mathematical theorems as finite strings of symbols without meaning, which could be generated in mechanical ways in the formal systems. But one should prove, by the restrictive methods allowed, that the formal systems of mathematics were consistent.

What would be the value of such a proof of consistency? Normally, the reason we don't want a formal system to be inconsistent is that not all of the theorems of an inconsistent system can be true. Since Hilbert thought that not all theorems of mathematics could be true, this was not his reason for demanding a proof of consistency. Another reason is to show that the system is not uninteresting, for an inconsistent system is uninteresting in the sense that it proves every sentence. But there were other reasons as well. We have proved for our own formalisms that if we have a Π_1 statement (x)L(x), where L(x) is a limited formula, first, we can decide, for any instance $L(\mathbf{0}^{(n)})$ of L(x), whether $L(\mathbf{0}^{(n)})$ is true or not. But second, and more important, that if the system is consistent, then if (x)L(x) is provable then all the instances of L(x) are true; for if some instance was false, it could be shown to be so by finite methods (limited statements, whose quantifiers involve only initial segments of the natural numbers, are the kind of statements taken to be legitimate by Hilbert), and then ~(x)L(x) would be provable, rendering the system inconsistent if (x)L(x) is provable too. In this way, a proof of consistency would provide a legitimation for theorems of the form (x)L(x).

What is known as Hilbert's Program was not merely the idea that proving consistency would be a good thing. The Program suggested by Hilbert actually included a particular and very plausible suggestion of how a proof might be attempted. At the time, it looked as if this suggestion (which we cannot explain here) really ought to work. That's why Gödel's second incompleteness theorem came as a shock, for it showed that consistency for a system could not be proved assuming that Hilbert's restricted finite methods were a subset of the methods

incorporated into the system itself. We can already see from Gödel's first incompleteness theorem that Hilbert's aim was unattainable. For if consistency was provable, then the statement that every Π_1 provable statement is true would be provable. But if this was provable, the Gödel sentence G, which is Π_1, would be such that 'G is provable $\supset$ G' would be a theorem; but G says of itself that it is not provable, so '~G $\supset$ G' would be a theorem, and so by logic G would be a theorem. And this would imply, by the first incompleteness theorem, that the system was not consistent after all.

Let us now give our proof of Gödel's second incompleteness theorem. First, let us see how to write out the first incompleteness theorem in the language of arithmetic. Pick a Π_1 formula A(x) which defines -K, and fix a recursive function ψ as in the uniform effective form of Gödel's theorem proved above. Then

For all e, if W_e is consistent and W_e extends Q, then $A(\mathbf{0}^{(\psi(e))})$ is true but unprovable,

from which it follows that

(†) For all e, if W_e is consistent and W_e extends Q, then $A(\mathbf{0}^{(\psi(e))})$ is true.

(We leave out the second part on the hypothesis of ω-consistency.) We shall write out (†) in the language of arithmetic. We have in effect already seen how to write out the statement that W_e is consistent. We have an RE formula Th(e, x) which says that x is a theorem of W_e; W_e is consistent just in case $\mathbf{0} \neq \mathbf{0}$ is not a theorem of W_e, so W_e is consistent iff e satisfies ~Th(e, $\mathbf{0}^{(n)}$), where n is a Gödel number of $\mathbf{0} \neq \mathbf{0}$; let us write Con(e) for ~Th(e, $\mathbf{0}^{(n)}$). (Alternatively, we could let Con(e) be the sentence $(\exists x)$~Th(e, x), since W_e is consistent iff at least one sentence is not provable from W_e; or we could let Con(e) be the statement that no sentence and its negation are both provable from W_e.) And we can easily write "W_e extends Q" within the system: Q has finitely many axioms A_1, ..., A_k, so let n_1, ..., n_k be their Gödel numbers; W_e extends Q just in case e satisfies Th(e, $\mathbf{0}^{(n_1)}$) $\wedge$ ... $\wedge$ Th(e, $\mathbf{0}^{(n_k)}$). Let us write "e ext. Q" for this formula. Finally, let PS(x, y) be some formula that weakly represents ψ in Q. Now consider the statement

(*) $(e)(\text{Con}(e) \wedge \text{e ext. Q} \supset (\exists y)(\text{PS}(e, y) \wedge A(y)))$

(*) is a partial statement of the first incompleteness theorem, and therefore ought to be provable in reasonably strong systems of number theory. Now consider the theory Q+(*).

Gödel's Second Incompleteness Theorem: If W_e is a consistent extension of Q+(*), then Con($\mathbf{0}^{(e)}$) is not a theorem of W_e, i.e. W_e does not prove its own consistency.
Proof: Suppose W_e extends Q* and Con($\mathbf{0}^{(e)}$) is one of its theorems. Then as (*) is a theorem of W_e, Con($\mathbf{0}^{(e)}$) $\wedge$ $\mathbf{0}^{(e)}$ ext. Q $\supset$ $(\exists y)$(PS($\mathbf{0}^{(e)}$, y) $\wedge$ A(y)) is also a theorem of

W_e; we already know that Con($\mathbf{0}^{(e)}$) is a theorem of W_e, and $\mathbf{0}^{(e)}$ ext. Q is a true sentence of RE and is therefore a theorem of Q and therefore of W_e; so $(\exists y)(PS(\mathbf{0}^{(e)}, y) \wedge A(y))$ is a theorem of W_e. Let $f = \psi(e)$. Since PS represents ψ in Q, W_e fi $PS(\mathbf{0}^{(e)}, \mathbf{0}^{(f)}) \wedge (y)(PS(\mathbf{0}^{(e)}, y) \supset y = \mathbf{0}^{(f)})$; it follows that $A(\mathbf{0}^{(f)})$ is a theorem of W_e. But we already know from the first incompleteness theorem that $A(\mathbf{0}^{(f)})$ is unprovable in W_e if W_e is a consistent extension of Q. Since W_e is an extension of Q, it follows that W_e is inconsistent.

The theorem does not show that there are no statements which might be thought of as expressing the consistency of a system which are not provable in the system, pathological statements of consistency, so to speak. To see this, let Γ be an arbitrary consistent r.e. extension of Q, let Pr'(x) be $Pr(x) \wedge x \neq \mathbf{0}^{(n)}$ (where Pr(x) is any Σ_1 formula defining the set of theorems of Γ, and n is the Gödel number of $\mathbf{0} \neq \mathbf{0}$), and let Con'_Γ be the sentence $\sim Pr'(\mathbf{0}^{(n)})$. Since Γ is consistent, $\mathbf{0} \neq \mathbf{0}$ is not a theorem of Γ, so Pr'(x) defines the set of theorems provable in Γ; if Γ is ω-consistent, then Pr'(x) weakly represents the theorems of Γ in Γ as well. So in a sense, Con'_Γ says that Γ is consistent. However, it is clear that Γ fi $\sim Pr(\mathbf{0}^{(n)})$, i.e. Γ fi Con'_Γ. Also, we know from the exercises that if we have two disjoint r.e. sets, we have weak representations of them which are provably disjoint in Q. If we take the two sets to be on the one hand the set of theorems of Γ, and on the other hand the set of sentences whose negation is a theorem of Γ, we therefore have weak representations of them which are provably disjoint in Q. We might think that the corresponding sentence expresses consistency. One of the aims of Feferman's, and of Jeroslow's, work, was to give conditions for distinguishing these pathological statements from statements for which Gödel's second incompleteness theorem goes through.

An important point about our presentation of Gödel's second incompleteness theorem, where it differs from other presentations, including Feferman's, is that in the hypothesis of the theorem we only require that a single statement (namely, the conjunction of Q and (*)) be a theorem of a system for it to fail to prove its consistency. In other presentations of the theorem, including Gödel's original presentation, the proof that a system does not prove its own consistency requires assuming that a certain sentence, different for each system, is a theorem of the system. Let G be a Gödel sentence for a system Γ which extends Q and let Con_Γ be a sentence in the language of arithmetic that says that Γ is consistent. The first incompleteness theorem states that if Γ is consistent, then G is true but unprovable, so in particular, if Γ is consistent, then G is true. So if Γ is a powerful enough system to prove the first incompleteness theorem, then Γ fi $Con_\Gamma \supset G$. If Γ fi Con_Γ, then Γ fi G; since G is true but unprovable from Γ, it follows that Con_Γ is not a theorem of Γ. This is how the second incompleteness theorem was originally proved, as a corollary of the first incompleteness theorem. Thus, the unprovability of consistency for different Γ's under this presentation is proved under the hypothesis that different sentences are provable in these different Γ's — if Γ and Δ are different systems, then to conclude that neither Γ nor Δ prove

their consistency one must assume that Γ fi $Con_\Gamma \supset G$, and that Δ fi $Con_\Delta \supset D$ (where D is a Gödel sentence for Δ).

On our approach, taking Con_Γ to be $Con(\mathbf{0}^{(e)})$, where e is an index for Γ, we give a single sentence (*) such that any consistent r.e. Γ which extends Q + (*) fails to prove Con_Γ. Without any job of formalization at all, it is shown that any extension of Q+(*) satisfies the second incompleteness theorem. And a system that does not contain Q+(*) is not sufficient for elementary number theory, since it should be clear that the methods used in class can be regarded as methods of elementary number theory.

This much we can say without any formalization at all. And we can presume that some systems are strong enough to contain elementary number theory, and therefore to prove Q+(*). So we know enough at this point to state the main philosophical moral of the second incompleteness theorem -a system in standard formalization strong enough to contain elementary number theory cannot prove its own consistency. Strictly speaking we have stated this only for formalisms whose language is the first-order language of arithmetic, but the technique is easily extended to first-order systems in standard formalization with a richer vocabulary. Some ideas as to how to consider such systems will become clear when we discuss the Tarski-Mostowski-Robinson theorem in a later lecture.

If one wishes to consider a specific system, such as the system we have called 'PA', we can say in advance that it satisfies the conditional statement that if it contains elementary number theory, it cannot prove its own consistency in the sense of $Con(\mathbf{0}^{(e)})$ above. However, we have a task of formalization if we wish to show that the system contains elementary number theory or at any rate Q+(*). Here is one of the misleading features of the name 'Peano arithmetic' that has been used for this system: it gives the impression that by definition the system contains elementary number theory, when in fact it requires a detailed formalization to show that this is so. If, for example, the properties of exponentiation or factorial could not be developed in it, it would not contain elementary number theory after all. We have seen the basic idea of how to do this, but the formalization here is not trivial. Thus it does require a considerable task of formalization to show that (*) can be proved in PA, and hence that the appropriate statement $Con(\mathbf{0}^{(e)})$ is not provable in PA. But it requires no formalization at all to claim that any system in standard formalization containing elementary number theory fails to prove its own consistency.

Lecture XIV

The Self-Reference Lemma.

When Gödel proved the incompleteness theorem, he used the fact that there is a sentence G with Gödel number n which is provably equivalent to the sentence ~Pr($\mathbf{0}^{(n)}$) saying that the formula with Gödel number n is not a theorem. Thus in a sense G says of itself that it is unprovable. We have already pointed out that it is difficult to even remember how G is constructed, and that Gödel's theorem is more naturally motivated by considering the properties of the sentence ~Prov($\mathbf{0}^{(n)}$,$\mathbf{0}^{(n)}$), where n is the Gödel number of ~Prov(x,x). In this sense, Gödel's use of the fact about "self-reference", had the negative effect of making his proof appear somewhat mysterious. On the other hand, it had the positive effect of calling attention to the fact that the argument for the existence of G does not depend in any way on the choice of the predicate ~Pr(x), and establishes a more general claim (which, although not stated by Gödel, can be reasonably attributed to him), usually referred to as 'the self-reference lemma'.

Self-Reference Lemma. Let A(x) be any formula in one free variable in the language of arithmetic (or RE). Then there is a sentence G of the language of arithmetic (of RE) such that G ≡ A($\mathbf{0}^{(n)}$) is a theorem of Q, where n is a Gödel number of G.

(In the case of RE, this could be made precise in two ways: either showing that the translation of G ≡ A($\mathbf{0}^{(n)}$) into the narrow language of arithmetic is provable in Q or showing that the appropriate sentence in the broad language of arithmetic is provable in the appropriate formalization of Q.)

Intuitively, G says of itself that it has the property A(x). To prove a version of the first incompleteness theorem using the lemma, let Γ be any consistent r.e. extension of Q, and let Pr(x) be a formula that defines the set of theorems of Γ in RE. Use the self-reference lemma to obtain a sentence G such that G ≡ ~Pr($\mathbf{0}^{(n)}$) is a theorem of Q and hence of Γ, where n is a Gödel number of G. If G is a theorem of Γ, then Pr($\mathbf{0}^{(n)}$) is a true sentence of RE, and hence is provable in Q and therefore in Γ; since Γ fi G ≡ ~Pr($\mathbf{0}^{(n)}$), Γ fi ~G, so ~G is also a theorem of Γ and Γ is inconsistent. Since we are assuming that Γ is consistent, G is not a theorem of Γ. However, since G says of itself that it is not a theorem of Γ, G is true; or more formally, ~Pr($\mathbf{0}^{(n)}$) is true since G is not a theorem of Γ, G ≡ ~Pr($\mathbf{0}^{(n)}$) is a theorem of Q and is therefore true, so G is true. So G is true but unprovable. The proof of the self-reference lemma reveals that G is a Π_1 sentence; from this it follows that if Γ is ω-consistent, ~G is not provable either.

Notice that we often state the Gödel theorems saying that the sentence obtained is one which is true but unprovable. If the self-reference lemma is stated for the language of

arithmetic, we know that the predicate Tr(x) saying that x is the Gödel number of a true sentence cannot be defined in arithmetic itself. We know also that the opposite situation holds for the language RE. Either way, we have the following corollary which, like the lemma itself, holds for both the language of arithmetic and the language RE:

Corollary: Let A(x) be any formula in one free variable in the language of arithmetic (or RE). Then there is a sentence G of the language of arithmetic (or of RE) with Gödel number n such that $G \equiv A(\mathbf{0}^{(n)})$ and $A(\mathbf{0}^{(n)}) \equiv Tr(\mathbf{0}^{(n)})$ are both true.

There are numerous ways of proving the self-reference lemma. Given our Gödel numbering, G cannot actually be the sentence $A(\mathbf{0}^{(n)})$, since the Gödel number of $A(\mathbf{0}^{(n)})$ must be larger than n. However, it is possible to devise a different Gödel numbering such that for every formula A(x), there is a number n such that $A(\mathbf{0}^{(n)})$ gets Gödel number n. (This method of proving the self-reference lemma was discovered independently by Raymond Smullyan and the author.) If we add extra constants to our language, then we can prove a version of the self-reference lemma for the expanded language. Specifically, let L* be the language obtained from the language of arithmetic by adding the constants a_2, a_3, ... (a_1 is already in L). Interpret the new constants as follows: if n is a Gödel number of a formula $A(x_1)$, then interpret a_{n+1} as the least Gödel number of $A(a_{n+1})$. Then the sentence $A(a_{n+1})$ says of itself that it is A. Note that if m_n is the Gödel number of $A(a_{n+1})$, the sentence $a_{n+1} = \mathbf{0}^{(m_n)}$ is true under this interpretation. If we let Q* be the axiom system obtained from Q by adding as axioms all sentences of the form $a_{n+1} = \mathbf{0}^{(m_n)}$, then Q* fi $A(a_{n+1}) \equiv A(\mathbf{0}^{(m_n)})$ for all n, so we can let G be the sentence $A(a_{n+1})$. So if we chose to work in the language L* rather than L, we could get the self-reference lemma very quickly; moreover, L* does not really have greater expressive power than L, since L* simply assigns new names to some things that already have names in L. Using this version of the self-reference lemma it is also possible to prove Gödel's incompleteness theorem, as we have seen in an exercise.

The proof of the self-reference lemma essentially due to Gödel employs the usual Gödel numbering and constructs the sentence G in a more complicated way. Let A(x) be given. Let ϕ be a recursive function such that if y is the Gödel number of a formula $C(x_1)$, then $\phi(n, y)$ is the Gödel number of $C(\mathbf{0}^{(n)})$. Let B(x, y, z) represent ϕ in Q, and let A'(x, y) be the formula $(\exists z)(B(x, y, z) \wedge A(z))$. If y is the Gödel number of a formula $C(x_1)$, then A'(n, y) holds iff the Gödel number of $C(\mathbf{0}^{(n)})$ satisfies A(x). (We can read A'(x, y) as "y is A of x"; for example, if A(x) is "x is provable", then A'(x, y) is "y is provable of x".) Let m be the Gödel number of $A'(x_1, x_1)$, and let G be the sentence $A'(\mathbf{0}^{(m)}, \mathbf{0}^{(m)})$. ($A(x_1, x_1)$ says that x_1 is A of itself, and G says that "is A of itself" is A of itself.) We shall show that Q fi $G \equiv A(\mathbf{0}^{(n)})$, where n is the Gödel number of G. Note that G is really $(\exists z)(B(\mathbf{0}^{(m)}, \mathbf{0}^{(m)}, z) \wedge A(z))$, where B represents ϕ in Q. Note also that $\phi(m, m)$ is the Gödel number of G itself, since m is the Gödel number of $A'(x_1, x_1)$ and G is $A'(\mathbf{0}^{(m)}, \mathbf{0}^{(m)})$; so Q fi $B(\mathbf{0}^{(m)}, \mathbf{0}^{(m)},$

$\mathbf{0}^{(n)}) \wedge (y)(B(\mathbf{0}^{(m)}, \mathbf{0}^{(m)}, y) \supset y = \mathbf{0}^{(n)})$. So Q fi $A(\mathbf{0}^{(n)}) \supset (\exists z)(B(\mathbf{0}^{(m)}, \mathbf{0}^{(m)}, z) \wedge A(z))$, i.e. Q fi $A(\mathbf{0}^{(n)}) \supset G$; and Q fi $G \supset B(\mathbf{0}^{(m)}, \mathbf{0}^{(m)}, \mathbf{0}^{(n)}) \wedge A(\mathbf{0}^{(n)})$, so Q fi $G \supset A(\mathbf{0}^{(n)})$. Therefore, Q fi $G \equiv A(\mathbf{0}^{(n)})$.

The proof of the self-reference lemma that will be the preferred one in our treatment is perhaps the standard one nowadays, and uses some of the recursion theory that we have already developed. It is as follows. Let the formula A be given. Let $PH(x_1, x_2, y)$ be a formula that functionally represents Φ in Q (recall that Φ is a function that enumerates the unary partial recursive functions). Let ψ be a recursive function such that $\psi(m)$ is a certain Gödel number of $(\exists y)(PH(\mathbf{0}^{(m)}, \mathbf{0}^{(m)}, y) \wedge A(y))$. That there is one such recursive function is clear by the familiar reasons. In fact, we may naturally let ψ be just an S^m_n function for the given formula $(\exists y)(PH(x_2, x_2, y) \wedge A(y))$ (which we may take to have number e). Let f be an index of ψ. Let G be the sentence $(\exists y)(PH(\mathbf{0}^{(f)}, \mathbf{0}^{(f)}, y) \wedge A(y))$. $\Phi(f, f) = \psi(f) =$ a Gödel number of $(\exists y)(PH(\mathbf{0}^{(f)}, \mathbf{0}^{(f)}, y) \wedge A(y))$ = a Gödel number of G. Letting $n = \psi(f)$, Q fi $G \supset A(\mathbf{0}^{(n)})$ (since Q fi $G \supset PH(\mathbf{0}^{(f)}, \mathbf{0}^{(f)}, \mathbf{0}^{(n)}) \wedge A(\mathbf{0}^{(n)})$, as PH functionally represents Φ in Q), and Q fi $A(\mathbf{0}^{(n)}) \supset G$ (since Q fi $PH(\mathbf{0}^{(f)}, \mathbf{0}^{(f)}, \mathbf{0}^{(n)})$). Thus, Q fi $G \equiv A(\mathbf{0}^{(n)})$.

Through a similar proof we can obtain an effective version of the self-reference lemma:

Self-Reference Lemma. Effective Version: There is a recursive function ϕ such that for all formulae A(x) of the language of arithmetic (RE) in one free variable, if m is a Gödel number of A(x), then $\phi(m)$ is a Gödel number of a sentence G_m of the language of arithmetic (RE) such that Q fi $G \equiv A(\mathbf{0}^{(\phi(m))})$.

Proof: Let $PH(x_1, x_2, x_3, y)$ be a formula that functionally represents Φ^3 in Q (recall that Φ^3 is a function that enumerates the 2-place partial recursive functions). Let ψ be a 2-place recursive function such that if p is a Gödel number of a formula A(y), $\psi(q,p)$ is a certain Gödel number of $(\exists y)(PH(\mathbf{0}^{(q)}, \mathbf{0}^{(q)}, \mathbf{0}^{(p)}, y) \wedge A(y))$. This may be taken again to be an S^m_n function. Let f be an index of ψ, and let $\phi(p)=\psi(f,p)$. Then $\phi(p)$ will be a code of the sentence $G_p = (\exists y)(PH(\mathbf{0}^{(f)}, \mathbf{0}^{(f)}, \mathbf{0}^{(p)}, y) \wedge A(y))$, if p is a Gödel number of A(y). So if p is a Gödel number of A(y), $\Phi(f, f, p) = \psi(f,p) = \phi(p) =$ a Gödel number of $(\exists y)(PH(\mathbf{0}^{(f)}, \mathbf{0}^{(f)}, \mathbf{0}^{(p)}, y) \wedge A(y))$ = a Gödel number of G_p. Letting $r = \phi(p)$, Q fi $G_p \supset A(\mathbf{0}^{(r)})$ (since Q fi $G_p \supset PH(\mathbf{0}^{(f)}, \mathbf{0}^{(f)}, \mathbf{0}^{(p)}, \mathbf{0}^{(r)}) \wedge A(\mathbf{0}^{(r)})$, as PH functionally represents Φ^3 in Q), and Q fi $A(\mathbf{0}^{(r)}) \supset G_p$ (since Q fi $PH(\mathbf{0}^{(f)}, \mathbf{0}^{(f)}, \mathbf{0}^{(p)}, \mathbf{0}^{(r)})$).

The proofs of the self-reference lemma do not depend on the fact that A has only one free variable. Noting this allows us to state a more general version of the self-reference lemma in which G is allowed to have free variables.

Self-Reference Lemma with Free Variables: Let $A(x, y_1, ..., y_m)$ be a formula of the language of arithmetic (or RE) with all free variables shown; then there is a formula $G(y_1, ..., y_m)$ of the language of arithmetic (or of RE) such that Q fi $(y_1)...(y_m)(G(y_1, ..., y_m) \equiv$

$A(\mathbf{0}^{(n)}, y_1, ..., y_m)$), where n is a Gödel number of G.

The version of the self-reference lemma in which G does not have free variables is simply the special case of this lemma in which n = 0. Naturally, there is an effective version of the self-reference lemma with free variables.

A corollary of the self-reference lemma with free variables is the following:

Corollary: Let A(x, y) be a formula of the language of arithmetic (or RE) with all free variables shown; then there is a formula G(y) of the language of arithmetic (or of RE) with Gödel number n such that $(y)(G(y) \equiv A(\mathbf{0}^{(n)},y))$ and $(y)(A(\mathbf{0}^{(n)},y) \equiv Sat(\mathbf{0}^{(n)},y))$ are both true.

(In the case of RE, Sat(x,y) is W(x,y). In the case of the language of arithmetic, Sat(x,y), which we use to mean that y satisfies the formula of the language of arithmetic with Gödel number x, is not itself a formula of the language.)

The self-reference lemma with free variables might be given the name "self-reference lemma with parameters", but this name is more appropriate for the following variant of the lemma.

Self-Reference Lemma With Parameters: For any formula A(x), there is a recursive function ψ and a formula PS(x, y) that represents ψ in Q, such that for all m, $\psi(m)$ is the Gödel number of the formula $(\exists z)(PS(\mathbf{0}^{(m)}, z) \wedge A(z))$, and furthermore this formula is provably equivalent in Q to $A(\mathbf{0}^{(\psi(m))})$.

Proof: Let χ be a recursive function such that if m is the Gödel number of a formula $B(x_1, x_2)$, then $\chi(m, n, p)$ is the Gödel number of the formula $B(\mathbf{0}^{(n)}, \mathbf{0}^{(p)})$. Let CH(x, y, z, w) be a formula that represents χ in Q. Let n be the Gödel number of the formula $(\exists x_3)(CH(x_1, x_1, x_2, x_3) \wedge A(x_3))$, and let PS(x, y) be the formula $CH(\mathbf{0}^{(n)}, \mathbf{0}^{(n)}, x, y)$. PS represents the function $\psi(x) = \chi(n, n, x)$; to prove the theorem, we only have to show that $\psi(m)$ is the Gödel number of the formula $(\exists z)(PS(\mathbf{0}^{(m)}, z) \wedge A(z))$, for any m. Since n is the Gödel number of $(\exists x_3)(CH(x_1, x_1, x_2, x_3) \wedge A(x_3))$, it follows that $\psi(m) = \chi(n, n, m)$ = the Gödel number of $(\exists x_3)(CH(\mathbf{0}^{(n)}, \mathbf{0}^{(n)}, \mathbf{0}^{(m)}, x_3) \wedge A(x_3))$, which is the formula $(\exists x_3)(PS(\mathbf{0}^{(m)}, x_3) \wedge A(x_3))$.

Now, notice that $(\exists z)(PS(\mathbf{0}^{(m)}, z) \wedge A(z))$ is provably equivalent to $A(\mathbf{0}^{(\psi(m))})$. Thus, writing G(x) for $(\exists z)(PS(x, z) \wedge A(z))$, we have

$$Q \text{ fi } G(\mathbf{0}^{(m)}) \equiv A(\mathbf{0}^{(\psi(m))})$$

for all m, where $\psi(m)$ is a Gödel number of $G(\mathbf{0}^{(m)})$.

An alternative proof of the self-reference lemma with parameters consists in noting that we

may take G(x) to be the formula $(\exists y)(PH(\mathbf{0}^{(f)}, \mathbf{0}^{(f)}, x, y) \wedge A(y))$, as in the proof of the effective form of the self-reference lemma, and ψ to be the function ϕ of that same proof.

The Recursion Theorem

Kleene seemed to use the term 'the recursion theorem' as an ambiguous term for two theorems that he proved. Later, the two theorems came to be called Kleene's first and second recursion theorems. Generally speaking, the second recursion theorem is the more powerful of the two. Nowadays, it is usually called '*the* recursion theorem'. We discuss this theorem here (the first recursion theorem will come later). In terms of our formalism, it is simply the self-reference lemma for the language RE with formulae of two variables. Recall that for any 2-place relation R and number e, R_e is the set $\{n: R(e, n)\}$.

Recursion Theorem: For any 2-place r.e. relation R, there is an e such that $W_e = R_e$.

Before proving the recursion theorem, it is worth noting that the result is somewhat surprising. Any r.e. relation R can be thought of as enumerating a subclass of the r.e. sets (namely, the class $\{R_e: e \in \mathbf{N}\}$). We may thus call such a relation a *subnumeration* or the r.e. sets. The recursion theorem says that every subnumeration coincides with W at some point. Offhand, we might have thought that we could obtain a subnumeration which did not coincide with W at any point at all; R might be some scrambling of W, for example. The recursion theorem shows that this is not so.

Note that, since W_e is the set of numbers satisfying the RE formula with Gödel number e, the second recursion theorem says that for any r.e. relation R there is an RE formula A(x) with Gödel number e such that for all n, n satisfies A just in case R(e,n). Since R is itself defined by some RE formula B, this is just to say that for any RE formula B(y, x) of two free variables, there is an RE formula A(x) of one free variable such that for all n, A(x) is true of n iff $B(\mathbf{0}^{(e)}, x)$ is true of n, and so, that $(x)(A(x) \equiv B(\mathbf{0}^{(e)},x))$ is true, where e is the Gödel number of A(x). That is, the recursion theorem is really the self-reference lemma with free variables for RE in the case of one free variable. We can thus prove the recursion theorem by imitating the proof of the self-reference lemma, by considering an S_n^m function for the RE formula $(\exists z)(PH(x_2, x_2, z) \wedge B(z, y))$. This was also the inspiration for Kleene's original proof of the recursion theorem, although he was not working with RE, but with a different formalism. We shall give a proof which, although based essentially on the same underlying facts, is shorter and more common in textbooks.

Proof of the Recursion Theorem: Let R be any 2-place r.e. relation. Consider the relation $S(x, y) = R(\Phi(x, x), y)$. S is an r.e. relation, so apply the S_1^1 theorem to obtain a recursive function ψ such that for all m, $W_{\psi(m)} = S_m = R_{\Phi(m, m)}$. Since ψ is recursive, it has

an index f. Let $e = \psi(f)$; $W_e = W_{\psi(f)} = S_f = R_{\Phi(f, f)} = R_e$ (since $\Phi(f, f) = \psi(f) = e$).

This proof is breathtakingly short. It only uses the fact that W is an enumeration for which the statement of the S_n^m theorem holds.

In the same way that there is an effective version of the self-reference lemma with free variables, there is an effective form of the recursion theorem that is easy to state and prove: there is a recursive function ϕ such that for any 2-place r.e. relation R with index e, $W_{\phi(e)}=R_{\phi(e)}$.Of course, the effective version, like the noneffective, can be proved for all appropriate formalisms, and not just for RE.

The recursion theorem can be generalized to n+1-place r.e. relations. If R is an n+1-place relation, then let R_e be the relation $\{<x_1, ..., x_n>: R(e, x_1, ..., x_n)\}$; the general form of the recursion theorem states that for every n+1-place r.e. relation R, there is an e such that $W_e^{n+1} = R_e$.

Besides being surprising, the recursion theorem has curious consequences. Let R(x, y) be the relation W(x+1, y). Then $W_e = R_e = W_{e+1}$ for some e; so W enumerates the r.e. sets in such a way that at least one such set is listed two consecutive times. More generally, we see that for all n there is an e such that $W_e = W_{e+n}$; so W has many repetitions. (It is natural to ask whether a repetition-free enumeration of the r.e. sets exists; it turns out that such enumerations do exist, but are hard to construct.) Also, we can find a number e such that $W_e = \{e\}$; just let R(e, x) be the identity relation. Since this relation is certainly r.e., we can use the recursion theorem to find an e such that W(e, x) iff x = e, i.e. we can find a formula A(x) which is satisfied only by its own Gödel number.

More generally still, let ψ be any recursive function; by letting $R(x, y) = W(\psi(x), y)$, we see that $W_e = W_{\psi(e)}$ for some e. So we have the following

Theorem: For every recursive function ψ, there is an e such that $W_e = W_{\psi(e)}$.

This theorem looks superficially like a fixed-point theorem, and we will sometimes refer to it as 'the fixed-point version of the recursion theorem'. Notice, however, that it is not quite a fixed point theorem. A fixed point theorem states that a function F has a fixed-point, i.e. there is an a such that F(a) = a. On the one hand, the theorem does not show that ψ itself has a fixed point, since we can have $\psi(e) \neq e$ and $W_e = W_{\psi(e)}$. On the other hand, the "function" $F(W_e) = W_{\psi(e)}$ is not really a function at all, since its value depends not only on its argument, the set W_e, but also on the index e (we can have $W_e = W_f$ and $W_{\psi(e)} \neq W_{\psi(f)}$). By contrast, Kleene's first recursion theorem, which we shall eventually prove, really is a fixed-point theorem.

There is also a version of the recursion theorem for Φ. In fact, there are two versions, corresponding to the first version and to the fixed-point version.

Recursion Theorem for Partial Recursive Functions: (a) For all partial recursive ψ

there is an e such that $\Phi(e, x) = \psi(e, x)$, all x; and (b) for all total recursive ψ there is an e such that $\Phi(e, x) = \Phi(\psi(e), x)$, all x.

Proof: For (a), recall that Φ is really a uniformization of the relation W^3. Let PS(x, y, z) be the graph of ψ. Find an e such that $W_e = PS_e$, i.e. for all y and z, W(e, y, z) iff PS(e, y, z) iff $\psi(e, y) = z$; then W_e is single-valued, so W(e, y, z) iff $\Phi(e, y) = z$; so $\Phi(e, y) = \psi(e, y)$.

(b) is immediate from (a): let $\chi(x, y) = \Phi(\psi(x), y)$, and let e be an index of χ; then $\Phi(e, x) = \chi(e, x) = \Phi(\psi(e), x)$.

Form (b) is the form that usually is referred to as 'the recursion theorem' in the literature.

The recursion theorem is interesting mainly because the relation R can itself involve W, as we saw in the case $R(x, y) = W(\psi(x), y)$. To illustrate why this is useful, we shall give a proof, using the recursion theorem, that the factorial function is recursive. (This illustrates, by the way, why the theorem is called 'the recursion theorem'.) To show this, it suffices to show that the graph of the factorial function is recursive. If R is a relation such that

$$(*) \qquad R(x, y) \equiv (x = 0 \wedge y = 1) \vee (\exists n)(\exists z)(x = n+1 \wedge y = (n+1)\cdot z \wedge R(n, z)),$$

then R is the graph of the factorial function. (This can be seen by showing, by induction on x, that there is exactly one y such that R(x, y), and $y = x!$.) So we only have to find an r.e. relation R that satisfies (*). If R is r.e., then $R = W^3_e$ for some e, so an appropriate R exists just in case

$$W(e, x, y) \equiv (x = 0 \wedge y = 1) \vee (\exists n)(\exists z)(x = n+1 \wedge y = (n+1)\cdot z \wedge W(e, n, z))$$

holds for some e. Setting $S(e, x, y) \equiv (x = 0 \wedge y = 1) \vee (\exists n)(\exists z)(x = n+1 \wedge y = (n+1)\cdot z \wedge W(e, n, z))$, we see that S is r.e. and that $y = x!$ is recursive if

$$W(e, x, y) \equiv S(e, x, y)$$

for some e; but by the recursion theorem, such an e exists. We can similarly show that the Ackermann function is recursive. More generally, we can use the recursion theorem to find partial recursive functions that satisfy arbitrary systems of equations. For example, consider the system consisting of the two equations

$$\psi(0) = 1$$
$$\psi(n+1) = \psi(n)\cdot(n+1)$$

We can use an argument similar to the one given above to show that there is a partial recursive function satisfying these equations. In this case, we see that the function in question is total. In general, however, we cannot guarantee this. For example, let our

system of equations consist of just the equation $\psi(x) = \psi(x)+1$. This does indeed have a solution, namely the function which is undefined everywhere.

So far, we have not used the recursion theorem to prove anything that could not be proved already using the generated sets theorem. However, there are some important applications of the recursion theorem that go beyond the generated sets theorem. Unfortunately, these applications are not as easy to state as the ones just given, and presuppose some knowledge of transfinite ordinals. Just as we can define functions on the natural numbers by ordinary induction, we can define functions on the ordinals by transfinite induction; and if α is a limit ordinal, $f(\alpha)$ will in general depend on the infinitely many values $f(\beta)$ for $\beta < \alpha$. Thus, we cannot use the generated sets theorem to show that such a function is recursive, since we cannot generate $f(\alpha)$ until we have generated $f(\beta)$ for all $\beta < \alpha$, and at no stage have we actually generated infinitely values. Nonetheless, we can intuitively define f by a system of equations. For example, we might define ordinal exponentiation by

$$\alpha^0 = 1$$
$$\alpha^{\beta+1} = \alpha^\beta \cdot \alpha$$
$$\alpha^\beta = \sup\{\alpha^\gamma:\ \gamma < \beta\} \text{ when } \beta \text{ is a limit.}$$

In fact, we can use the recursion theorem to show that this system of equations defines a recursive function on the recursive ordinals (i.e. those ordinals which are order types of recursive well-orderings), in essentially the way we showed that the factorial function is recursive. (However, for this to make sense we need a way of coding up the recursive ordinals as natural numbers.) Thus, we can use the recursion theorem to get around the problem that the value of α^β depends on that of infinitely α^γ's for $\gamma < \beta$ when β is a limit. (Since what we are really defining is an *index* e of the ordinal exponentiation function, the set $\{\alpha^\gamma:\ \gamma < \beta\}$ is coded up in a finite way in terms of α, β and e; in effect, this is what allows us to talk about infinitely many values of the function at once.)

Exercises

1. (a) Let S be an r.e. set. Prove that there is a 1-1 recursive function χ such that for all m, $W_{\chi(m)} = \mathbf{N}$ if $m \in S$ and $W_{\chi(m)} = \emptyset$ if $m \notin S$.

(b) Show that K is 1-1 complete. (This is a result that has been long awaited.)

2. (a) Show that an r.e. set S is nonrecursive iff there is a total function ψ such that for all x, $\psi(x) \in S$ iff $\psi(x) \in W_x$. S is called *completely creative* if ψ is recursive, and *1-1 completely creative* if ψ is also 1-1. Observe that K is 1-1 completely creative, where ψ is the identity function.

(b) Prove that every completely creative set is many-one complete, and that every 1-1 completely creative set is 1-1 complete.

(c) Prove that if $S_1 \leq_m S_2$ and S_1 is completely creative, then so is S_2. Also show that if $S_1 \leq_1 S_2$ and S_1 is 1-1 completely creative, then so is S_2.

(d) Show that every many-one complete set is completely creative, and that every 1-1 complete set is 1-1 completely creative.

3. (a) Recall the set S of exercise 6 in Lecture XII. There is a formula $(x_2)L(x_1, x_2)$ with L in Lim that defines the complement of S. Why? Prove that if Γ is a consistent r.e. extension of Q, then only finitely many sentences of the form $(x_2)L(\mathbf{0}^{(m)}, x_2)$ are provable in Γ even though infinitely many such sentences are true. Hence conclude that if Γ is ω-consistent, all but a finite number of true sentences of the form $(x_2)L(\mathbf{0}^{(m)}, x_2)$ are undecidable.

(b) Prove that if the effective form of the Gödel theorem holds for an r.e. set T which is not recursive (in the sense in which it holds for K), then there is an infinite r.e. set that is disjoint from T. Conclude that though the noneffective form of the Gödel theorem holds for the set S of exercise 6 of the midterm assignment, S does not satisfy the effective form. (An r.e. set T with properties (b) and (c) of exercise 6 is called 'simple'. Observe that exercise 6 shows that every simple set is neither recursive nor 1-1 complete. Property (a) of the present exercise also follows from the fact that the set is simple.)

(c) Also show that if T is a nonrecursive r.e. set and T is completely creative, then T satisfies the effective form of Gödel's theorem. (It follows that K satisfies the effective form of Gödel's theorem, which we have already seen to be the case.)

Lecture XV

The Recursion Theorem with Parameters.

Let R be a 3-place r.e. relation, or in other words, a subnumeration of the 2-place r.e. relations. For any given m, let R^m be the relation $\{\langle e, x\rangle: R(e, x, m)\}$; R^m is a subnumeration of the r.e. sets. It follows from the recursion theorem that for any given m, there is an e such that $W_e = R^m_e$. But more is true: we can find e effectively from m.

Recursion Theorem with One Parameter: For any 3-place r.e. relation R, there is a recursive function ψ such that for all m, $W_{\psi(m)} = R^m_{\psi(m)}$ i.e. for all x and m, $W(\psi(m), x)$ iff $R(\psi(m), x, m)$.
Proof: First, let χ be a recursive function such that $W_{\chi(m, a)} = R^m_{\Phi(a,a)}$ for all m, a. Since $R^y_{\Phi(x,x)}$ is an r.e. relation, such a χ exists by the S^m_n theorem (taking m and a as the parameters). Next, let ϕ be a recursive function such that $\Phi(\phi(m), a) = \chi(m, a)$ for all m, a; ϕ is easily obtainable from a two place function α guaranteed by the S^m_n theorem for partial recursive functions, by taking an index of χ as fixed as the first argument of α. Finally, let $\psi(m) = \Phi(\phi(m), \phi(m)) = \chi(m, \phi(m))$. Then $W_{\psi(m)} = W_{\chi(m, \phi(m))} = R^m_{\Phi(\phi(m),\phi(m))} = R^m_{\psi(m)}$, all m.

This proof should be compared to the proof of the parameter-free recursion theorem; all we have done is to make the number f of that proof depend effectively on m. The theorem can be generalized to more than one parameter via the usual methods, i.e. either by imitation of the proof for one parameter, or via the pairing function.

The more usual statement of the theorem is this: for all 2-place recursive χ there is a 1-place recursive function ψ such that for all m, $W_{\psi(m)} = W_{\chi(\psi(m), m)}$. This follows from the version we have just proved: simply let $R(y, x, m)$ iff $W(\chi(y, m), x)$, and find a ψ such that $W_{\psi(m)} = R^m_{\psi(m)}) = W_{\chi(\psi(m), m)}$.

The recursion theorem with parameters has even spookier applications than the parameter-free version.

Arbitrary Enumerations.

We shall now take a different approach to the S^m_n and recursion theorems, by considering arbitrary enumerations of the r.e. sets rather than simply the specific relation W. This approach has the virtue of making the recursion theorem appear less mysterious than the usual presentation.

For most applications of either the recursion theorem or the S^m_n theorem, we don't need

any specific properties of the relation W except that it is an enumeration. For most applications of the S_n^m and recursion theorems, it suffices to have available the fact that there is some enumeration of the r.e. sets with the properties stated in the S_n^m and recursion theorems for W. Eventually, the approach that we will develop establishes that W has these properties, but it first "cooks up" enumerations with those properties. One can find in the literature the awareness that it is possible to cook up enumerations with the S_n^m property; however, the rest of the theory does not appear in the literature and is due to the author, who developed it without knowing that it had been developed for the S_n^m case.

Let W' be an enumeration of the r.e. sets. For each k, we can easily obtain an enumeration W'^k of the k-place relations from W' via the pairing function. The *diagonal enumeration* of an enumeration of the two-place r.e. relations $W'^2(x,z,y)$, $Diag(W'^2)$, is the relation $W'^2(x, x, y)$. We say that W' is a *recursion enumeration* (or that it has the *recursion property*) if for all r.e. two-place relations R there is an e such that $W'_e = R_e$. We also say that a subnumeration S is a *recursion subnumeration* if for all r.e. two-place relations R there is an e such that $S_e = R_e$; every recursion subnumeration is an enumeration: let A be an r.e. set and let R be the r.e. relation such that R(e,x) iff x is in A; then $A=R_e$ for every e; since S is a recursion subnumeration, there is an e such that $S_e = R_e=A$.

Theorem: For any enumeration in two variables $W'^2(x,z,y)$, its diagonal enumeration $Diag(W'^2)$ is a recursion enumeration.
Proof: That $W'^2(x,z,y)$ is an enumeration means that for every r.e. two-place relation R there is an e such that for all z,y, R(z,y) iff $W'^2(e,z,y)$. In particular, for every R there is an e such that for every y, R(e,y) iff $W'^2(e,e,y)$, i.e. for every R, $R_e=Diag(W'^2)_e$. This proves that $Diag(W'^2)$ is a recursion subnumeration of the r.e. sets, and hence, by our previous result, that it is a recursion enumeration.

This proof of the existence of a recursion enumeration of the r.e. sets from the existence of an enumeration of the two-place r.e. relations is as breathtakingly short as the standard proof that W has the recursion property, if not more so. However, it is much more natural and less mysterious than the latter. Suppose you had an enumeration of the 2-place r.e. relations, and you wanted to construct an enumeration of the r.e. sets with the recursion property. Each 2-place r.e. relation can be thought of as a list of r.e. sets, and the given enumeration of the r.e. relations can be thought of as a list of all these lists; in constructing an enumeration with the recursion property, what you really want to do is to construct a list $W^\dagger$ of r.e. sets which coincides with each of the other r.e. lists at some point. If R is the eth such list, what could be more natural than having $W^\dagger$ coincide with R at the eth place? This is just what we have done in defining $Diag(W'^2)$ above.

We say that W' is a *fixed-point enumeration* (or that it has the *fixed-point property*) if for all total recursive functions ψ there is an e such that $W'_e = W'_{\psi(e)}$. In calling these 'fixed

point enumerations' we are referring to the fact that the fixed-point version of the recursion theorem resembles a fixed point theorem (as we have pointed out, however, it is not really a fixed point theorem). By a proof similar to the proof of the fixed point theorem from the recursion theorem, we can prove the following

Theorem: Every recursion enumeration is a fixed-point enumeration.

The converse fails; that is, there are fixed-point enumerations which are not recursion enumerations.

Let us now define the notion of an enumeration that satisfies the S_n^m theorem. We can say that W' is a *substitution enumeration* (or that it has the *substitution property*) if for any 2-place r.e. relation R there is a 1-1 recursive function ψ such that $W'_{\psi(e)} = R_e$ for all e. Another way of stating the definition of a substitution enumeration is as follows. If R and S are subnumerations of the r.e. sets (i.e. 2-place r.e. relations), and ψ is a recursive function, let us say that ψ is a *translation* of R into S (in symbols, $\psi: R \rightarrow S$) if for all e, $R_e = S_{\psi(e)}$. Let us say that a subnumeration S is *maximal* if for every r.e. R, there is a recursive ψ such that $\psi: R \rightarrow S$; if we can require ψ to be 1-1, then we say that S is *1-1 maximal*. Translation is analogous to reducibility, and maximality (1-1 maximality) is analogous to m-completeness (1-completeness). Clearly, an enumeration is a substitution enumeration just in case it is 1-1 maximal. (A 1-1 maximal enumeration can also be called an *effective* enumeration) Assuming that an enumeration W' exists, it will follow that every maximal subnumeration S is an enumeration, because there will be a recursive function ψ such that for all e, $W_e = S_{\psi(e)}$, and so S enumerates the r.e. sets.

We shall now show that given any enumeration, we can find a 1-1 maximal enumeration.

Theorem: If W' is an enumeration of the r.e. sets, the relation W"([e, n], x) which holds iff $W'^2(e, n, x)$ is a 1-1 maximal enumeration.
Proof: Let W' be an arbitrary enumeration. Let W" be the enumeration such that W"([e, n], x) = $W'^2(e, n, x)$; W" is called the *pairing contraction* of W'^2. (Formally, W" is the r.e. relation defined by $(\exists e)(\exists n)(z = [e, n] \wedge W'^2(e, n, x))$. Note that $W''_z = \emptyset$ when z is not of the form [e, n].) To see that W" is 1-1 maximal, let R be any r.e. relation, and let $R = W'^2_{e_0}$. Let $\psi(n) = [e_0, n]$. $W''(\psi(n), x)$ iff $W'^2(e_0, n, x)$ iff R(n, x), so $W''_{\psi(n)} = R_n$. Since ψ is 1-1, ψ is a 1-1 translation of R into W".

Once we know that W" is a substitution enumeration, it follows that it is a recursion enumeration (and therefore a fixed-point enumeration). In fact, the standard proof of the recursion theorem using the S_n^m theorem establishes that every substitution enumeration is a recursion enumeration, since it doesn't appeal to any properties of W besides its being an enumeration. Actually, the following is also true:

Theorem: If W'_1 and W'_2 are enumerations such that for some recursive ψ, ψ: $W'_1 \rightarrow W'_2$, then, if W'_1 has the recursion property, W'_2 has also the recursion property.
Proof: That W'_1 has the recursion property means that for all r.e. two-place relations R there is an e such that $W'_{1e} = R_e$; that ψ: $W'_1 \rightarrow W'_2$ means that for all e, $W'_{1e} = W'_{2\psi(e)}$. We want to prove that for all r.e. two-place relations R there is an e such that $W'_{2e} = R_e$. Let R be an r.e. two-place relation. There is an e such that for all x, $W'_1(e,x)$ iff R(e,x). Consider the relation R'(x,y) which holds iff $R(\psi(x),y)$. This is an r.e. relation and so there is an e such that for all y, $W'_1(e,y)$ iff R'(e,y) iff $R(\psi(e),y)$ iff $W'_2(\psi(e),y)$. So $\psi(e)$ is such that $W'_{2\psi(e)} = R_{\psi(e)}$, and W'_2 has the recursion property.

The theorem has as an immediate corollary that a maximal enumeration must have the recursion property, since any recursion enumeration gets translated into it.

We mentioned that not every fixed-point enumeration is a recursion enumeration. A fixed-point enumeration which is maximal is also a recursion enumeration.

As we said, most of the results in recursion theory that use W really only depend on the fact that there is an enumeration with certain properties (specifically, the substitution property, the recursion property, and the recursion property with parameters); as far as recursion theory is concerned, little is gained by showing that the particular enumeration W has these properties, since a cooked up enumeration with those properties will in general do the job as well.

Lecture XVI

The Tarski-Mostowski-Robinson Theorem

Recall from lecture X that if Γ is a set of true sentences in the language of arithmetic, then every r.e. set is weakly representable in Γ. Specifically, if A(x) is a formula of RE that defines a set S, then $Q \supset A(x)$ weakly represents S in Γ: if $n \in S$, then Q fi $A(\mathbf{0}^{(n)})$, so *a fortiori* Γ, Q fi $A(\mathbf{0}^{(n)})$, and so by the deduction theorem, Γ fi $Q \supset A(\mathbf{0}^{(n)})$; if, on the other hand, Γ fi $Q \supset A(\mathbf{0}^{(n)})$, then $Q \supset A(\mathbf{0}^{(n)})$ is true (since it follows from a set of true sentences), and Q is true, so $A(\mathbf{0}^{(n)})$ is true and therefore $n \in S$. It follows that every r.e. set is 1-1-reducible to the set of theorems of Γ; if Γ is r.e., then the set of theorems of Γ is 1-complete. But whether or not Γ is r.e., Γ is undecidable.

Alfred Tarski, Andrzej Mostowski, and Raphael Robinson generalized this result, developing a technique for showing that various theories are undecidable. The theorem summing up this technique that we will state here, which says that certain theories are 1-1 complete, can be reasonably attributed to Bernays. We will call our basic result the 'Tarski-Mostowski-Robinson theorem', since it is essentially due to them, although Myhill and Bernays deserve credit for stating it in this form.

The basic idea behind the proof of the Tarski-Mostowski-Robinson theorem is to weaken the hypothesis that Γ be true (in the standard model of the language of arithmetic) in such a way that the argument of the last paragraph still goes through. We shall prove the theorem in stages, finding successively weaker hypotheses.

First, note that we can find a slight weakening of the hypothesis already. We already know that if Γ is a true theory in a language with two three-place predicates A and M for addition and multiplication (or, from an exercise, even with a single three-place predicate for exponentiation) then Γ is 1-complete. Weakening the hypothesis still further: suppose Γ is a theory in some language L' which contains the language L of arithmetic (or simply the language {A, M}) but contains extra vocabulary. Then the reasoning still goes through, as long as Γ has a model whose restriction to L is the standard model of L (or isomorphic to it). To see this, we need only verify that if Γ fi $Q \supset A(\mathbf{0}^{(n)})$ then $n \in S$ (where A(x) defines S in RE and 'Q' is some appropriate formulation of Q if the language considered is {A,M}). So suppose Γ fi $Q \supset A(\mathbf{0}^{(n)})$ and I is a model of Γ whose restriction to L is the standard model of L. Then $Q \supset A(\mathbf{0}^{(n)})$ is true in I and therefore in the standard model, since $Q \supset A(\mathbf{0}^{(n)})$ is a sentence of L. So we have the result that if Γ is a theory in some language L' which contains the language L of arithmetic (or simply which contains {A,M}) and Γ has a model whose restriction to L is the standard model of L (or isomorphic to it) then Γ is 1-complete.

Even in this form, the result is difficult to apply in practice, since, first, some theories we

might want to apply it to are formulated in languages which do not contain the language of arithmetic; and second, few if any interesting theories whose languages extend the language of arithmetic have models whose restriction to this language is isomorphic to the structure of the natural numbers. The full Tarski-Mostowski-Robinson theorem will show that the theories of various sorts of algebraic structures (e.g. groups, rings, etc.) are undecidable; to use the form of the theorem just mentioned to show that the theory of some class C of structures is undecidable, the structure <**N**, 0, ', +, ·> must be a member of C, and few if any such classes that have actually been studied include this structure. For example, we cannot yet show that the theory of rings is undecidable, since the natural numbers under addition and multiplication do not form a ring, as they are not closed under additive inverse.

However, the integers do form a ring, and moreover they include the natural numbers as a part. This suggests another weakening of the hypothesis that Γ is true: roughly, we shall show that as long as Γ has a model I such that the natural numbers under addition and multiplication are a submodel of I and they can be "picked out" using the language of Γ, then Γ is 1-complete. Actually, we shall prove a result that turns out to be equally powerful: we shall show that if Γ is a theory in some first-order language L, and L' is a language obtained from L by adding finitely many constants, and Γ has a model I in the language L' such that the natural numbers under addition and multiplication (or a structure isomorphic to this) are definable as a submodel of I, then the set of theorems of Γ in L is a set to which all r.e. sets are 1-1 reducible.

Tarski-Mostowski-Robinson Theorem: Let Γ be a theory in some first-order language L, and let L' be obtained from L by adding finitely many constants (possibly 0). Suppose Γ has a model I in the language L' such that the natural numbers are definable as a submodel of I. Then the set of theorems of Γ in L is a set to which all r.e. sets are 1-1 reducible.

The proof of the theorem will occupy us for the most part of the rest of this lecture.

As a first step to spelling the content of the theorem out, let Γ be a theory in some first-order language L, and let L' be obtained from L by adding finitely many constants. What does it mean to say that Γ has a model I in L' such that the natural numbers under addition and multiplication (or a structure isomorphic to this) are definable as a submodel of I? It means that there is a model I in L' of Γ and there are formulae N'(x), A'(x, y, z), and M'(x, y, z) of L' such that the structure $<I_{N'},I_{A'},I_{M'}>$ is the structure of the natural numbers under addition and multiplication (or a structure isomorphic to it), where $I_{N'}$={a: a satisfies N'(x) in I}, $I_{A'}$={<a,b,c> ∈ $I_{N'}^3$: <a,b,c> satisfies A'(x, y, z) in I}and $I_{M'}$={<a,b,c> ∈ $I_{N'}^3$: <a,b,c> satisfies M'(x, y, z) in I}.

If N', A' and M' are not already primitive predicate letters in L, we add corresponding predicates N, A, M to L and sentences (x)(N(x) ≡ N'(x)), (x)(y)(z)(A(x,y,z) ≡ A'(x,y,z)), (x)(y)(z)(M(x,y,z) ≡ M'(x,y,z)) as "definitional" axioms to Γ. We also add symbols for zero and successor and definitional axioms for them, as follows: (x)(N(x) ⊃ (x=**0** ≡ A(x,x,x)))

for zero, (y)(x)(N(x) ∧ N(y) ⊃ (x'=y ≡ (∃w)(N(w) ∧ (z)~A(z,z,z) ∧ M(w,w,w) ∧ A(x,w,y)))) for successor. The resulting theory is the set of consequences in L'∪{N,**0**,',A,M} of Γ plus the finite set D of definitional axioms.

Now, if B is a sentence, let B_N be the result of restricting all of B's quantifiers to N; that is, B_N comes from B by replacing (∃x)... by (∃x)(N(x) ∧ ...) throughout and (x)... by (x)(N(x) ⊃ ...) throughout. B_N is called the *relativization* of B to N. It is simple enough to show that B_N holds in I iff B holds in the submodel of I defined by N. Call Q_N the theory whose theorems are the consequences of a conjunction of the relativizations of the axioms of Q to N.

We then know that for every r.e. set S, if B(x) defines S in RE, then $B_N(x)$ is such that (1) $B_N(x)$ defines S on the natural numbers (or the copy of S in the structure defined by N) and (2) for all n, Q_N fi $B_N(\mathbf{0}^{(n)})$ iff n ∈ S. First, $B_N(x)$ clearly defines S (or the copy of S in the structure defined by N). Now, suppose that n ∈ S. Then for the usual reasons, fi Q ⊃ $B(\mathbf{0}^{(n)})$; it is easy enough to show that fi $Q_N \supset B_N(\mathbf{0}^{(n)})$, and therefore that Q_N fi $B_N(\mathbf{0}^{(n)})$. Now suppose that Q_N fi $B_N(\mathbf{0}^{(n)})$. Then $B_N(\mathbf{0}^{(n)})$ is true in the natural numbers (or in the structure defined by N), and so n ∈ S.

Now consider the theory $\Gamma+D+Q_N$, the set of consequences in the language L'∪{N,**0**,',A,M} of Γ plus the finite set D of definitional axioms, plus Q_N. Then for every r.e. set S, if B(x) defines S in RE, then $B_N(x)$ defines S (or the copy of S in the structure defined by N), and for all n, (i) if n ∈ S then $\Gamma+D+Q_N$ fi $B_N(\mathbf{0}^{(n)})$ (by the same reasoning as in the preceding paragraph) and (ii) if $\Gamma+D+Q_N$ fi $B_N(\mathbf{0}^{(n)})$ then n ∈ S, for suppose n ∉ S; B(x) defines S, so $B(\mathbf{0}^{(n)})$ is false, and so $B_N(\mathbf{0}^{(n)})$ is false in the structure defined by N, and hence in I; but $\Gamma+D+Q_N$ are true in I, so not $\Gamma+D+Q_N$ fi $B_N(\mathbf{0}^{(n)})$. (i) and (ii) establish, in other words, that $B_N(x)$ weakly represents S (or its copy) in $\Gamma+D+Q_N$.

Then, by the deduction theorem, for all n, n ∈ S iff Γ+D fi $Q_N \supset B_N(\mathbf{0}^{(n)})$. This indicates how to prove, using the familiar arguments employing the recursiveness of substitution, that S is 1-reducible to the set of theorems of Γ+D, i.e. that there is a 1-1 recursive function ψ such that n ∈ S iff ψ(n) is a Gödel number of a theorem of Γ+D; ψ(n) will be a Gödel number of a sentence of the form $(x)(x=\mathbf{0}^{(n)} \supset (Q_N \supset B_N(x))$. This shows that the set of theorems of Γ+D is 1-complete if it is r.e.

But we have not shown yet that every r.e. set is 1-reducible to the set of theorems of Γ (in the language L). Let us first see how the proof of this will go if we suppose that L and L' are the same, i.e., that no extra constants are aded to L, so that the definitional axioms only contain symbols from L and Γ+D is a theory in L∪{N,**0**,',A,M}. Intuitively, the addition of the new non-logical symbols by means of definitions does not add expressive power to L. More precisely, if B is a theorem of Γ+D then there is a translation B* of B into L, obtained by replacing "definienda" by "definientes" throughout, such that B* is a theorem of Γ (the converse trivially obtains). In other words, there is a function ϕ such that if m is a Gödel number of a sentence of the language L∪{N,**0**,',A,M}, ϕ(m) is a Gödel number of its translation into L. If we could show that we may require ϕ to be recursive and 1-1, then we

would have shown that every r.e. set is 1-reducible to the set of theorems of Γ (in the language L), because the composition of ϕ and ψ would be 1-1 and recursive, and would reduce S to the set of theorems of Γ.

In fact, we will show how to define directly, for each r.e. set S, a function β whose value for n is (a Gödel number of) a translation of $(x)(x=\mathbf{0}^{(n)} \supset (Q_N \supset B_N(x))$ (where $B_N(x)$ is as before). It is clear that the parts Q_N and $B_N(x)$ of one such formula are (recursively) translatable into appropriate formulae of L (a fixed translation Q* of the conjunction of the axioms of Q and a fixed formula B*(x) defining S (or a copy of it) in L). The part $x=\mathbf{0}^{(n)}$ is the only one that depends on n. Recall that L need not contain symbols for successor and zero. Now, clearly there is some formula $D_n(x)$ of L, obtained by repeated applications of the definitions for **0** and ' and Russell's trick, and such that the sentence $(x)(x=\mathbf{0}^{(n)} \equiv D_n(x))$ is a theorem of Γ+D. To obtain $D_n(x)$ in this way we would need a cumbersome application of the generated sets theorem. But we can obtain an appropriate formula $E_n(x)$ in a simpler fashion using the uniformization theorem. Notice that there must be a formula $E_n(x)$ of L such that $(x)(x=\mathbf{0}^{(n)} \equiv E_n(x))$ is a theorem of D alone (intuitively, we only need the definitions to prove an appropriate equivalence). But D is finite, so its set of theorems is r.e. Therefore the relation R={(n,m): m is a Gödel number of a formula E(x) and E(x) is in L and $(x)(x=\mathbf{0}^{(n)} \equiv E(x))$ is a theorem of D} is an r.e. relation, for the familiar reasons. Clearly for all n there is an m such that R(n,m). So R can be uniformized to a recursive function α such that α(n) is a Gödel number of a formula E_n such that $(x)(x=\mathbf{0}^{(n)} \equiv E_n(x))$ is a theorem of Γ+D (in fact, of D alone); α is clearly 1-1, because otherwise $(x)(x=\mathbf{0}^{(p)} \equiv x=\mathbf{0}^{(q)})$ for some p, q, p≠q would be a theorem of Γ+D, which is impossible, since that sentence must be true in a model isomorphic to the natural numbers, and any such model makes that sentence false.

Finally, β(n) will be definable in RE using concatenation as e.g. the least Gödel number of $(x)(E_n(x) \supset (Q^* \supset B^*(x))$, where $E_n(x)$ is cashed out in the definition of β in RE by means of α. β is thus clearly recursive and 1-1 (since α is). β 1-reduces S to the set of theorems of Γ, since for all n, $n \in S$ iff β(n) is a Gödel number of a theorem of Γ.

But we will have proved the Tarski-Mostowski-Robinson theorem only when we prove the same result without assuming that L' is equal to L. So far our proof only establishes (or can be minimally modified to establish) that every r.e. set is 1-reducible to the set of theorems of Γ in L', not in L. But we can easily show how to obtain recursively and in a 1-1 fashion, for a formula of the form $(x)(E_n(x) \supset (Q^* \supset B^*(x))$ possibly containing extra constants, a formula C of L (thus without extra constants) such that Γ fi $(x)((E_n(x) \supset (Q^* \supset B^*(x)) \equiv C)$. Since Γ is a theory in L, any property of the extra constants provable from Γ must be provable in Γ for arbitrary objects; thus, if $F(a_1,...,a_n)$ is provable from Γ, $(y_1)...(y_n)F(y_1,...,y_n)$ (where $y_1,...,y_n$ are the first variables that do not occur in $F(a_1,...,a_n)$) must be provable from Γ.

This concludes our proof of the Tarski-Mostowski-Robinson theorem.

(It may be remarked that we could have proved a weaker result which does not mention

extra constants at all. We will see how the addition of extra constants can be profitably applied in an exercise.)

Both Bernays and Myhill stated a theorem whose statement is closely related to the one we have given, although Myhill (and perhaps also Bernays) did not have an appropriate justification for it. The theorem they stated says that if a theory has a model with a definable submodel which is a model of Q, then the theory is 1-1 complete. This theorem is true (see the exercises) but it is harder to prove than our theorem. What Myhill and Bernays proved, essentially, was this theorem under the hypothesis that the theory is ω-consistent.

The Tarski-Mostowski-Robinson theorem can be applied to show that several algebraic theories are undecidable. Among them, the elementary theories of rings, commutative rings, integral domains, ordered rings, ordered commutative rings (all with or without unit), the elementary theory of fields, etc. The proof for the theory of rings is given as an exercise.

Despite its simplicity, the Tarski-Mostowski-Robinson theorem is a very striking result, since it states that for a theory to be undecidable, it is enough that it have just one model in which the natural numbers are definable as a submodel. Part of the reason it is so striking is that it is commonly applied to theories (like the theory of rings) for which there is no single standard interpretation. However, it is really no different in principle from the result that Q is undecidable. Q also has many different interpretations, but we tend to think of one particular interpretation as "standard" or "intended", so we are less surprised when that interpretation is used to show that Q is undecidable; nonetheless, mathematically speaking, using the standard interpretation of Q to show that it is undecidable is no different from using the fact that the integers form a ring to show that the theory of rings is undecidable.

If we have already shown that a given theory is decidable and that I is a model of that theory, it will follow that the set of natural numbers is not definable in I. For example, consider the model in the language of arithmetic whose domain is the real numbers. It is a famous theorem of Tarski that the first-order theory of this model (i.e. the set of sentences true in this model) is decidable; it follows from the Tarski-Mostowski-Robinson theorem that the set of natural numbers is not definable in this model. Similar remarks apply to the complex numbers. This also illustrates the fact that, for the theorem to apply, the formula that picks our the natural numbers must be a formula of the object language, since in the metalanguage we can certainly pick out the natural numbers from the real numbers.

Note also that the theorem relates the undecidability of Γ in L to the existence of a certain kind of model of Γ in a possibly larger language L'. It is important to notice that L' is only allowed to differ from L' by the addition of finitely many constants; the theorem does not hold if we allow L' to have additional predicates or function symbols as well. To see this, recall that the first-order theory Γ of the reals in the language L of arithmetic is decidable. However, letting $L' = L \cup \{N\}$ (where N is any unary predicate), we see that Γ is undecidable *in L'*: simply let I be the model for L whose domain is the set of reals, etc., let I' be the expansion of I to L' in which N is interpreted as applying to the natural numbers, and apply the Tarski-Mostowski-Robinson theorem.

Exercises

1. A classical theorem of elementary number theory says that every positive integer is the sum of four squares. Use this to prove that the elementary theory of rings is 1-1 complete. (Remark: For those who know about such things, the same argument can be used to prove that the elementary theories of commutative rings with or without unit, of integral domains, of ordered rings and ordered integral domains, etc. are 1-1 complete. It is more difficult to prove the 1-1 completeness of the elementary theory of fields, which uses a similar but more difficult method.)

2. (a) Show that the theorem that every maximal enumeration is a recursion enumeration can be proved using the method employed in the lectures to prove the self-reference lemma (with the recursion theorem for W as a special case). Remember that a maximal enumeration is one with the substitution property.

(b) Formulate an appropriate version of the recursion property with parameters, and prove that the diagonalization of any maximal subnumeration has the recursion property with parameters.

3. Recall the recursively inseparable sets S_1 and S_2 from the lectures.

(a) Let C be an r.e. set containing S_1 and disjoint from S_2. Prove that C is completely creative. Hint: Let A(x, y, z) be the r.e. relation $(y \in C \wedge z = \mathbf{0}') \vee (W(x, y) \wedge z = \mathbf{0})$. Let $\psi(x, y)$ be a uniformization of A(x, y, z). Prove that there is a recursive function χ such that $\psi(x, y) = \Phi(\chi(x), y)$, for all x, y. Prove that χ is a completely creative function for C.

(b) Give an example of a formula A(x) in the language L of arithmetic such that if Γ is any consistent r.e. extension of Q in L, then A(x) weakly represents a completely creative set in Γ. (A(x) need not represent the same completely creative set in all these systems.)

(c) Prove that if Γ is as above, every r.e. set is weakly representable in Γ.

(d) Prove that if Γ is as above, the set of all theorems of Γ is one-to-one complete.

Comment: this finally shows that the results we stated before under the hypothesis that Γ extends Q and is ω-consistent, concerning weak representability, 1-completeness, etc. all hold if ω-consistency is weakened to consistency. (Or almost all: this does not show that the result about *nice* weak representability still holds. This can also be proved, but requires another argument.) Rosser's work gave a start for this, but it took several decades to reach the point of this exercise.

(e) Use the results above to show how to prove the Tarski-Mostowski-Robinson results, stated in class under the hypothesis that Γ has a model with a definable submodel

isomorphic to the standard model of L, under the weaker hypothesis that Γ has a model with a definable submodel which is a model of Q.

Comment: Tarski, Mostowski and Robinson, as said in class, used a less model-theoretic formulation. However, their work would have implied the result in (e) with the conclusion of undecidability only. I know of no significant application to a specific theory, however, where the generalization to models of Q is really useful.

4. (a) An r.e. set S is *creative* if there is a recursive function ψ such that whenever W_x is disjoint from S, $\psi(x) \notin S \cup W_x$. Prove that every creative set is many-one complete. Hint: Let S* be any r.e. set. Prove that there is a recursive function χ such that, for all x,

$$W_{\chi(x)} = \begin{cases} \{\psi(\chi(x))\} \text{ if } x \in S^* \\ \emptyset \text{ if } x \notin S^* \end{cases}$$

(b) Conclude from what we have done so far that the concepts creative, completely creative, and many-one complete are equivalent. Also show that the concepts 1-1 complete, 1-1 completely creative, and 1-1 creative (defined in the obvious way) are equivalent. Later on it will turn out that all six concepts are equivalent.

(c) Another equivalent concept: prove that a set S satisfies the effective form of Gödel's theorem, as defined for nonrecursive r.e. sets, iff S is creative.

Comment: all of the concepts $\leq_m$, $\leq_1$, m-complete, 1-complete, creative, and simple are due to Post. Many theorems relating them are also due to Post, as (essentially) is the connection between creativeness and Gödel's theorem (which inspired the term "creative"). Other important properties of these concepts were proved by Myhill.

5. Show that the set of all valid formulae in the first-order language with one two-place predicate letter and no others, is 1-1 complete. Also show that the elementary theory of one irreflexive relation and the elementary theory of one asymmetric relation are 1-1 complete. Sketch of the method: consider a certain structure with set-membership as the only relation between elements of the structure. Set membership is irreflexive and asymmetric. The structure will consist of the natural numbers, the sets of natural numbers, the sets whose elements are natural numbers and sets of natural numbers, and so on, through all finite levels. Kuratowski defined the ordered pair $\langle x, y\rangle$ as $\{\{x\}, \{x, y\}\}$. Prove that this has the property of a pairing function: that is, if $\langle x_1, x_2\rangle = \langle y_1, y_2\rangle$ then $x_1 = y_1$ and $x_2 = y_2$. An ordered triple etc. can be defined in terms of ordered pairs. This pairing function is an important tool in the proof. The proof is much simpler if one realizes that definitions with extra constants are allowed.

Comment: some of you may know that a model of the natural numbers together with definitions of + and · can be defined in set theory. This fact could have been used to do this exercise, but the method given above presupposes much less prior background, and shows that this is not needed.

6. A *reduction class* is a recursive class of formulae in a first-order language such that there is an effective mapping ψ of arbitrary formulae of the full language of first-order logic (i.e. the language of first-order logic with all predicates and constants) into formulae of the class such that a formula A of the full language is valid iff ψ(A) is valid. Reduction classes were an active topic of research even before the development of recursion theory.

(a) A recursive class C of formulae is a reduction class iff the set of all (Gödel numbers of) valid formulae in C is . . . Fill in the dots with a concept already defined in this course, and prove the correctness of your answer.

(b) Give a non-trivial example of a reduction class, using the answer to part (a).

Lecture XVII

The Evidence for Church's Thesis.

In most courses on recursion theory, some mention is usually made of the evidence for Church's thesis. The evidence that is usually cited includes Turing's original analysis of the notion of computability which led to his definition of Turing machines, the now very considerable experience of recursion theorists in showing that intuitively computable functions can be shown to be recursive, and the fact that a large class of formal notions of computability have been proved equivalent. Here we shall discuss a piece of evidence for Church's thesis of a different kind.

Let Γ be any r.e. set of axioms in a language that includes the language of arithmetic but which may contain extra predicates and function symbols. Then the set of theorems of Γ will be r.e., and therefore any set or relation weakly representable in Γ will be r.e. (The proof that Γ's theorems form an r.e. set is just as before, except that the possibility of extra function letters makes matters a bit more complicated. In particular, the universal instantiation axiom will have to be given a more complicated set of restrictions.) Therefore, one way to show that a set or relation is r.e. is to find a suitable Γ in which it is weakly representable.

For example, we may use this method to show that the factorial function is recursive, by finding a Γ in which its graph is weakly representable. We form Γ by adding to the language of arithmetic the new unary function letter f and adding to the axioms of Q the following new axioms:

$$f(\mathbf{0}) = \mathbf{0};$$
$$(x)(f(x') = f(x)\cdot(x')).$$

(We could give similar axioms, and include an axiom of existence and uniqueness) for a two-place predicate letter instead of a function letter). It is easy enough to see that Γ fi $f(\mathbf{0}^{(n)}) = \mathbf{0}^{(k)}$ iff k = n!. (To see that k = n! implies Γ fi $f(\mathbf{0}^{(n)}) = \mathbf{0}^{(k)}$, argue by induction on n. To see that Γ fi $f(\mathbf{0}^{(n)}) = \mathbf{0}^{(k)}$ implies k = n!, we need only show that Γ is consistent; but the standard model for the language of arithmetic, expanded by interpreting f as the factorial function, is a model of Γ.) Thus the formula f(x) = y weakly represents the graph of the factorial function in Γ. We thus see how to show, in a wide range of cases, that a function is recursive by defining it by a system of equations.

We can also use this idea to give an informal argument for Church's thesis. If we have a set of discrete directions in classical mathematics, then it should be a corollary of our ability to construct appropriate formalisms to codify mathematical practice that that set of

directions codifies a recursive procedure. A computation procedure is a set of instructions which says what to do at any stage in the computation explicitly in terms of what went on at the previous stage. Thus, given the state of a system at stage n of a computation, the state at stage n+1 should follow as a matter of logic. Assuming that informal logical reasoning can be carried out within a formal deductive system, it ought to be possible to give a set Γ of axioms such that whenever A is a description of the state of the system at stage n of the computation and B is a description of stage n+1, then Γ, A fi B. Thus, if I is a description of the initial conditions, we should have Γ, I fi A whenever A is a description of the state of the system at stage n, for any n. If A does indeed follow from Γ, I, we know that it will be provable from them, by the completeness theorem. Moreover, since there are only finitely many instructions, Γ ought to be finite, and therefore r.e. Thus, any relation weakly representable in Γ will be r.e. If I(x) is a formula in the language of Γ that says that the computation starts with input n, and O(x) says that the computation eventually halts with output x, then we should have Γ fi $I(\mathbf{0}^{(n)}) \supset O(\mathbf{0}^{(k)})$ whenever input n yields output k; thus the formula $I(x) \supset O(y)$ will weakly represent the graph of the function that the procedure computes, and that function will therefore be partial recursive.

Besides being an argument for Church's thesis, the foregoing can be tightened up in particular cases to yield a proof that all functions computable by some particular sort of computation procedure are in fact partial recursive. For example, we could prove that all functions computable by a Turing machine are in fact partial recursive, by setting up a formal system Γ containing, besides the language of arithmetic, predicates relating to squares on the machine's tape and axioms relating the state of the system at one time to its state at the next time. This could be done by adding only finitely many extra predicates and only finitely many new axioms, so Γ would certainly be r.e. Then we could write out a formula I(x) which says that in its initial state, the tape contains marks representing the number x; and a formula O(x) which says that when the machine halts, the tape contains marks representing the number x. Then the formula $I(x) \supset O(y)$ will weakly represent the graph of the function that the machine computes.

Note that Γ may contain, besides new predicates of numbers, names of new objects besides the numbers; we may also give Γ an interpretation in which the domain contains objects besides natural numbers. That domain may contain squares on a Turing machine's tape, for example. We can still talk about the system Γ within the language RE, since Γ is still a collection of formulae, which we can code up as numbers, even though we are thinking of Γ as being about objects other than numbers.

Relative Recursiveness.

We have already seen, in our study of 1-1 and many-one reducibility, ways in which one decision problem can be "reduced" to another. If a set A is many-one reducible to a set B,

then if you had an "oracle" which told you, for any given number y, whether $y \in B$, then you could tell effectively whether $x \in A$ for any given x: simply compute $\phi(x)$ (where ϕ is the function that reduces A many-one to B) and ask the oracle whether $\phi(x) \in B$.

In general, a set A is said to be reducible to a set B if we can find a computation procedure for deciding membership in A which is allowed to use an oracle to B. In the case of many-one reducibility, the way in which the oracle can be used is very limited: it can only be consulted once in the course of the computation, for example. By allowing the oracle to be used in different ways, we get broader reducibility notions; in this section, we shall concentrate on the broadest such notion.

Let us say that a set S_1 is *semi-computable from a positive oracle for S_2* (or, is *semi-computable from a semi-computation of S_2*) if there is a semi-computation procedure for S_1 which is allowed to consult, at arbitrarily many times in the course of the computation, an oracle that gives positive information about S_2. That is, when the oracle is asked a question of the form "is $x \in S_2$?", it always answers "yes" if $x \in S_2$, but remains silent when $x \notin S_2$. The oracle reserves the right to take as long as it wants in answering any given question, so if at any given time the oracle has not answered, the semi-computation procedure cannot conclude that the answer is "no". The procedure can do other things while it is waiting for the oracle to answer; it can also ask the oracle several questions at once (or ask it a question before it has answered a previous question).

(Equivalently, rather than answering questions, the oracle could list the elements of S_2, not necessarily in order. Consulting the oracle about whether $x \in S_2$ would then amount to waiting for x to appear in the listing of S_2. This is the approach used by Hartley Rogers.)

Similarly, let us say that S_1 is *computable in S_2* if there is a computation procedure for S_1 which has an oracle to S_2, i.e. an oracle which gives both positive and negative information about S_2, which it is allowed to consult at arbitrary points in the computation. There is also a mixed notion: we say that S_1 is *semi-computable in S_2* if there is a semi-computation procedure for S_1 which has an oracle that gives both positive and negative information about S_2.

For all of these notions, we can allow, not just one oracle, but several oracles to several different sets. That is, we can say that S is semi-computable from semi-computations for S_1, ..., S_n if there is a semi-computation procedure for S with positive oracles to S_1, ..., S_n; and similarly for the other notions.

Given the notion of being semi-computable in a semi-computation of a set, we can define the other notions. For example, S_1 is semi-computable in S_2 just in case S_1 is semi-computable from semi-computations of S_2, $-S_2$, and S_1 is computable in S_2 if both S_1 and $-S_1$ are semi-computable in S_2. Equivalently, S_1 is semi-computable in S_2 if S_1 is semi-computable from a semi-computation of the characteristic function of S_2. (Here we identify the function ϕ with the set $\{[n, \phi(n)]: n \in \mathbf{N}\}$.)

Now let us give formal counterparts for these intuitive notions. Alongside the notion of semi-computability in a semi-computation, we have the notion of enumeration reducibility:

we say that S_1 is *enumeration reducible* to S_2 and write $S_1 \leq_e S_2$ if S_1 is definable in the language RE[P], the result of adding to RE the unary predicate P and interpreting it as applying to the elements of S_2. More generally, S is enumeration reducible to the sets S_1, ..., S_n if S is definable in RE[P_1, ..., P_n], the result of adding to RE the new unary predicates P_1, ..., P_n and interpreting each P_i (i = 1, ..., n) as applying to the elements of S_i. More generally still, we could add new k-place predicates (for k > 1) and new function symbols to RE and define a notion of enumeration reducibility to a collection of sets, relations, and/or functions. We shall see that all of this reduces to the case of enumeration reducibility to a single set.

We say that S_1 is *r.e. in* S_2 if $S_1 \leq_e S_2, -S_2$ (equivalently, iff $S_1 \leq_e$ the characteristic function of S_2), and that S_1 is *recursive in* or *Turing reducible to* S_2 ($S_1 \leq_T S_2$) iff both S_1 and $-S_1$ are r.e. in S_2. So $S_1 \leq_T S_2$ iff both $S_1 \leq_e S_2, -S_2$ and $-S_1 \leq_e S_2, -S_2$. We do not use a notation for "r.e. in" involving "≤", for it will turn out that the relation *S_1 is r.e. in S_2* is not transitive.

There is a relativized form of Church's thesis: a set S_1 is recursive in S_2 iff S_1 is computable in S_2 (or in terms of semi-computability, S_1 is enumeration reducible to S_2 iff S_1 is semi-computable from a semi-computation for S_2). As in the unrelativized form, there is an easy direction which we can prove (i.e. that anything satisfying the formal notion satisfies the informal notion) and a harder, converse direction which has not been proved.

Let us now check that the relations we have written with a "≤" are transitive. We have already checked this for $\leq_1$ and $\leq_m$, so we only have to check it for $\leq_e$ and $\leq_T$. Suppose $S_1 \leq_e S_2$ and $S_2 \leq_e S_3$. Then S_1 is defined by some formula A(x) in the language RE[P_2] and S_2 is defined by some formula B(x) in the language RE[P_3], where P_2 has as its extension the set S_2 and P_3 has as its extension the set S_3. Let C(x) be the formula obtained from A(x) by replacing each occurrence of $P_2(y)$ by B(y), for any variable y. P_2 and B define the same set, so A(x) and C(x) define the same set, namely S_1; but C(x) is a formula of RE[P_3], so S_1 is definable in RE[P_3], i.e. $S_1 \leq_e S_3$.

Now suppose that $S_1 \leq_T S_2$ and $S_2 \leq_T S_3$. Both S_1 and $-S_1$ are $\leq_e S_2, -S_2$, and both S_2 and $-S_2$ are $\leq_e S_3, -S_3$, so S_1 and $-S_1$ are $\leq_e S_3, -S_3$, i.e. $S_1 \leq_T S_3$. (Actually, for this to work, we need to use something slightly stronger than the transitivity of $\leq_e$ for single sets: we need that if $X \leq_e Y, Z$ and both Y and Z are $\leq_e U, V$, then $X \leq_e U, V$.)

However, this proof will not show that the relation *r.e. in* is transitive. Suppose we tried to show that this relation is transitive in this way. Given that S_1 is r.e. in S_2 and that S_2 is r.e. in S_3, we can conclude that $S_1 \leq_e S_2, -S_2$ and that $S_2 \leq_e S_3, -S_3$. But to show that S_1 is r.e. in S_3, we must show that $S_1 \leq_e S_3, -S_3$, and we can't conclude this from the transitivity of $\leq_e$, since we don't know that $-S_2 \leq_e S_3, -S_3$. In other words, given both positive and negative information about S_3, we only get positive information about S_2, and we need positive *and negative* information about S_2 to get positive information about S_1.

If a set S is r.e., then for all S_1, $S_1 \leq_e S$ iff S_1 is r.e. This is simply because RE[P] has the same expressive power as RE in this case, since the set P defines is already r.e.

Similarly, if S is recursive, then $S_1 \leq_T S$ iff S_1 is recursive. Thus, all r.e. sets bear $\leq_e$ to each other and collectively form a bottom element in the $\leq_e$ ordering; similarly, the recursive sets form a bottom element in the $\leq_T$ ordering. Observe that -K is not enumeration reducible to K, since K is r.e. but -K is not r.e.; however, $-K \leq_T K$, since $-S \leq_T S$ for any set S ($S \leq_T S$ by reflexivity, and whenever $S_1 \leq_T S_2$, $-S_1 \leq_T S_2$ by the definition of $\leq_T$). Notice also that -K is r.e. in K, since every set is r.e. in its complement, and that K is r.e. in Ø, since any r.e. set is r.e. in any other set. But -K is not r.e. in Ø, since -K is r.e. in Ø iff $-K \leq_e$ Ø, **N** iff -K is r.e. So transitivity fails for the relation *r.e. in*.

While the relation *r.e. in* is not transitive, it has the following "weak transitivity" property: if A is r.e. in B and B is recursive in C, then A is r.e. in C. To see this, suppose A is r.e. in B and $B \leq_T C$. Then $A \leq_e B, -B$ and both B and -B are $\leq_e C, -C$; so by the transitivity of $\leq_e$, $A \leq_e C, -C$, i.e. A is r.e. in C. If $A \leq_T B$ and B is r.e. in C, however, it does not follow that A is r.e. in C. (e.g. $-K \leq_T K$ and K is r.e. in Ø, but -K is not r.e. in Ø.) Nonetheless, if $A \leq_e B$ and B is r.e. in C, it does follow that A is r.e. in C, again by the transitivity of $\leq_e$.

The relations $\leq_e$ and $\leq_T$ are also reflexive, as is easily seen.

Let us now prove some elementary facts about our reducibility notions. First of all, both $S_1 \leq_e S_2$ and $S_1 \leq_T S_2$ imply that S_1 is r.e. in S_2, as is easily seen from the definitions. The converses fail, however. We also have that $S_1 \leq_1 S_2 \Rightarrow S_1 \leq_m S_2 \Rightarrow S_1 \leq_T S_2$, so the relations $\leq_T$, $\leq_m$, and $\leq_1$ are progressively stronger reducibility notions. (It is clear that $S_1 \leq_1 S_2$ implies $S_1 \leq_m S_2$; we shall see shortly that the other implication holds.)

$S_1 \leq_m S_2 \Rightarrow S_1 \leq_e S_2$: suppose $S_1 \leq_m S_2$, and let ϕ be a recursive function such that $x \in S_1$ iff $\phi(x) \in S_2$. Let F(x, y) define ϕ's graph in RE. Then $(\exists y)(F(x, y) \wedge P(y))$ defines S_1 in RE[P] (where P is given S_2 as its extension in RE[P]). Also, we have $S_1 \leq_m S_2 \Rightarrow -S_1 \leq_m -S_2 \Rightarrow -S_1 \leq_e -S_2$. So if $S_1 \leq_m S_2$, then $S_1 \leq_e S_2, -S_2$ (since $S_1 \leq_e S_2$), and $-S_1 \leq_e S_2, -S_2$ (since $-S_1 \leq_e -S_2$), so $S_1 \leq_T S_2$. We therefore have $S_1 \leq_m S_2 \Rightarrow S_1 \leq_T S_2$. Let us summarize what we have now proved:

$$\begin{array}{ccccc} S_1 \leq_1 S_2 & \Rightarrow & S_1 \leq_m S_2 & \Rightarrow & S_1 \leq_T S_2 \\ & & \Downarrow & & \Downarrow \\ & & S_1 \leq_e S_2 & \Rightarrow & S_1 \text{ r.e. in } S_2 \end{array}$$

In each case, the converse fails (though so far we have only proved this for the case $S_1 \leq_1 S_2 \Rightarrow S_1 \leq_m S_2$). Also,

$$\begin{array}{ccc} S_1 \leq_1 S_2 & \Leftrightarrow & -S_1 \leq_1 -S_2 \\ \Downarrow & & \Downarrow \\ S_1 \leq_m S_2 & \Leftrightarrow & -S_1 \leq_m -S_2 \\ \Downarrow & & \Downarrow \\ S_1 \leq_T S_2 & \Leftrightarrow & -S_1 \leq_T -S_2 \end{array}$$

The only part of this we haven't explicitly proved is the final equivalence $S_1 \leq_T S_2 \Leftrightarrow -S_1 \leq_T -S_2$. However, this follows trivially from the definition of $\leq_T$.

Notice that in proving that $S_1 \leq_m S_2 \Rightarrow S_1 \leq_T S_2$, we actually showed that $S_1 \leq_1 S_2$ implies that $S_1 \leq_e S_2$ and $-S_1 \leq_e -S_2$. When this relation obtains between S_1 and S_2, let us say that S_1 is *enumeration bireducible* to S_2 and write $S_1 \leq_{ee} S_2$. (Neither the term nor the notation is standard, as the notion has not been explored in the literature.) It is easy to see that $S_1 \leq_{ee} S_2$ implies both $S_1 \leq_e S_2$ and $S_1 \leq_T S_2$, but the converses do not obviously hold (and in fact are false). We can thus extend our diagram:

$$\begin{array}{lllll} S_1 \leq_1 S_2 & \Rightarrow S_1 \leq_m S_2 & \Rightarrow \quad S_1 \leq_{ee} S_2 & \Rightarrow & S_1 \leq_T S_2 \\ & & \Downarrow & & \Downarrow \\ & & S_1 \leq_e S_2 & \Rightarrow & S_1 \text{ r.e. in } S_2 \end{array}$$

Lecture XVIII

Recursive Union.

We will now show how the notion of enumeration reducibility to a relation, or to a function, can be reduced to the notion of enumeration reducibility to a set. In fact, the most obvious thing works here: if R is an n-place relation and S is a set, let R' = $\{[x_1, ..., x_n]: R(x_1, ..., x_n)\}$; then $S \leq_e R$ iff $S \leq_e R'$. Similarly, if ϕ is a total n-place function and F = $\{[x_1, ..., x_n, y]: \phi(x_1, ..., x_n) = y\}$, then $S \leq_e \phi$ iff $S \leq_e F$. To show this is simply to show that the languages $RE[P_1^1]$ and $RE[P_1^n]$ (resp. $RE[f_1^n]$) have the same expressive power, where P_1^n is interpreted as the relation R (resp. f_1^n is interpreted as the function ϕ), and P_1^1 is interpreted as the set R' (resp. as the set F). To show this, we simply show how a formula in either language can be translated into the other language. For simplicity, we concentrate on the case of the 2-place relation R. If A is a formula of $RE[P_1^2]$, let A* be the formula of $RE[P_1^1]$ that comes from A by replacing all occurrences of $P_1^2(x, y)$ by $P_1^1([x, y])$; and if B is a formula of $RE[P_1^1]$, let B† be the formula of $RE[P_1^2]$ that comes from B by replacing all occurrences of $P_1^1(x)$ by $(\exists y)(\exists z)(x = [y, z] \wedge P_1^1(y, z))$. It then suffices to check that A is equivalent to A* and that B is equivalent to B†.

We can use this result to show that being r.e. in a relation or function (resp. Turing reducibility to a relation or function) reduces to being r.e. in (resp. Turing reducibility to) a set. Again, we focus on binary relations for simplicity. Suppose S is r.e. in R; then $S \leq_e R$, -R, and the above proof will show that $S \leq_e R'$, -R', so S is r.e. in R'; the converse is proved similarly. (Matters are a bit delicate here, since -(R') is not the same set as (-R)'; so we really have to show that $S \leq_e R'$, (-R)' iff $S \leq_e R'$, -R'.) Now suppose $S \leq_T R$. Then both S and -S are r.e. in R, and so, as we have just seen, both S and -S are r.e. in R', so $S \leq_T R'$. Again, the converse is proved similarly.

We can also show that reducibility to several sets is nothing over and above reducibility to a single set. What we really want is a pairing function on *sets*; if π is such a function, then we want to show that $S \leq_e S_1, S_2$ iff $S \leq_e \pi(S_1, S_2)$. (This is analogous to our use of a recursive pairing function on *numbers* to reduce relations and functions to sets.) In fact, we do have a suitable pairing function. For any sets S_1 and S_2, we define the *recursive union* of S_1 and S_2 (written $S_1 \mathbf{U} S_2$) to be the set $\{2n: n \in S_1\} \cup \{2n + 1: n \in S_2\}$. It is easy to verify that the function $\mathbf{U}$ is indeed a pairing function on sets of natural numbers. In fact, it is an *onto* pairing function, i.e. every set S is $S_1 \mathbf{U} S_2$ for some S_1 and S_2. S_1 and S_2 are called the *even* and *odd parts* of S, respectively.

The idea behind recursive union is one that is familiar from other branches of mathematics: $S_1 \mathbf{U} S_2$ is the union of disjoint copies of S_1 and S_2. It is different from ordinary unions in the following striking way: whereas $-(S_1 \cup S_2) = -S_1 \cap -S_2$, $-(S_1 \mathbf{U} S_2) = -S_1 \mathbf{U} -S_2$. (Proof: the even part of $-(S_1 \mathbf{U} S_2)$ is $\{n: 2n \in -(S_1 \mathbf{U} S_2)\} = \{n: 2n \notin (S_1 \mathbf{U}$

$S_2)\} = -S_1$, and similarly for the odd part.)

It can then be shown that $S \leq_e S_1, S_2$ iff $S \leq_e S_1 \cup S_2$, and similarly that S is r.e. in S_1, S_2 iff S is r.e. in $S_1 \cup S_2$, and that $S \leq_T S_1, S_2$ iff $S \leq_T S_1 \cup S_2$. Using this it is easy to show how to reduce the case of several sets to the case of one set by iterating the recursive union function.

Observe that S_1 and S_2 are both 1-1 reducible to $S_1 \cup S_2$: in the case of S_1 by the map $x \rightarrow 2x$, and in the case of S_2 by the map $x \rightarrow 2x + 1$. It follows that S_1 and $S_2 \leq_m S_1 \cup S_2$, that S_1 and $S_2 \leq_e S_1 \cup S_2$, that S_1 and S_2 are r.e. in $S_1 \cup S_2$, and that S_1 and $S_2 \leq_T S_1 \cup S_2$. Thus, the set $S_1 \cup S_2$ is an *upper bound* of S_1 and S_2 with respect to all of these reducibility notions.

In fact, something more is true: for any set S*, if $S_1, S_2 \leq_e S^*$, then $S_1 \cup S_2 \leq_e S^*$; and the same holds for $\leq_T$. For suppose $S_1, S_2 \leq_e S^*$; then we have a formula A(x) and a formula B(x) of RE[P] that define S_1 and S_2, respectively; then the formula $(\exists x)((A(x) \wedge y = 2x) \vee (B(x) \wedge y = 2x + 1))$ defines $S_1 \cup S_2$ in RE[P]. Similarly, if S_1 and S_2 are r.e. in S*, then S_1 and S_2 are $\leq_e S^*, -S^*$, so $S_1 \cup S_2 \leq_e S^*, -S^*$, i.e. $S_1 \cup S_2$ is r.e. in S*. Now suppose S_1 and S_2 are Turing reducible to S*. They are r.e. in S*, so $S_1 \cup S_2$ is r.e. in S*. Also, $-S_1$ and $-S_2$ are both r.e. in S*, so $-(S_1 \cup S_2) = -S_1 \cup -S_2$ is r.e. in S*. So $S_1 \cup S_2 \leq_T S^*$.

Thus, besides being an upper bound of S_1 and S_2, the set $S_1 \cup S_2$ is a *least* upper bound of S_1 and S_2 with respect to $\leq_e$ and $\leq_T$, in the sense that whenever S_1 and S_2 are both reducible to a given set, $S_1 \cup S_2$ is also reducible to it.

Enumeration Operators.

Let us concentrate on the relation $\leq_e$. $S_1 \leq_e S_2$ iff S_1 is defined by some formula of RE[P]; let A(x) be such a formula. What set A(x) defines will depend on the extension of the new predicate P; in fact, the set A(x) defines is a *function* of P's extension. Given any formula of RE[P], we can therefore associate with it an operator ψ from sets to sets, such that $\psi(S)$ = the set defined by A(x) when P is given S as its extension. Such an operator is called an *enumeration operator*. We see that $S_1 \leq_e S_2$ just in case $S_1 = \psi(S_2)$ for some enumeration operator ψ. Note also that $\psi(S) \leq_e S$ for all ψ and S.

We can also allow ψ to have several arguments (by letting the corresponding formula have several extra predicates), and we can allow its values to be relations (by letting the corresponding formula have several free variables). So in general, for any n and k, each formula $A(x_1, ..., x_k)$ of $RE[P_1, ..., P_n]$ corresponds to an n-place enumeration operator from sets to k-place relations. (Or we could allow the arguments themselves to be relations, by considering formulae with extra non-unary predicates.) In general, we will be concerned with the cases in which n = 1. We do not require that k > 0; when k = 0, the values of the enumeration operator are truth values.

Let us now verify two important properties of enumeration operators. The first is *monotonicity*. An operator ψ is said to be monotonic if whenever $S_1 \subseteq S_2$, $\psi(S_1) \subseteq \psi(S_2)$. (When ψ takes as its values the truth values T and F, then we say that ψ is monotonic if whenever $S_1 \subseteq S_2$ and $\psi(S_1) = T$, then $\psi(S_2) = T$.) The second is *finiteness*. An operator is said to be finite if for all x and S, $x \in \psi(S)$ iff there is some finite $S_0 \subseteq S$ such that $x \in \psi(S_0)$.

Once we have proved monotonicity and finiteness for the case k = 0, the result will follow for all k > 0. To see this, suppose ψ is an enumeration operator corresponding to a formula $A(x_1, ..., x_n)$ of RE[P], and suppose $S_1 \subseteq S_2$. Suppose $<a_1, ..., a_n> \in \psi(S_1)$. Then $<a_1, ..., a_n>$ satisfies $A(x_1, ..., x_n)$ when P is interpreted as S_1, so the sentence $A(\mathbf{0}^{(a_1)}, ..., \mathbf{0}^{(a_n)})$ of RE[P] is true. By monotonicity for k = 0, $A(\mathbf{0}^{(a_1)}, ..., \mathbf{0}^{(a_n)})$ remains true when P is interpreted as S_2, and so $<a_1, ..., a_n>$ still satisfies $A(x_1, ..., x_n)$, i.e. $<a_1, ..., a_n> \in \psi(S_2)$. Since the n-tuple $<a_1, ..., a_n>$ was arbitrary, it follows that $\psi(S_1) \subseteq \psi(S_2)$. Similarly, suppose finiteness holds for the case k = 0, and suppose $<a_1, ..., a_n> \in \psi(S)$. Then $<a_1, ..., a_n>$ satisfies $A(x_1, ..., x_n)$ when P is interpreted as S, so $A(\mathbf{0}^{(a_1)}, ..., \mathbf{0}^{(a_n)})$ is true; by finiteness, $A(\mathbf{0}^{(a_1)}, ..., \mathbf{0}^{(a_n)})$ is true when P is interpreted as S_0 for some finite $S_0 \subseteq S$, so $<a_1, ..., a_n> \in \psi(S_0)$.

Theorem: Monotonicity holds for enumeration operators.
Proof: By the foregoing discussion, we need only show that if A is a sentence of RE[P] and A is true when P is interpreted as S_1, then A remains true when P is interpreted as S_2 whenever $S_1 \subseteq S_2$. (To save words, let us say that A is true *in S* to mean that A is true when P is interpreted as S.) We show this by induction on the complexity of RE[P] sentences. Atomic sentences are either sentences of RE or sentences of the form $P(\mathbf{0}^{(n)})$. The former are true or false independently of how P is interpreted, and $P(\mathbf{0}^{(n)})$ is true in S iff $n \in S$, so obviously the theorem holds for $P(\mathbf{0}^{(n)})$. If the theorem holds for A and B, and $A \wedge B$ is true in S_1, then both A and B are true in S_1 and by the inductive hypothesis remain true in S_2; therefore, $A \wedge B$ is true in S_2. The remaining cases offer no difficulty and are left to the reader.

Theorem: Finiteness holds for enumeration operators.
Proof: Again, we only have to show that a sentence A of RE[P] is true when P is interpreted as some set S iff A is true when P is interpreted as S_0 for some finite $S_0 \subseteq S$. The "if" part is trivial, by monotonicity. We prove the "only if" part by induction on the complexity of sentences. If a sentence is atomic, it is either a sentence of RE or of the form $P(\mathbf{0}^{(n)})$. In the former case, the interpretation of P is irrelevant to its truth, so we can take $S_0 = \emptyset$. In the latter case, if $P(\mathbf{0}^{(n)})$ is true in S, then $n \in S$, so we can take $S_0 = \{n\}$.

If $A \vee B$ is true in S, then either A or B is true in S; suppose A is. Then by the inductive hypothesis, A is true in S_0 for some finite $S_0 \subseteq S$, so the sentence $A \vee B$ is also true in S_0.

If $A \wedge B$ is true in S, then both A and B are true in S, so by the inductive hypothesis

there are finite sets S_1, $S_2 \subseteq S$ such that A and B are true in S_1 and S_2, respectively. So by monotonicity, both A and B are true in the finite set $S_0 = S_1 \cup S_2$, and so $A \wedge B$ is true in S_0.

If $(\exists x)A(x)$ is true in S, then the case is like disjunction. $A(\mathbf{0}^{(n)})$ is true for some n, so by the inductive hypothesis $A(\mathbf{0}^{(n)})$ is true in S_0 for some finite $S_0 \subseteq S$; so $(\exists x)A(x)$ is true in S_0.

If $(x < \mathbf{0}^{(n)})A(x)$ is true in S, then the case is like conjunction. $A(\mathbf{0})$, ..., $A(\mathbf{0}^{(n-1)})$ are all true in S, so by the inductive hypothesis they are true in finite sets S_1, ..., S_n, respectively. So by monotonicity, they are all true in the finite set $S_0 = S_1 \cup ... \cup S_n$. So the sentence $(x < \mathbf{0}^{(n)})A(x)$ is itself true in S_0.

The set S_0 is sometimes called a *finite support* for the sentence.

We can use the last theorem to prove a normal form theorem for sentences or RE[P]. Let A be such a sentence. A is true in S iff for some finite $S_0 \subseteq S$, A is true in S_0. (And by the discussion above, this also holds if A has free variables.) We can write out the right side of the "iff" in RE[P]. Let s be some variable that does not occur in A, and let A* be the result of replacing all occurrences of P(t) by $t \in s$, for t a term. A* is thus a formula of RE. Let $s \subseteq P$ abbreviate the RE[P] formula $(x < s)(x \notin s \vee P(x))$. Then A is equivalent to the formula $(\exists s)(s \subseteq P \wedge A^*)$. Thus, the extra predicate P can be segregated off, as it were, so that it only occurs in the conjunct $s \subseteq P$.

The normal form theorem gives us an enumeration theorem: to get an enumeration of the n-place relations definable in RE[P], we simply replace A* by the formula $W(e, s, x_1, ..., x_n)$. If an n-place relation R is definable in RE[P], then it is definable by a normal form formula $(\exists s)(s \subseteq P \wedge A^*(x_1, ..., x_n))$, and $A^*(x_1, ..., x_n)$ is equivalent to $W(\mathbf{0}^{(e)}, s, x_1, ..., x_n)$ for some e, so R is defined by the formula $(\exists s)(s \subseteq P \wedge W(\mathbf{0}^{(e)}, s, x_1, ..., x_n))$; so $(\exists s)(s \subseteq P \wedge W(e, s, x_1, ..., x_n))$ (in which e is now a variable) defines an n+1-place relation that enumerates the n-place relations definable in RE[P], since R was arbitrary. (e is an index of the relation R here.) In fact, this also gives us an enumeration of the enumeration operators; we will sometimes write ψ_e to denote the eth operator in this enumeration. (We could have also proved an enumeration theorem by imitating the proof of the enumeration theorem for RE.)

The Enumeration Operator Fixed-Point Theorem.

We shall now prove that every enumeration operator has a least fixed point, and that this fixed point is r.e. This theorem is closely related to Kleene's *first recursion theorem*.Kleene stated his first recursion theorem in terms of partial recursive functions, but, just as in the case of the second recursion theorem, we first give the version for r.e. sets and relations. We will consider Kleene's form of the theorem at the end of the section.

A fixed point of a function ϕ is an x such that $\phi(x) = x$, so in particular a fixed point of an enumeration operator ψ is a set S such that $\psi(S) = S$. A set S is said to be *closed* (under ψ) if $\psi(S) \subseteq S$. S is said to be *sound* if $S \subseteq \psi(S)$. So S is fixed iff S is both sound and closed. If S is a fixed point of ψ, we say that S is the *least* fixed point of ψ if $S \subseteq S'$ for all fixed points S' of ψ.

Theorem: Every monotonic operator has a least fixed point.
Proof: We shall give two proofs of this theorem; the first is shorter, but the second gives us more information about the least fixed point, and this information will be useful later.

Let $G = \cap\{S: S \text{ is closed}\}$. ($\{S: S \text{ is closed}\}$ is not empty, since we know that there is at least one closed point, namely **N**.) First, we show that G is closed. If S is closed, then $G \subseteq S$ by the definition of G, so $\psi(G) \subseteq \psi(S) \subseteq S$ by ψ's monotonicity and S's closedness. So $\psi(G) \subseteq S$ for all closed S, and therefore $\psi(G) \subseteq G$ by the definition of G, and G is closed. Next, we show that G is sound. Note that $\psi(G)$ is closed: $\psi(G) \subseteq G$ as we have seen, so $\psi(\psi(G)) \subseteq \psi(G)$ by monotonicity. Since $\psi(G)$ is closed, $G \subseteq \psi(G)$, so G is sound. So G is a fixed point. Finally, G is the least fixed point: if S is a fixed point, then S is closed, so $G \subseteq S$ by the definition of G.

In the second proof, we construct a fixed point by transfinite induction. Let $S_0 = \emptyset$, and for all n, let $S_{n+1} = \psi(S_n)$. After we have constructed S_n for all $n \in \mathbf{N}$, we let $S_\omega = \cup\{S_n: n \in \mathbf{N}\}$. In general, if S_α has been defined, we set $S_{\alpha+1} = \psi(S_\alpha)$, and if α is a limit ordinal (i.e. an ordinal which is not $\beta+1$ for any β), we set $S_\alpha = \cup\{S_\beta: \beta < \alpha\}$. We show by induction on α that S_α is sound for all α; given the definition of S_α, this means that $S_\alpha \subseteq S_{\alpha+1}$. Clearly, S_0 is sound. If S_α is sound, i.e. $S_\alpha \subseteq S_{\alpha+1}$, then $\psi(S_\alpha) \subseteq \psi(S_{\alpha+1})$ by monotonicity, i.e. $S_{\alpha+1} \subseteq S_{\alpha+2}$. Now let α be a limit ordinal, and suppose S_β is sound for all $\beta < \alpha$. Let $x \in S_\alpha$. By the definition of S_α, $x \in S_\beta$ for some $\beta < \alpha$, and $S_\beta \subseteq S_\alpha$. By monotonicity, $\psi(S_\beta) \subseteq \psi(S_\alpha) = S_{\alpha+1}$, and by the inductive hypothesis $S_\beta \subseteq \psi(S_\beta)$, so $x \in S_{\alpha+1}$. Since x was arbitrary, $S_\alpha \subseteq S_{\alpha+1}$.

So the sequence $<S_\alpha:$ α an ordinal$>$ is increasing. It can't be strictly increasing, since if it were, a new natural number would be added to S_α at each stage; so at an uncountable stage, uncountably many natural numbers would have been added, which is impossible. (We can make this precise, as follows. If $S_\alpha \neq S_{\alpha+1}$ for each α, then $S_{\alpha+1} - S_\alpha$ must be nonempty for each α, so let $\phi(\alpha)$ be the least element of $S_{\alpha+1} - S_\alpha$. Then ϕ is a 1-1 function from the ordinals into **N**. So if α is an uncountable ordinal, then ϕ maps $\{\beta: \beta < \alpha\}$ 1-1 into **N**, which is impossible.) So the sequence must stop increasing eventually, that is there must be a λ such that $S_\lambda = S_{\lambda+1}$; indeed there must be a countable such λ. But this means that $\psi(S_\lambda) = S_\lambda$, i.e. S_λ is a fixed point of ψ.

Finally, we can show that S_λ is the least fixed point by showing, by ordinal induction on α, that if S' is any fixed point, then $S_\alpha \subseteq S'$; it follows that $S_\lambda \subseteq S'$.

Exercises

1. Show that for all sets S, S_1, and S_2, $S \leq_e S_1, S_2$ iff $S \leq_e S_1 \cup S_2$.

2. Suppose S is a completely creative set, and let ψ be a completely creative function for S (i.e. for all x, $\psi(x) \in S$ iff $\psi(x) \in W_x$). First, show that there is a recursive function χ_1 such that $W_{\chi_1(x, s)} = W_x - \{a_1, ..., a_n\}$, where s codes $\{a_1, ..., a_n\}$, and a recursive function χ such that $W_{\chi(x, y)} = W_x \cup \{y\}$. Next, define α and ψ^* simultaneously, as follows. $\alpha(n, 0) = \chi_1(n, s)$, where s is the smallest code of $\{\psi^*(0), ..., \psi^*(n-1)\}$; $\alpha(n, m+1) = \chi(\alpha(n, m), \psi(\alpha(n, m)))$; $\psi^*(0) = \psi(0)$; and $\psi^*(n+1) = \psi(\alpha(n+1, q_0))$, where q_0 is the least q such that $\psi(\alpha(n+1, q))$ is distinct from all of $\psi^*(0), ..., \psi^*(n)$. Prove that ψ^* is total recursive, 1-1, and a completely creative function for S.

Use this and previous exercises to show that the notions 1-complete, m-complete, creative, 1-1 creative, completely creative, 1-1 completely creative and being an r.e. nonrecursive set satisfying the effective form of Gödel's theorem are all equivalent.

Remark: remember that I said that r.e. sets that arise naturally, as opposed to being cooked up by recursion theorists, are all either recursive or 1-1 complete. The latter case can be characterized in all the ways on the list above.

3. Use the method of axiomatizing in first-order logic, as given in class, to show that all Turing-computable functions are recursive.

4. Recall the self-reference lemma with parameters from class: if A(x) is any formula, there is a recursive function ψ and a formula PS(x, y) that represents ψ in Q, such that for all m, $\psi(m)$ is the Gödel number of the formula $(\exists z)(PS(\mathbf{0}^{(m)}, z) \wedge A(z))$, which is provably equivalent in Q to $A(\mathbf{0}^{(\psi(m))})$. Use this to prove that every r.e. set is *nicely* weakly representable in every consistent r.e. extension of Q, as follows. Let Γ be any consistent r.e. extension of Q and let S be any r.e. set. Let R(x, y) be a formula of Lim such that the formula $(\exists y)R(x, y)$ defines S. Let Pr(x, y) be a formula of Lim such that $(\exists y)Pr(x, y)$ defines the set of theorems of Γ. Let A(x) be the formula $(\exists z)(PS(x, z) \wedge (\exists y)(R(x, y) \wedge (w < y) \sim Pr(z, w)))$, where PS represents the function ψ such that $\psi(m)$ is the Gödel number of the formula $A(\mathbf{0}^{(m)})$. Show that A(x) weakly represents S in Γ, and moreover that A(x) defines S.

Remark: a previous exercise proved that every r.e. set is weakly representable in every consistent extension of Q, but not that the weak representation was nice. Shepherdson gave this as an alternative way of getting the earlier result, and then Kreisel pointed out that this method gives a nice weak representation.

5. Consider the language that is like RE except that the bounded universal quantifier is replaced by the ordinary unbounded universal quantifier. This can be called the *positive*

language of arithmetic, PL.

(a) Prove that the same sets and relations are definable in the (ordinary) language of arithmetic, L, as are definable in PL.

Now consider the language PL[P_1^1] obtained by adding to PL a single monadic predicate P_1^1, just as in the case of RE. Analogously to enumeration operators, we can define positive arithmetical operators. Also, let $\phi(S)$ be the set of all (Gödel numbers of) true sentences of PL[P_1^1], where the extra predicate P_1^1 is interpreted as S.

(b) Show that in contrast to the case of RE[P_1^1], $\phi(S)$ is not a positive arithmetical operator. Also show that every enumeration operator is a positive arithmetical operator. Show as well that positive arithmetical operators need not in general be finite.

(c) Prove that $\phi(S)$ is monotonic, and show how to deduce that every positive arithmetical operator is monotonic.

(d) In class it was proved that every monotonic operator has a least fixed point. Prove the following statements by similar methods: every monotonic operator has a unique largest fixed point. Also, for every monotonic operator, every sound point S has a least fixed point above S, and every closed point S has a largest fixed point below S.

Notice that by (c) the conclusions of (d) apply to ϕ and to every positive arithmetical operator. In particular, they all have least fixed points and largest fixed points.

(e) If P_1^1 is interpreted by any fixed point of ϕ, show that the language PL[P_1^1] contains its own truth predicate and its own satisfaction predicates $Sat_k(x, m_1,...,m_k)$, for each k.

(f) The self-reference lemma for the language PL[P_1^1] (for the case of formulae with one free variable) says that for any formula of this language $A(x_1)$, with only x_1 free and x_1 never bound, there is a formula G with Gödel number m such that, independently of the interpretation of the extra predicate P_1^1, $G \equiv A(\mathbf{0}^{(m)})$ is always provable from the axioms of Q, if we consider the theorems of Q derivable in the broad language of arithmetic supplemented by the predicate P_1^1. A corollary is that $G \equiv A(\mathbf{0}^{(m)})$, where m is the Gödel number of G, is always true, regardless of how the extra predicate is interpreted. Prove the self-reference lemma for PL[P_1^1]. (In fact, all the forms of the self-reference lemma proved in class for the language of arithmetic generalize over to this case in a similar manner. However, here we only consider the form of the lemma we need for part (g).)

(g) Consider a sentence G such that $G \equiv P_1^1(\mathbf{0}^{(m)})$ (where m is the Gödel number of G) is true, regardless of the interpretation of P_1^1. Such a sentence exists by (f). Prove that there is at least one fixed point S_1 of ϕ such that if P_1^1 is interpreted by S_1, G is true, and another fixed point S_2 such that if P_1^1 is interpreted by S_2, G is false. Prove that G is true if P_1^1 is interpreted by the largest fixed point of ϕ and false if P_1^1 is interpreted by the least fixed point of ϕ. (This shows that there are at least two distinct fixed points and that in fact the largest and the least fixed points are distinct. In fact, the number of fixed points is the cardinality of the continuum.)

Remark: this finally shows that a language even with unbounded quantifiers of both

kinds, and with an expressive power greater than or equal to the language of arithmetic, can express its own truth and satisfaction predicates, as long as it lacks negation. Any interpretation of P_1^1 in PL by a fixed point has these properties. We have seen that there is more than one such interpretation of P_1^1. The same argument could be used for RE[P_1^1], but it is less interesting there, because *all* the languages RE[P_1^1], and RE itself, contain their own truth and satisfaction predicates.

The construction is related to one I have discussed elsewhere, but differs in that it is for classical languages without negation.

Lecture XIX

The Enumeration Operator Fixed-Point Theorem (Continued)

We could adapt either proof that every monotonic operator has a least fixed point to give us additional information. For example, for any sound point S, there is a least fixed point S' such that $S \subseteq S'$. We can show this either by letting $G = \cup\{S': S' \text{ is closed and } S \subseteq S'\}$ in the first proof, or by letting $S_0 = S$ in the second proof. Also, for any closed point S', there is a *greatest* fixed point $S \subseteq S'$. Again, we can imitate the first proof (switching "closed" and "sound" and making similar changes throughout) or fiddle with the second proof (letting $<S_\alpha>$ be a decreasing sequence with $S_0 = S$).

In the second proof, we know that a fixed point is reached at some countable stage. If the operator ψ is finite, then it is reached at stage ω.

Theorem: If ψ is a monotonic and finite operator, then the set S_ω from the proof of the last theorem is ψ's least fixed point.
Proof: We only have to show that $S_\omega = S_{\omega+1}$; since $S_\omega \subseteq S_{\omega+1}$, we just have to show that $S_{\omega+1} \subseteq S_\omega$. Let $x \in S_{\omega+1} = \psi(S_\omega)$. By finiteness, we can find a finite $X \subseteq S_\omega$ such that $x \in \psi(X)$. Since X is finite, $X \subseteq S_n$ for some $n < \omega$. By monotonicity, $x \in \psi(S_n)$. But $S_{n+1} = \psi(S_n)$, so $x \in S_{n+1} \subseteq S_\omega$.

Since enumeration operators are finite and monotonic, we know already that each enumeration operator has a least fixed point, and that it is constructed by stage ω. To show that this fixed point is r.e., we need to generalize the generated sets theorem slightly.

When a set is generated from a basis set and a collection of rules in the sense of the usual generated sets theorem, the rules are finite in number and each has a fixed finite number of premises. However, since we can code up finite sets of numbers as individual numbers, we can make sense of an r.e. generating rule having a *variable* finite number of premises. Specifically, we can identify such a rule with a binary relation R(s, x), where s codes a finite set of premises and x is the conclusion. We can formulate an appropriate notion of proof sequence for such a relation; specifically, we may say that $<x_1, ..., x_n>$ is a proof sequence for R if for every $i \leq n$ there is a finite set s such that all elements of s are in $<x_1, ..., x_n>$ before x_i and $R(s, x_i)$. Then naturally we define the set generated by R to be the set of all numbers that have proof sequences. As long as R is r.e., the notion of proof sequence will be r.e., and therefore the set generated by R will also be r.e. We leave the details to the reader.

Enumeration Operator Fixed Point Theorem: Every enumeration operator has a least

fixed point (namely S_ω), and that fixed point is r.e.

Proof: Let ψ be an enumeration operator, and let S_ω be as above. Let R be the relation {<s, x>: s codes a set S such that $x \in \psi(S)$}. R is r.e., as is easily seen (if A is the RE[P] formula corresponding to ψ, then R is defined by the formula A' obtained from A by replacing P(t) by $t \in s$ throughout). Let G be the set generated by R, which is therefore r.e. We show that $G = S_\omega$.

First, $G \subseteq S_\omega$. We show by induction on the length of proof sequences that if x occurs on a proof sequence, then $x \in S_\omega$. Let $<x_1, ..., x_n>$ be a proof sequence for R. Then R(s, x_n), where s codes a finite set of x_i's, $i < n$. By the inductive hypothesis, s codes a subset S of S_ω, so by monotonicity $\psi(S) \subseteq \psi(S_\omega) = S_\omega$. Since $x_n \in \psi(S)$, $x_n \in S_\omega$.

Next, $S_\omega \subseteq G$. We show by induction on n that $S_n \subseteq G$. $S_0 = \emptyset \subseteq G$. Suppose $S_n \subseteq G$, and let $x \in S_{n+1} = \psi(S_n)$. By finiteness, $x \in \psi(S)$ for some finite $S \subseteq S_n$. Since $S_{n+1} \subseteq G$, we can find proof sequences for all the elements of S; by stringing them together, we can find a proof sequence for x. So $x \in G$.

Kleene's first recursion theorem is stated in terms of partial recursive functions. An enumeration operator that maps partial functions into partial functions is called a *partial recursive operator*; Kleene showed that every partial recursive operator has a least fixed point, and that this fixed point is a partial recursive function. We can prove this using the enumeration operator fixed point theorem as follows. By identifying partial functions with their graphs, and identifying relations with sets of coded pairs, we can see that any partial recursive operator ψ has a least fixed point R, where R is an r.e. relation. To see that R is single valued, we use the fact that R is R_ω. $R_0 = \emptyset$ is single valued; if R_n is single valued, then since y is a partial recursive operator, $R_{n+1} = \psi(R_n)$ is single valued; so each R_n is single valued. Suppose [x, y] and $[x, z] \in R_\omega$. Then for some m, n, $[x, y] \in R_m$ and $[x, z] \in R_n$. Let $p = \max(m, n)$; then $[x, y], [x, z] \in R_p$. Since R_p is single valued, $y = z$. So R_ω is single valued.

The First and Second Recursion Theorems.

Here are the two recursion theorems:

(1) For all enumeration operators ψ, there is a least set S such that $\psi(S) = S$, and moreover S is r.e.
(2) For all recursive functions ϕ, there is an e such that $W_e = W_{\phi(e)}$.

Neither of these theorems implies the other. On the one hand, the second recursion theorem implies that every enumeration operator has an r.e. fixed point, but not that it has a least fixed point. To see this, let ψ be any enumeration operator, and let A be a formula of RE[P]

corresponding to it. Let A*(e, x) be the formula of RE obtained from A by replacing P(x) by W(e, x) throughout. Then A*(e, x) defines the relation $\{<e, n>: n \in \psi(W_e)\}$, so that relation is r.e. Using the S^m_n theorem, we can find a recursive function ϕ such that $W_{\phi(e)} = \psi(W_e)$ for all e. To find an r.e. fixed point for ψ, apply (2) to find an e such that $W_e = W_{\phi(e)} = \psi(W_e)$. W_e need not be the least fixed point, however.

On the other hand, we can use (1) to prove (2) only in a special case. Since the ϕ in (2) is a function on numbers rather than sets, it is quite possible that $W_e = W_f$ and $W_{\phi(e)} \neq W_{\phi(f)}$ for some e and f; in that case, the "operator" $F(W_e) = W_{\phi(e)}$ will not even be well-defined, let alone an enumeration operator. However, let us say that a function ϕ is *extensional* if for all e and f, if $W_e = W_f$ then $W_{\phi(e)} = W_{\phi(f)}$. Then the operator $F(W_e) = W_{\phi(e)}$ is well-defined. It turns out that whenever ϕ is extensional, there is an enumeration operator ψ such that $\psi(W_e) = W_{\phi(e)}$ for all e. We can thus apply (1) to ψ to obtain an e such that $W_e = \psi(W_e) = W_{\phi(e)}$.

If we only applied (2) with extensional ϕ in practice, then this would not be much of a limitation. However, there are important applications of (2) in which ϕ is nonextensional, or at least in which there is no good reason to think that ϕ is extensional; the study of recursive ordinals is an example of this.

(1) and (2) have many applications in common. For example, we can use (1), as we used (2), to prove that certain functions defined in terms of themselves are recursive. Take the factorial function, for example. We can define a partial recursive operator as follows: $\Psi(\phi) = \chi$, where $\chi(0) = 0$ and $\chi(n+1) = \phi(n)\cdot(n+1)$. It is easy to check that Φ is a partial recursive operator; applying the version of (1) for such operators, we see that there is a partial recursive ϕ such that $\Psi(\phi) = \phi$, so that $\phi(0) = 0$ and $\phi(n+1) = \phi(n)\cdot(n+1)$, i.e. $\phi(n) = n!$ for all n. (In fact, this proof that the factorial function is recursive boils down to the proof we gave earlier in terms of the generated sets theorem; the operator Ψ is really a kind of generating rule.) (1) and (2) are called recursion theorems because of these common applications.

The Intuitive Reasons for Monotonicity and Finiteness.

We have shown, in terms of our formalism, that enumeration operators are monotone and finite; we can also give intuitive proofs of the corresponding claims about the intuitive notion of semi-computability. Let P be a semi-computation procedure which consults an oracle; let us say that P semi-computes a set S_1 from S_2 if, whenever P is given an oracle to S_2, it answers "yes" to input n iff $n \in S_1$. In this case, let us write $S_1 = P(S_2)$. We want to show that if $S_1 \subseteq S_2$ then $P(S_1) \subseteq P(S_2)$, and that if $n \in P(S)$, then $n \in P(S_0)$ for some finite $S_0 \subseteq S$.

Suppose $n \in P(S_1)$. Then whenever P is given an oracle to S_1 and gets input n, P halts after a finite amount of time with answer "yes". Since P halts after a finite amount of time,

P only asks the oracle finitely many questions, so a finite amount of information about S1 suffices for P to decide that $n \in P(S_1)$. Moreover, this information is positive information, since the oracle only gives "yes" answers to P's questions. Let $S_0 = \{x \in S_1$: the oracle gives an answer "yes" to the question "$x \in S_1$?"$\}$; then S_0 is finite, $S_0 \subseteq S_1$, and the information in S_0 suffices for P to decide that $n \in P(S_1)$. It follows that $n \in P(S_0)$ and that $n \in P(S_2)$ when $S_1 \subseteq S_2$: if P is given an oracle to the set S_0 (or S_2) and given input n, then it will proceed as it did when given an oracle to S_1, asking it exactly the same questions, and it will get the same "yes" answers, which suffice to make P halt and give an answer "yes".

We can use this fact to prove a normal form theorem for semi-computability. If S_1 is semi-computable in a semi-computation of S_2, then $S_1 = P(S_2)$ for some P, and by our monotonicity and finiteness result, $n \in P(S_2)$ iff $n \in P(S_0)$ for some finite $S_0 \subseteq S_2$. Now, the relation $n \in P(S_0)$ (holding between n and S_0) is clearly a semi-computable relation, since we can transform P into a procedure P* that semi-computes it: whenever P consults the oracle about whether n is an element of the set in question, let P* invoke a semi-computation procedure for the relation $n \in S_0$. Note that P* is a semi-computation procedure *without* oracles. We have thus shown that whenever a set S_1 is semi-computable in a semi-computation of S_2, there is a semi-computable relation R such that $S_1 = \{n: (\exists$ finite $S_0)(S_0 \subseteq S_2 \wedge R(n, S_0))\}$. If the unrelativized version of Church's thesis is true, then R must be r.e., and therefore there is an *r.e.* relation such that $S_1 = \{n: (\exists$ finite $S_0)(S_0 \subseteq S_2 \wedge R(n, S_0))\}$. But this holds precisely when $S_1 \leq_e S_2$. So the unrelativized version of Church's thesis implies the relativized version.

Degrees of Unsolvability.

Suppose a binary relation $\leq$ is reflexive and transitive; then the relation $\equiv$, defined by $a \equiv b$ iff $a \leq b$ and $b \leq a$, is an equivalence relation. To verify this, we must show that $\equiv$ is reflexive, symmetric, and transitive. That $\equiv$ is symmetric is immediate from the definition and does not depend on any properties of $\leq$. $\equiv$'s reflexivity follows from that of $\leq$. Finally, if $a \equiv b$ and $b \equiv c$, then $a \leq b$ and $b \leq c$ by the definition of $\equiv$, so $a \leq c$ by $\leq$'s transitivity, and similarly $c \leq a$, so $a \equiv c$. We have shown that all of our reducibility notions are reflexive and transitive, so in each case the relation of interreducibility is an equivalence relation. We write $A \equiv_e B$ for $A \leq_e B$ & $A \leq_e B$, and similarly for $\equiv_1$, $\equiv_m$, and $\equiv_T$. The equivalence classes are called *degrees of unsolvability*, or simply *degrees*. In particular, the $\equiv_e$-, $\equiv_1$-, $\equiv_m$- and $\equiv_T$- equivalence classes are called *enumeration degrees*, *1-degrees*, *m-degrees*, and *Turing degrees*, respectively. (We use lowercase letters to denote degrees.) The idea behind this terminology is that when a set A is reducible to a set B, B is harder to compute than A, i.e. the decision problem for B has a higher degree of difficulty than that of A. (Or the semi-decision problem, in the case of enumeration degrees.) Degrees, especially Turing

degrees, have been studied extensively.

Let us write $\deg_e(A)$ ($\deg_T(A)$, etc.) for the enumeration degree (Turing degree, etc.) of a set A, i.e. the degree of which A is a member. We can place an ordering on degrees corresponding to the reducibility relation between sets: we say that $\deg_T(A) \leq \deg_T(B)$ iff $A \leq_T B$ (and similarly for other kinds of degrees). It is easy to check that $\leq$ is well-defined, and that it partially orders the degrees. When "$\leq$" denotes this relation between degrees, we do not write a subscript: if a and b are both degrees of the same sort, then there is only one less-than relation which is defined between them, so if we know what sort of degrees a and b are, "$a \leq b$" is unambiguous. There is a least enumeration degree (under this ordering), namely the degree consisting of the r.e. sets. There is also a least Turing degree, namely the one consisting of the recursive sets.

If $\leq$ is one of our reducibility relations, we define $A < B$ to mean $A \leq B$ and not $B \leq A$. Equivalently, $A < B$ iff $A \leq B$ and not $A \equiv B$. Similarly, if a and b are degrees, we say that $a < b$ if $a \leq b$ and not $b \leq a$, or equivalently if $a \leq b$ and $a \neq b$; note that $\mathrm{Deg}(A) < \mathrm{Deg}(B)$ iff $A < B$.

The Jump Operator.

Recall that $A \leq_e B, C$ iff $A \leq_e B \mathbf{U} C$, so in particular, A is r.e. in B iff $A \leq_e B, \text{-}B$ iff $A \leq_e B \mathbf{U} \text{-}B$. Recall also that $A \mathbf{U} B \leq_e C$ iff $A \leq_e C$ and $B \leq_e C$. It follows that $A \leq_T B$ iff $A \mathbf{U} \text{-}A \leq_e B \mathbf{U} \text{-}B$.

Recall our enumeration of the sets enumeration reducible to a set S, namely the relation given by $(\exists s)(s \subseteq S \wedge W(e, x, s))$. Given a set S, we define S* to be the set $\{[e, m]: (\exists s)(s \subseteq S \wedge W(e, m, s))\}$. S* captures all the sets enumeration reducible to S, and is itself enumeration reducible to S, since we have in effect just defined S* in RE[P], with P interpreted as S.

Let us prove some basic properties of the * operator. First of all, for all A, if $A \leq_e S$, then $A \leq_1 S^*$. For suppose $A \leq_e S$; then A has some index e in the enumeration of the sets $\leq_e S$, so for all m, $m \in A$ iff $(\exists s)(s \subseteq S \wedge W(e, m, s))$ iff $[e, m] \in S^*$, so $A \leq_1 S^*$ by the map $m \rightarrow [e, m]$. It follows, by taking A = S, that $S \leq_1 S^*$ for all S. Since $S \leq_1 S^*$ implies $S \leq_e S^*$, we have $S \leq_e S^*$ and $S^* \leq_e S$, i.e. $S \equiv_e S^*$. We also have the following equivalences:

$$A \leq_1 S^* \Leftrightarrow A \leq_m S^* \Leftrightarrow A \leq_e S^* \Leftrightarrow A \leq_e S.$$

We have $A \leq_1 S^* \Rightarrow A \leq_m S^* \Rightarrow A \leq_e S^*$ immediately. $A \leq_e S^* \Rightarrow A \leq_e S$ because $S^* \equiv_e S$. Finally, $A \leq_e S \Rightarrow A \leq_1 S^*$, as we saw earlier.

In practice, we will forget about the exact definition of * and apply these equivalences directly. There are alternative definitions of * which would also yield these facts. The idea

behind our definition of * is that S* encodes an enumeration of all the sets $\leq_e$ S; to get this effect, we could have taken S* to be {[e, m]: m satisfies the formula of RE[P] whose Gödel number is e}. Or, since we can reduce satisfaction to truth, we could have taken S* to be the set of Gödel numbers of true sentences of RE[P]. Both of these sets are recursively isomorphic to S* as we actually defined it.

Another important equivalence involving * is the following:

$$A \leq_e B \Leftrightarrow A^* \leq_1 B^*.$$

For suppose $A \leq_e B$. Then since $A \equiv_e A^*$ and $B \equiv_e B^*$, $A^* \leq_e B^*$; but then $A^* \leq_1 B^*$. On the other hand, suppose $A^* \leq_1 B^*$. Then $A^* \leq_e B$ by the equivalences for *, and so $A \leq_e B$ since $A \equiv_e A^*$.

While S* is always $\leq_e$ S, -S* is *never* $\leq_e$ S. The proof is analogous to the proof that K is not recursive. Suppose -S* $\leq_e$ S, and let A = {m: [m, m] $\notin$ S*}; A $\leq_e$ -S*, so by the transitivity of $\leq_e$, A $\leq_e$ S. A has some index e; so for all m, m $\in$ A iff [e, m] $\in$ S*, and in particular, e $\in$ A iff [e, e] $\in$ S*; but e $\in$ A iff [e, e] $\notin$ S* by the definition of A, contradiction.

We now define S' to be the set (S **U** -S)*. S' is called the *jump* of S. (While the operation * is not a standard part of recursion theory, the jump operation is very standard.) Just as S* $\leq_e$ S, S' is r.e. in S: (S **U** -S)* $\leq_e$ S **U** -S by the properties of *, i.e. (S **U** -S)* is r.e. in S, i.e. S' is r.e. in S. However, -S' is never r.e. in S: if -S' is r.e. in S, then -(S **U** -S)* $\leq_e$ S **U** -S, which we have just seen to be impossible. So S' is never recursive in S. However, S $\leq_T$ S': by the basic properties of *, (S **U** -S) $\leq_e$ (S **U** -S)*, so S $\leq_e$ S' and -S $\leq_e$ S'. So S' is always of a higher Turing degree than S.

As in the case of *, the exact definition of ' is less important than its basic properties. We could have defined S' to be {[e, m]: m satisfies the formula of RE[P_1, P_2] with Gödel number e}, where P_1 and P_2 are interpreted as S and -S, respectively. We could also have defined S' to be the diagonal set {e: e satisfies the formula of RE[P_1, P_2] with Gödel number e}. In this way, we see that S' can be viewed as a relativization of K to the set S.

As with *, we have the following equivalences involving ':

$$A \leq_1 S' \Leftrightarrow A \leq_m S' \Leftrightarrow A \leq_e S' \Leftrightarrow A \text{ is r.e. in S.}$$

In general, we will forget about the definition of ' and work directly from these equivalences. Since A is r.e. in S iff A $\leq_e$ S **U** -S, this follows directly from our equivalences for * by replacing S by S **U** -S.

We also have the following:

$$A \leq_T B \Leftrightarrow A' \leq_1 B' \Leftrightarrow A' \leq_m B' \Leftrightarrow A' \leq_e B'.$$

$A' \leq_1 B' \Leftrightarrow A' \leq_m B' \Leftrightarrow A' \leq_e B'$ is immediate from the above. $A \leq_T B \Leftrightarrow A' \leq_1 B'$ is a special case of $A \leq_e B \Leftrightarrow A^* \leq_1 B^*$, replacing A and B by $A \cup \bar{A}$ and $B \cup \bar{B}$, respectively. Notice also that $A \leq_T B$ implies that $A' \leq_T B'$ ($A \leq_T B \Rightarrow A' \leq_1 B' \Rightarrow A' \leq_T B'$). However, the converse is false.

It follows immediately from this that $A \equiv_r B \Rightarrow A' \equiv_r B'$, whether $\equiv_r$ is $\equiv_1$, $\equiv_m$, or $\equiv_T$. Let us write this as $A \equiv_{1,\, m,\, T} B \Rightarrow A' \equiv_{1,\, m,\, T} B'$. If a is the degree of A (under one of these three reducibilities), then we define a' to be the degree of A'. We see that a' is well defined, because if $B \in \deg(A)$, then $B \equiv_r A$ (r = 1, m, or T), so $B' \equiv_r A'$, i.e. $\mathrm{Deg}(B') = \mathrm{Deg}(A')$. It also follows from the above that whenever $a \leq b$, $a' \leq b'$.

Thus, we see that the jump operator is an order-preserving map on the degrees. It can also be regarded as an *embedding* of the T-degrees into the 1-degrees, i.e. an isomorphism of the structure $\langle\{\text{T-degrees}\}, \leq\rangle$ onto a subset of the structure $\langle\{\text{1-degrees}\}, \leq\rangle$. More precisely, the map $\mathrm{Deg}_T(A) \rightarrow \mathrm{Deg}_1(A')$ is such an embedding. This is simply because $\mathrm{Deg}_T(A) \leq \mathrm{Deg}_T(B)$ iff $A \leq_T B$ iff $A' \leq_T B'$ iff $\mathrm{Deg}_1(A') \leq \mathrm{Deg}_1(B')$. In fact, the same argument shows that the map $\mathrm{Deg}_e(A) \rightarrow \mathrm{Deg}_1(A^*)$ is an embedding of the enumeration degrees into the 1-degrees.

There is also an embedding of the Turing degrees into the enumeration degrees. We have already seen that $A \leq_T B$ iff $A \cup \bar{A} \leq_e B \cup \bar{B}$; it follows that the map $\mathrm{Deg}_T(A) \rightarrow \mathrm{Deg}_e(A \cup \bar{A})$ is well-defined and is also an embedding. An enumeration degree in the range of this embedding is called *total*. Clearly, an enumeration degree is total just in case it contains a set of the form $A \cup \bar{A}$. An enumeration degree is also total iff it contains the graph of a total function (hence the name).

Let f be the embedding $\mathrm{Deg}_T(A) \rightarrow \mathrm{Deg}_e(A \cup \bar{A})$, and let g be the embedding $\mathrm{Deg}_e(A) \rightarrow \mathrm{Deg}_1(A^*)$. If we compose f and g, the result is an embedding h of the Turing degrees into the 1-degrees. Moreover, h is precisely the map $\mathrm{Deg}_T(A) \rightarrow \mathrm{Deg}_1(A')$ which we have already seen to be an embedding.

Lecture XX

More on the Jump Operator.

As we have seen, there is a least T-degree, namely 0, the set of all recursive sets. Using the jump operator, we can form an increasing sequence of degrees: 0, 0', 0", ... In general, we write $0^{(n)}$ (the *nth jump* of 0) for the result of applying ' to 0 n times. We know that this sequence is strictly increasing, i.e. $0 < 0' < 0''$ ..., since A' is never recursive in A.

If a and b are degrees, where a = deg(A) and b = deg(B), we define $a \cup b$ to be deg(A **U** B). For any of the four kinds of degrees we have been considering, $a \cup b$ is well-defined and is an upper bound of a and b (i.e. $a, b \leq a \cup b$). Moreover, if a and b are either Turing or enumeration degrees, $a \cup b$ is the least upper bound of a and b, i.e. for all degrees c, if $a, b \leq c$, then $a \cup b \leq c$.

First, let us verify that $a \cup b$ is well defined, i.e. that the degree of A **U** B does not depend on which sets A and B we pick from the degrees a and b. That is, we must show that if $A \equiv_r A_1$ and $B \equiv_r B_1$, then $A \mathbf{U} B \equiv_r A_1 \mathbf{U} B_1$ (for r = 1, m, T, e). We assume that $A \equiv_r A_1$ and $B \equiv_r B_1$, and show that $A \mathbf{U} B \leq_r A_1 \mathbf{U} B_1$ (as the proof that $A_1 \mathbf{U} B_1 \leq_r A \mathbf{U} B$ will be exactly the same). We know already from our previous work that $A \mathbf{U} B \leq_T A_1 \mathbf{U} B_1$ iff both A and B are $\leq_T A_1 \mathbf{U} B_1$, iff both A and B are $\leq_T A_1, B_1$. But $A \leq_T A_1, B_1$ since $A \leq_T A_1$ by hypothesis, and similarly $B \leq_T A_1, B_1$. The same holds for $\leq_e$. So consider the case r = m. $A \leq_m A_1$ and $B \leq_m B_1$, so let ϕ and ψ be recursive functions such that ϕ: $A \leq_m A_1$ and ψ: $B \leq_m B_1$. Let χ be the recursive function such that $\chi(2n) = 2\phi(n)$ and $\chi(2n+1) = 2\psi(n)+1$; χ: $A \mathbf{U} B \leq_m A_1 \mathbf{U} B_1$. Finally, if ϕ and ψ are 1-1, then χ is also 1-1, so $A \mathbf{U} B \leq_1 A_1 \mathbf{U} B_1$.

Next, since A **U** B is an upper bound of A and B in all of our reducibility notions, $\mathrm{Deg}_r(A)$ and $\mathrm{Deg}_r(B)$ are $\leq \mathrm{Deg}_r(A \mathbf{U} B)$ for all A and B, i.e. $a, b \leq a \cup b$. Finally, as we saw, $A, B \leq_{T,\, e} C$ implies $A \mathbf{U} B \leq_{T,\, e} C$, so $a, b \leq c$ implies $a \cup b \leq c$ if a, b, and c are enumeration degrees or Turing degrees.

We say that a partially ordered set is an *upper semilattice* if any two elements of it have a least upper bound, and a *lower semilattice* if any two elements have a greatest lower bound. A partially ordered set which is both an upper and a lower semilattice is called a *lattice*. Thus, we see that the degrees form an upper semilattice; however, it turns out that they do not form a lower semilattice, and hence do not form a lattice.

It is easy to check that the operator $\cup$ is associative and commutative, and that for all a_1, ..., a_n, $a_1 \cup ... \cup a_n$ is the least upper bound of a_1, ..., a_n. (These facts depend only on the fact that $a \cup b$ is the least upper bound of a and b.) Thus, any finite set of degrees has a least upper bound. It does not follow, however, that every set of degrees has a least upper bound. In fact, this is not the case: if F is a family of degrees, then for F to have a least upper bound, it is necessary and sufficient that there be a finite $E \subseteq F$ such that for all $a \in F$

there is a $b \in E$ with $a \leq b$. Thus, in particular, the sequence 0, 0', 0", ... has no least upper bound, since no such E exists.

Notice that 0' is the degree of Ø' (since 0 is the degree of Ø), and that Ø' is a 1-1 complete set. To see this, note that Ø' = (Ø **U** **N**)* = {odds}* = {[e, m]: $(\exists s)(s \subseteq \{odds\} \wedge W(e, m, s))$}; so if A is any r.e. set, the relation R given by R(x, y) iff $x \in A \wedge y \in \mathbf{N}$ is r.e., and therefore has an index e: so for all m, $m \in A$ iff for some (or any) s, R(m, s), iff W(e, m, s) for some such s, iff $[e, m] \in Ø'$. So $A \leq_1 Ø'$ by the map $m \rightarrow [e, m]$. Thus, we see that 0' is the degree of a 1-1 complete set.

Any set $S \in 0'$ is called *T-complete*, or simply *complete*. 0' contains sets which are not 1-complete; for example, Post's simple set is an element of 0'. In fact, Post invented this set in an attempt to solve what is known as *Post's problem*: the problem of finding an r.e. set which is neither recursive nor complete (or showing that there is no such set). A Turing degree is said to be an *r.e. degree* if it contains an r.e. set; so Post's problem is equivalently stated as the problem of whether there are any r.e. degrees other than 0 and 0'. Post failed in his search for such degrees, and it was conjectured by some that 0 and 0' are the only r.e. degrees there are. However, the problem was solved in 1956 by Friedberg and Mucnik (working independently). They proved this by finding two incomparable r.e. sets, i.e. sets A and B such that neither $A \leq_T B$ nor $B \leq_T A$. It follows that their degrees a and b are incomparable in the ordering ≤; since 0 and 0' are comparable, it follows that a can be neither 0 nor 0', since then it would be comparable with b.

Clearly, $0 \leq a \leq 0'$ for any r.e. degree a (since $Ø \leq_T A \leq_T Ø'$ for any r.e. set A); however, there are degrees between 0 and 0' which are not r.e. (It is easy to see that not all sets recursive in Ø' are r.e.: $-K \leq_T Ø'$ for example; it turns out that there are sets $\leq_T Ø'$ which are not even $\equiv_T$ any r.e. sets.) It is relatively easy to produce incomparable degrees between 0 and 0', but harder to produce r.e. degrees with this property.

It turns out (though we shall not prove this) that the jump operator is first-order definable from the relation ≤. That is, the graph of the jump operator is definable in the interpreted first order language whose domain consists of all the Turing degrees, and in which there is only a single binary relation which is interpreted as the ≤ relation between degrees.

The Arithmetical Hierarchy.

A Σ_n *formula* (for $n \geq 1$) is a formula consisting of a block of unbounded quantifiers, followed by a block of bounded quantifiers, followed by a quantifier-free formula, where the block of unbounded quantifiers begins with an existential quantifier, is of length n, and alternates between existential and universal quantifiers. (Thus, for example, $(\exists x)(y)(\exists z)\ x + y = z$ is a Σ_3 formula.) We also write "Σ^0_n" for "Σ_n". Since any formula of Lim is equivalent to a formula which consists of a string of bounded quantifiers followed by a

quantifier-free formula, every Σ_n formula is equivalent to a formula consisting of a string of n alternating quantifiers (of which the first is existential) followed by a formula of Lim. A *Π_n formula* (or Π_n^0) begins with a universal quantifier; otherwise the definition is the same. We sometimes call a formula Σ_n (or Π_n) if it is equivalent to a Σ_n (Π_n) formula.

A *set or relation* is said to be Σ_n (Π_n) if it is defined by a Σ_n (Π_n) formula. A set or relation is said to be Δ_n if it is both Σ_n and Π_n. Sometimes we use Σ_n (Π_n, Δ_n) to denote the set of all Σ_n (Π_n, Δ_n) sets; thus we write $\Delta_n = \Sigma_n \cap \Pi_n$, for example. As we have already seen, then, Σ_1, Π_1 and Δ_1 are the sets of r.e., co-r.e., and recursive sets, respectively.

There are two basic facts about the Σ-Π hierarchy that we shall prove in this section. The first is that every arithmetical set (i.e. every set definable in the language of arithmetic) belongs to this hierarchy (which is why it is called the "arithmetical hierarchy"); the second is that Σ_n and Π_n get more inclusive as n increases. Before doing this, we shall prove an enumeration theorem for this hierarchy.

Note that $S \in \Sigma_n$ iff $-S \in \Pi_n$, and $S \in \Pi_n$ iff $-S \in \Sigma_n$. To see this, suppose $S \in \Sigma_n$, and let A be a Σ_n formula that defines it. Then $\sim$A defines -S; but $\sim$A is equivalent to a Π_n formula, since we can push the negation sign through the initial string of quantifiers, changing universals to existentials and vice versa, and then through the bounded quantifiers. So -S is defined by a Π_n formula, i.e. $-S \in \Pi_n$. Similarly, we can show that if $S \in \Pi_n$ then $-S \in \Sigma_n$. It follows from this that $\Delta_n = \{S: S, -S \in \Sigma_n\} = \{S: S, -S \in \Pi_n\}$.

Note also that if S is Σ_n, then S is also Π_{n+1}: if A is a Σ_n formula defining S, and z is a variable not occurring in A, then (z)A is a Π_{n+1} formula which also defines S. ((z) is a vacuous quantifier here.) Similarly, if S is Σ_n then S is Σ_{n+1}: if A is a Σ_n formula that defines S, then let A' come from A by adding a vacuous quantifier onto the end of A's string of unbounded quantifiers; then A' is a Σ_{n+1} formula that defines S. Thus, $\Sigma_n \subseteq \Delta_{n+1}$, and by similar reasoning, $\Pi_n \subseteq \Delta_{n+1}$.

Suppose $\Sigma_n = \Sigma_{n+1}$ and $\Pi_n = \Pi_{n+1}$ for some n. Then as $\Sigma_n \subseteq \Pi_{n+1}$ and $\Pi_n \subseteq \Sigma_{n+1}$, it follows that $\Sigma_n \subseteq \Pi_n$ and $\Pi_n \subseteq \Sigma_n$, i.e. $\Sigma_n = \Pi_n$. Thus, if we can show that $\Sigma_n \neq \Pi_n$, it will follow that $\Sigma_n \subset \Sigma_{n+1}$ or $\Pi_n \subset \Pi_{n+1}$ (here we use $A \subset B$ to mean $A \subseteq B$ & $A \neq B$). In fact, both will follow: if $S \in \Sigma_{n+1} - \Sigma_n$, then $-S \in \Pi_{n+1} - \Pi_n$, so $\Sigma_n \subset \Sigma_{n+1}$ implies $\Pi_n \subset \Pi_{n+1}$, and by the same reasoning the converse holds. We know that $\Sigma_1 \neq \Pi_1$; we only have to show that $\Sigma_n \neq \Pi_n$ for $n > 1$.

Now let us prove the enumeration theorem we mentioned above.

Theorem: For all n, the Σ_n (Π_n) sets can be enumerated by a Σ_n (Π_n) relation.
Proof: Suppose A is a Σ_n formula and that n is odd, so that A's string of unbounded quantifiers ends in an $\exists$. Then A is $(\exists x_1)...(\exists x_n)R(x_1, ..., x_n, y)$ for some formula R of Lim. Consider the Σ_1 formula $(\exists x_n)R(x_1, ..., x_n, y)$. This formula is equivalent to $W(\mathbf{0}^{(e)}, x_1, ..., x_{n-1}, y)$ for some e, and the formula $W(e, x_1, ..., x_{n-1}, y)$ (where e is now a variable) is itself equivalent to $(\exists x_n)T(e, x_1, ..., x_n, y)$ for some formula T of Lim. It follows that A is equivalent to the Σ_n formula $(\exists x_n)...(\exists x_n)T(\mathbf{0}^{(e)}, x_1, ..., x_n, y)$. Since A was arbitrary, we see

that every Σ_n formula is equivalent to a Σ_n formula of the form $(\exists x_1)...(\exists x_n)T(\mathbf{0}^{(e)}, x_1, ..., x_n, y)$. Thus, the formula $(\exists x_1)...(\exists x_n)T(e, x_1, ..., x_n, y)$ (where e is now a variable) defines an enumeration of the Σ_n sets. Thus, for all odd n, there is a binary Σ_n relation that enumerates the Σ_n sets. The same proof shows that if n is even, then there is a binary Π_n relation that enumerates the Π_n sets. We can cover the remaining cases as follows. If n is even and R is a Π_n enumeration of the Π_n sets, then the relation -R is Σ_n, and moreover -R enumerates the Σ_n sets: if $S \in \Sigma_n$, then $-S \in \Pi_n$, so $-S = \{x: R(e, x)\}$ (for some e) and $S = -\{x: R(e, x)\} = \{x: -R(e, x)\}$; similarly, if n is odd and R is a Σ_n enumeration of the Σ_n sets, then -R is a Π_n enumeration of the Π_n sets. We can therefore conclude that for all n, there is a Σ_n relation that enumerates the Σ_n sets and a Π_n relation that enumerates the Π_n sets.

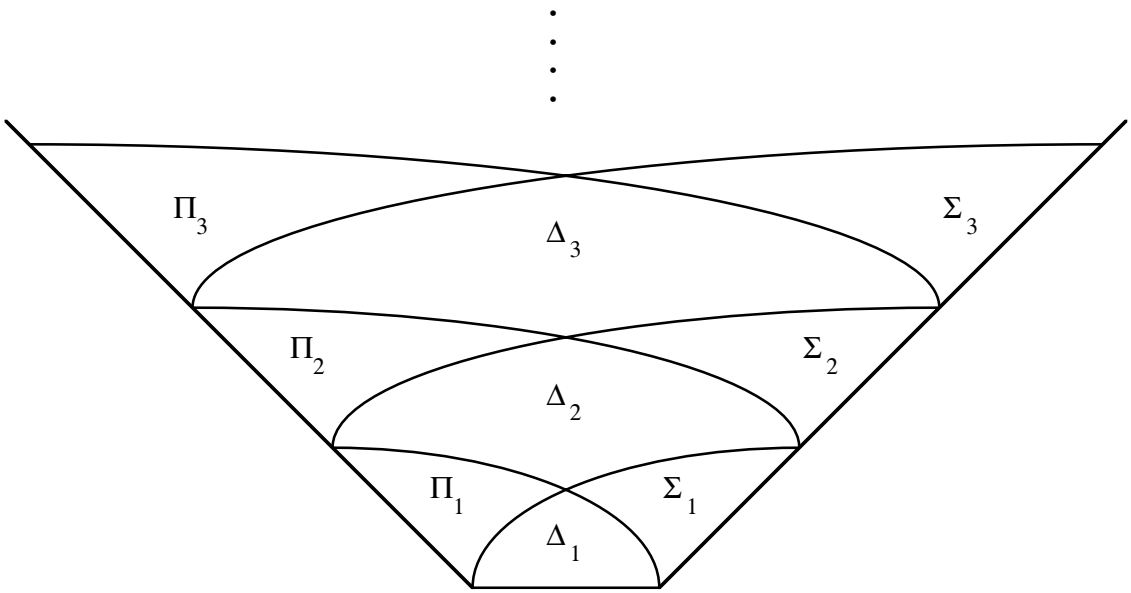
(There is also a Σ_n (Π_n) enumeration of the Σ_n (Π_n) k-place relations, for all k; we could either generalize the proof in the case k = 1, or use the pairing function.)

We are now ready to prove the desired

Hierarchy Theorem: $\Sigma_n \neq \Pi_n$ for all n.
Proof: Let n be given, and let $D = \{x: R(x, x)\}$, where R is an enumeration of Σ_n. $D \in \Sigma_n$, so $-D \in \Pi_n$. However, $-D \notin \Sigma_n$: if $-D \in \Sigma_n$, then $-D = \{x: R(e, x)\}$ for some e, so $e \in -D$ iff R(e, e) iff $e \in D$, contradiction.

Thus, the arithmetical hierarchy goes up without end. Note that this is a direct generalization of the proof that $\Sigma_1 \neq \Pi_1$, i.e. the proof that there is a nonrecursive r.e. set.

The arithmetical hierarchy gives us a way to classify the sets that occur in it. By the *level* of a set in the hierarchy, we mean the least inclusive of the various sets Σ_n, Π_n, and Δ_n of which it is an element. That is, if S is any set in the hierarchy and n is the least n such that $S \in \Sigma_n \cup \Pi_n$, then we call S properly Σ_n, properly Π_n, or Δ_n, as S is an element of $\Sigma_n - \Pi_n$, $\Pi_n - \Sigma_n$, or Δ_n.



The Arithmetical Hierarchy

Next, we prove a theorem which implies that every arithmetical set belongs to the hierarchy, and which allows us to make good estimates of the level of a given set.

Theorem: If a set or relation is definable in RE[P_1, ..., P_m], where P_1, ..., P_m are interpreted as Σ_n sets, then it is itself Σ_n.

Proof: We prove this by showing that every formula of RE[P_1, ..., P_m] is equivalent to some Σ_n formula. The proof is a double induction: we prove the theorem by an induction on n, and for each particular n, we prove that it holds for n by induction on the complexity of RE[P_1, ..., P_m] formulae.

Note that if the theorem holds for n, then a conjunction, disjunction, universal quantification, or bounded existential quantification of a Π_n formula is Π_n. To see this, suppose that A and B are Π_n. Then $\sim(A \wedge B)$ is equivalent to $\sim A \vee \sim B$, where $\sim A$ and $\sim B$ are Σ_n; so by the theorem, $\sim A \vee \sim B$, and hence $\sim(A \wedge B)$, is Σ_n, and therefore $A \wedge B$ is Π_n. Similarly, if A is Π_n, then $\sim(x)A$ is equivalent to $(\exists x)\sim A$, and $\sim A$ is Σ_n, so by the theorem $(\exists x)\sim A$ is Σ_n, so $(x)A$ is Π_n. The other cases are similar.

n = 1: A is a formula of RE[P_1, ..., P_m], where P_1, ..., P_m are interpreted as Σ_1 sets; so A is equivalent to the formula A' obtained from A by replacing each P_i by a Σ_1 formula that defines its extension. By the normal form theorem for RE, A', and hence also A, is equivalent to a Σ_n formula.

n > 1: we now prove the induction step by an induction on the complexity of RE[P_1, ..., P_m] formulae. Let A be such a formula.

A is atomic: then either A is an atomic formula of RE or a formula $P_i(x)$; in either case, A is equivalent to a Σ_n formula.

$A = B \wedge C$: then B and C are equivalent to Σ_n formulae $(\exists y_1)B'$ and $(\exists y_2)C'$, so A is equivalent to $(\exists y_1)(\exists y_2)(B' \wedge C')$. This in turn is equivalent to $(\exists y)(\exists y_1 < y)(\exists y_2 < y)(B' \wedge C')$. Now since B' and C' are Π_{n-1}, it follows (by the inductive hypothesis on n and the remarks at the beginning of the proof) that $(\exists y_1 < y)(\exists y_2 < y)(B' \wedge C')$ is also Π_{n-1}. That is, $(\exists y_1 < y)(\exists y_2 < y)(B' \wedge C')$ is equivalent to some Π_{n-1} formula D, so $(\exists y)(\exists y_1 < y)(\exists y_2 < y)(B' \wedge C')$, and hence A, is equivalent to the Σ_n formula $(\exists y)D$.

$A = B \vee C$: B and C are equivalent to Σ_n formulae $(\exists y)B'$ and $(\exists y)C'$, where B' and C' are Π_{n-1}; so A is equivalent to $(\exists y)(B' \vee C')$, and again $B' \vee C'$ is Π_{n-1}, so A is Σ_n.

$A = (\exists y)B$. Then B is equivalent to a Σ_n formula $(\exists z)B'$, so A is equivalent to $(\exists y)(\exists z)B'$, where B' is a Π_{n-1} formula; this in turn is equivalent to $(\exists w)(\exists y < w)(\exists z < w)B'$, and again $(\exists y < w)(\exists z < w)B'$ is Π_{n-1}, so the whole formula is Σ_n.

$A = (x < t)B$: then A is equivalent to $(x < t)(\exists y)B'$ for some Π_{n-1} formula B', which is in turn equivalent to $(\exists w)(x < t)(\exists y < w)B'$; again, $(x < t)(\exists y < w)B'$ is Π_{n-1}, so the whole formula is Σ_n.

Notice that the proof is really just an elaboration of the proof of the normal form theorem for RE.

We now have:

Theorem: All arithmetical sets and relations are Σ_n for some n.
Proof: We show, by induction on the complexity of formulae of the language of arithmetic, that those formulae define relations that are Σ_n for some n. Atomic formulae define Σ_1 sets. Suppose A defines a Σ_m relation and B defines a Σ_p relation. Letting $n = \max(m, p)$, A and B both define Σ_n relations. $A \wedge B$, $A \vee B$, and $(\exists y)A$ define Σ_n relations, as we have seen. $\sim A$ defines a Π_n relation, which is also a Σ_{n+1} relation. Finally, $(y)A$ defines a Π_{n+1} relation, which is also a Σ_{n+2} relation.

We could have proved this more quickly. We could, for example, have used Kleene's proof: to show that a formula A of the language of arithmetic is equivalent to a Σ_n formula, put A into prenex normal form, and then contract blocks of like quantifiers (i.e. all existential or all universal) into a single quantifier. (The contraction could use the pairing function, or it could imitate the above proof.) More quickly still, to obtain a Σ_n formula equivalent to A, put A into prenex normal form and then add enough vacuous quantifiers to make the unbounded quantifiers alternate.

The virtue of the above theorem is that it gives us a way of calculating a good estimate of the level of arithmetical sets and relations. If A is a formula of the language of arithmetic, first move all negation signs in (either all the way in, or far enough in that they only occur before formulae whose levels are known). The resulting formula will be built up via

conjunction, disjunction, and existential and universal quantification from formulae whose levels are known. We can then use the theorem to get an estimate of the level of the formula, using e.g. the facts that $(\exists x)B$ is Σ_n if B is and that if B and C are Σ_m and Σ_p, respectively, then $A \wedge B$ and $A \vee B$ are Σ_n, where $n = \max(m, p)$. Alternatively, if all of the unbounded quantifiers occur at the beginning of the formula, we can use the theorem to contract blocks of like quantifiers and read an estimate of the formula's level directly off the result.

Suppose the predicates $P_1, ..., P_m$ in the theorem all define Π_n sets. In that case, they define Σ_{n+1} sets, so every set or relation definable in $RE[P_1, ..., P_m]$ is Σ_{n+1}. Since being definable in $RE[P_1, ..., P_m]$ is the same as being enumeration reducible to $S_1, ..., S_m$, it follows that any set or relation enumeration reducible to a Π_n set is Σ_{n+1}. In fact, the converse is true: any Σ_{n+1} set is $\leq_e$ some Π_n set. To see this, let S be any Σ_{n+1} set, and let $(\exists z)A(x, z)$ be a Σ_{n+1} formula that defines it. Then $A(x, z)$ defines a Π_n relation R, so $S \leq_e R$ (since S is defined by the formula $(\exists z)P_1^2(x, z)$ of $RE[P_1^2]$), and therefore $S \leq_e \{[x, y]: R(x, y)\}$, which is easily seen to be Π_n. So a set or relation is Σ_{n+1} iff it is $\leq_e$ some Π_n set.

Thus, we begin to see a relation between the arithmetical hierarchy and the various reducibility notions. We shall examine this relation further, and prove a famous theorem of Post relating the arithmetical hierarchy to the jump hierarchy (i.e. the hierarchy 0, 0', 0",...).

Exercises

1. Calculate upper bounds as good as you can find for the levels in the arithmetical hierarchy of the following sets:

{e: W_e is infinite};
{e: W_e is recursive};
{e: W_e is nonempty};
{e: Φ_e is a total function}.

2. (a) In the class we defined a set S as *total* (with respect to enumeration reducibility) iff $-S \leq_e S$. (i) Prove that if S is any set, $S \cup -S$ is always total. (ii) Prove also that a set S consisting of ordered pairs [m,n] that codes the graph of a total function (not necessarily recursive) is total. (iii) Which r.e. sets are total? (iv) If S is any set, and S^+ is the set of pairs coding the graph of the characteristic function of S, prove that $S^+ \equiv_e S \cup -S$. (v) Prove the following normal form theorem, whenever the predicate P_1^1 is interpreted by a set S coding the graph of a total function: every enumeration operator when confined to such sets can be written in the form $(\exists s)(R(x,s) \wedge s \subseteq P_1^1 \wedge (j \leq s)(n \leq s)([j,n] \in s \supset (i<j)(\exists m \leq s)([i,m] \in s)))$ where R is an r.e. relation. (Given that S codes the graph of a total function, the clauses at the end mean that s codes a partial function whose domain is a finite initial segment of the

natural numbers.)

(b) An enumeration degree is called *total* if it contains at least one total set. (i) Prove that an enumeration degree is total iff it contains a set of pairs that codes the graph of some total function. (ii) Prove that the Turing degrees, as a partially ordered structure, are isomorphic to the total enumeration degrees. (iii) Prove that every enumeration degree contains a set coding the graph of a partial function. (For this reason, enumeration degrees are sometimes called partial degrees.) (iv) Give an example of a set that is not total but whose enumeration degree is nevertheless total. (v) Show that if S is any fixed-point of the function ϕ defined in exercise 5 of Lecture XVIII, then the enumeration degree of S is not total.

3. Here is yet another variation on the notion of a 1-complete set. A set S is said to be "weakly creative" iff S is r.e. and there is a *partial* recursive function ϕ such that whenever $W_x \cap S = \emptyset$, $\phi(x)$ is defined and $\phi(x) \notin S \cup W_x$. The difference between the notions "weakly creative" and creative is that here ϕ need not be total. (We can call ϕ a "weakly creative" function for S.) Actually, this definition was the original definition of "creative". Prove that all weakly creative sets are creative. (Hint: show that for every partial recursive function ϕ there is a total recursive χ such that $W_{\chi(x)} = W_x$ if $\phi(x)$ is defined, $W_{\chi(x)} = \emptyset$ otherwise. Define $\psi(x) = \phi(\chi(x))$. Show that ψ is total recursive and is a creative function for S if ϕ is a weakly creative function for S.)

This will complete our list of equivalent notions: weakly creative, creative, 1-1 creative, completely creative, 1-1 completely creative, many-one complete, 1-1 complete, and satisfies the effective form of Gödel's theorem. There are a few others, but we'll stop here.

Lecture XXI

The Arithmetical Hierarchy and the Jump Hierarchy.

Let us now look at some of the interrelations between the notions Σ_n and Π_n on the one hand, and the notions connected with relative recursiveness on the other. We proved that a set is Σ_{n+1} iff it is enumeration reducible to some Π_n set. If S is enumeration reducible to a Π_n set, then *a fortiori* it is r.e. in a Π_n set, or equivalently, in a Σ_n set. (S_1 is r.e. in S_2 iff S_1 is r.e. in $-S_2$, by the definition of "r.e. in".) Now suppose S_1 is r.e. in a Π_n set S_2. Then S_1 is definable in $RE[P_1, P_2]$, with P_1 and P_2 interpreted as S_2 and $-S_2$, respectively. Both S_2 and $-S_2$ are Σ_{n+1}, so S_1 is itself Σ_{n+1}. Thus we have the following result.

Theorem: A set S is Σ_{n+1} iff S is enumeration reducible to a Π_n set, iff S is r.e. in a Π_n set, iff S is r.e. in a Σ_n set.

Let us now relate this to the hierarchy 0, 0', 0", ... of degrees. We first prove the following

Lemma: For all n, a set is r.e. in $\emptyset^{(n)}$ iff it is Σ_{n+1}.
Proof: We prove this by induction on n. For n = 0, the theorem states that a set is Σ_1 iff it is r.e. in $\emptyset$. But a set is r.e. in $\emptyset$ iff it is r.e., so the theorem states that a set is r.e. iff it is Σ_1, which we already know to be the case.

Now let n > 0, and suppose the theorem holds for everything less than n.

$\Rightarrow$: Suppose S is r.e. in $\emptyset^{(n)}$. By the properties of the jump operator, $\emptyset^{(n)} = \emptyset^{(n-1)'}$ is r.e. in $\emptyset^{(n-1)}$. By the inductive hypothesis, then, $\emptyset^{(n)}$ is Σ_n. So S is r.e. in a Σ_n set and is therefore Σ_{n+1}.

$\Leftarrow$: Suppose S is Σ_{n+1}. Then S is r.e. in some Σ_n set S_1. By the inductive hypothesis, S_1 is r.e. in $\emptyset^{(n-1)}$. By the jump properties, $S_1 \leq_1 \emptyset^{(n-1)'} = \emptyset^{(n)}$, so *a fortiori* $S_1 \leq_T \emptyset^{(n)}$. By the weak transitivity property of *r.e. in*, S is r.e. in $\emptyset^{(n)}$.

If d is a Turing degree, we can say that a set S is *r.e. in d* iff S is r.e. in some set in d. If S is r.e. in a given set in d, then S is r.e. in *every* set in d: suppose S is r.e. in $S_1 \in$ d, and $S_2 \in$ d; then $S_1 \leq_T S_2$, so by the weak transitivity property, S is r.e. in S_2. By the same reasoning (this time using the transitivity of $\leq_T$), we can say that a set is *recursive in d* iff it is recursive in some, or equivalently every, set in d. Thus we can restate the above result as follows:

Corollary: A set is Σ_{n+1} iff it is r.e. in $0^{(n)}$.
Proof: $0^{(n)}$ is the degree of $\emptyset^{(n)}$.

We also have the following:

Corollary (Post's Theorem): A set is recursive in $0^{(n)}$ iff it is Δ_{n+1}.
Proof: S is recursive in $0^{(n)}$ iff both S and -S are r.e. in $0^{(n)}$, iff both S and -S are Σ_{n+1}, iff S is Δ_{n+1}.

This theorem was proved in a paper by Post in the early 40's; he also introduced the notions of simple and creative sets in that paper. The paper, by the way, was very important methodologically, as it was the first to rely heavily on the intuitive notion of a computation: previous work in recursion theory was all written out in a very formal way.

Post's theorem might be more of a surprise given other formalisms than our own (e.g. the Turing machine formalism), as it displays an intimate connection between the recursion-theoretic jump hierarchy on the one hand and on the other the arithmetical hierarchy, which was defined in terms of definability in a certain language.
* Given our own formalism this should be less surprising, since on our approach recursion-theoretic notions are themselves given in terms of definability, and Post's theorem simply shows that two different notions given in terms of definability match up in a certain way.

A set is said to be *1-complete Σ_n* (or simply *complete Σ_n*) if it is a Σ_n set to which all Σ_n sets are 1-1 reducible. Thus, a set is 1-complete Σ_1 just in case it is 1-complete. We could define *m-complete Σ_n* analogously, but it turns out that, just as in the special case n = 1, the two notions coincide.

Before going on, we should notice that every set many-one reducible to a Σ_n set is itself Σ_n. (So in particular, every set 1-1 reducible to a Σ_n set is Σ_n.) To see this, suppose $S_1 \leq_m S_2$ and S_2 is Σ_n. Then there is a recursive function ψ such that $S_1 = \{x: \psi(x) \in S_2\}$, so S_1 is defined by the formula $(\exists y)(PS(x, y) \wedge A(y))$, where A is a Σ_n formula that defines S and PS(x, y) is a Σ_1 formula that defines the graph of ψ. We can then calculate the whole formula to be Σ_n. (A(y) and PS(x, y) are both Σ_n, so their conjunction is, too; and adding an existential quantifier to a Σ_n formula just yields another Σ_n formula.) Therefore, the set S_1 is Σ_n.

It is immediate from this that any set many-one reducible to a Π_n set is itself Π_n. For suppose $S_1 \leq_m S_2$ and S_2 is Π_n; then $-S_1 \leq_m -S_2$ and $-S_2$ is Σ_n, so $-S_1$ is Σ_n, and so S_1 is Π_n. If S many-one reduces to a Δ_n set, then S many-one reduces to a set that is both Σ_n and Π_n, and is therefore itself both Σ_n and Π_n, i.e. it is Δ_n.

We can therefore show that A set S is complete Σ_n just in case for all S_1, S_1 is $\Sigma_n \Leftrightarrow S_1 \leq_1 S$. Clearly, if S_1 is $\Sigma_n \Leftrightarrow S_1 \leq_1 S$ for all S_1, then S is Σ_n (since $S \leq_1 S$) and every Σ_n set 1-1 reduces to S, i.e. S is complete Σ_n. If, on the other hand, S is complete Σ_n, then S_1 is $\Sigma_n \Rightarrow S_1 \leq_1 S$ for all S_1, so we only have to show that $S_1 \leq_1 S \Rightarrow S_1$ is Σ_n. But we know that S is Σ_n, so by the preceding remarks we know that any set 1-1 reducible to S is also Σ_n.

As a corollary to the lemma, we can deduce that each $\emptyset^{(n)}$ is complete Σ_n.

Theorem: For all $n > 0$, $\emptyset^{(n)}$ is complete Σ_n.
Proof: This is just to say that for all S, $S \leq_1 \emptyset^{(n)}$ iff S is Σ_n. But $S \leq_1 \emptyset^{(n)}$ iff S is r.e. in $\emptyset^{(n-1)}$, by the jump properties, iff S is Σ_n, by the lemma.

We therefore have several different characterizations of th
e Σ_{n+1} sets:

$$S \text{ is } \Sigma_{n+1} \Leftrightarrow S \leq_e \text{some } \Pi_n \text{ set} \Leftrightarrow S \text{ is r.e. in some } \Sigma_n \text{ set}$$
$$\Leftrightarrow S \text{ is r.e. in } \emptyset^{(n)} \Leftrightarrow S \leq_1 \emptyset^{(n+1)} \Leftrightarrow S \leq_m \emptyset^{(n+1)}.$$

(As for the last biconditional: if $S \leq_1 \emptyset^{(n+1)}$ then obviously $S \leq_m \emptyset^{(n+1)}$, and if $S \leq_m \emptyset^{(n+1)}$ then, since $\emptyset^{(n+1)}$ is Σ_{n+1}, S is also Σ_{n+1}.)

Trial-and-Error Predicates.

In the special case $n = 1$, Post's theorem implies that the Δ_2 sets are precisely the sets recursive in 0'. There are also other interesting characterizations of the Δ_2 sets.

One such characterization has to do with a modification of the notion of a computation. Consider a computing machine that gives tentative answers to questions that are put to it. When asked "is x in S?", it may answer "yes", but then later on change its mind and answer "no". In fact, it may change its mind several times; however, we require it to settle on a single answer after a finite number of changes of mind. If M is such a machine, the set computed by M is the set {x: M eventually settles on a "yes" answer for the input x}. Once this notion is made precise, it turns out that the sets computed by such machines are precisely the Δ_2 sets. (The notion of this kind of computation, and this result, are due to Hilary Putnam.)

One way to make this precise is as follows. Consider a total recursive function ψ in two variables which takes only the values 0 and 1. Suppose that for any m, there is an s_0 such that $\psi(m, s) = \psi(m, s_0)$ for all $s \geq s_0$. (s_0 need not be the same for all m.) ψ represents a machine of the sort we are considering, and $\psi(m, s)$ represents the sth answer given for the input m. (0 and 1 represent the answers "no" and "yes", respectively.) The set associated with the function ψ is the set $S = \{m: \psi(m, s_0) = 1$, where $\psi(m, s_0) = \psi(m, s)$ for all $s \geq s_0\}$. (Since s_0 depends on m, we can equivalently define $S = \{m: \psi(m, \rho(m)) = 1\}$, where ρ is any function such that for all m and for all $s \geq \rho(m)$, $\psi(m, s) = \psi(m, \rho(m))$. $\rho(m)$ need not be the least such s_0. ρ is called a *modulus of convergence* for ψ. S will always be recursive in any modulus of convergence for ψ.)

Let us call a set associated with such a ψ in the indicated way a *trial-and-error predicate*. It can be shown that the trial-and-error predicates are precisely the Δ_2 sets. The

proof that all trial-and-error predicates are Δ_2 is easy and is left as an exercise for the reader. The other direction is harder, and we shall sketch an informal proof. First, notice that any r.e. set is a trial-and-error predicate, for suppose S is r.e. and let P be any semi-computation for S. Then we can compute S (in the present sense of "compute") by setting P going and giving "no, no, no, ..." as output. If and when P says "yes", then we change our minds and start giving "yes, yes, ..." as output; if at any point P has not said anything, however, we continue to say "no". Thus, our outputs involve only a finite number of changes of mind (either one or none at all), so we have computed S in the appropriate sense. So all Σ_1 sets are trial-and-error predicates; the same applies to Π_1 sets by reversing "yes" and "no".

Now consider the general case. Suppose a set S is Δ_2; then it is recursive in some particular r.e. set (Ø', for example). So S is computed by some procedure P with an oracle to Ø'. Since Ø' is r.e., we have a trial-and-error machine for Ø'. For a given x, we can compute tentative answers to the question "is x in S?" as follows. Suppose we are being asked for the nth time. We run P for n steps, except that when P consults its oracle about whether $a \in$ Ø', we ask our trial-and-error machine whether $a \in$ Ø'. (If $n > 1$, then we may have asked it this question before.) If after n steps we have obtained an answer, we give that answer; otherwise we say "no" (or "yes"; it doesn't matter which). Now, when P is run with an oracle to Ø', the oracle is consulted only finitely many times before P halts with the correct answer to whether $x \in$ S, i.e. there is a finite collection $a_1, ..., a_k$ of sets such that when P is given correct answers to the questions "$a_1 \in$ Ø'?", ..., "$a_k \in$ Ø'?", and is given enough time to run, it will halt with the correct answer to the question "$x \in$ S?". So we will eventually reach a stage in our computation such that we have asked the trial and error machine the questions "$a_1 \in$ Ø'?", ..., "$a_k \in$ Ø'?" often enough to get correct answers, and such that we run P long enough to get an answer, which must be the correct answer, to whether $x \in$ S. So for any x, there is an n large enough that our computation always gives the correct answer to "$x \in$ S?" after stage n.

The Relativization Principle.

There is a general principle in recursion theory, which is hard to make precise but which ought to be stated nonetheless. It is that whenever we have a proof of some statement about the absolute notion of recursiveness or recursive enumerability, then we can demonstrate, using essentially the same proof, an analogous statement about the relative notion of recursiveness in a set or of recursive enumerability in a set. Or in general, any statement involving an absolute notion relativizes to the corresponding relative notion and by the same proof, provided the relative notion involves an oracle (or extra predicate, etc.) to both a set and its complement. This must be taken with a grain of salt, since if we have shown that some particular set is not recursive, or that it is not r.e., we do not thereby show that there is no set in which it is recursive or r.e. However, this is not the sort of statement that is

intended in the principle; once one has some experience with this principle, one gets a feel for what sort of statements are and are not allowed.

Consider, for example, the result that K is r.e. but not recursive. Let us define, for any given set S, W^S to be the relation $\{<e, m>: (\exists s)(s \subseteq S \cup -S \wedge W(e, x, s))\}$. W^S is thus an enumeration of the sets r.e. in S. Thus, for any e, W_e^S is the set $\{m: (\exists s)(s \subseteq S \cup -S \wedge W(e, x, s))\}$, and S' is simply the set $\{[e, m]: m \in W_e^S\}$. We can define K^S to be the set $\{e: e \in W_e^S\}$. K^S really has the same definition as K, except that we now relativize the relevant notions to S. The analog of the fact that K is r.e. but not recursive is the fact that K^S is r.e. in S but not recursive in S; this holds, and is shown by the very same proof we used to show that K is r.e. but not recursive.

As another example, we can relativize the Σ_n-Π_n hierarchy to a set S by considering formulae in the language of arithmetic plus an extra predicate interpreted as S. (We thereby get an atomic formula, namely ~P(x), which defines the complement of S, since the language of arithmetic has negation.) Thus, we have the relativized notions *Σ_n in S* and *Π_n in S*, with the obvious definitions. We similarly say that a set or relation is *arithmetical in S* if it is defined by some formula in the language of arithmetic with the extra predicate interpreted as S. We can prove, by the same proofs we used to prove the corresponding absolute theorems, that every set arithmetical in S is either Σ_n or Π_n in S for some n, that there is an enumeration of the sets Σ_n (or Π_n) in S which is itself Σ_n (Π_n) in S, and that there is always a set that is Π_n in S but not Σ_n in S. We also have a relativized version of Post's theorem, and by the same proof: if d is the degree of S, then a set is Δ_{n+1} in S iff it is recursive in $d^{(n)}$.

Now, people have tried to state the relativization principle formally, but every attempt so far has been unsuccessful. That is, every formal claim which has been put forth as a candidate statement of the principle has turned out to have counterexamples; however, these counterexamples are not intuitively counterexamples to the relativization principle itself.

The relativization principle does *not* hold for complexity theory. Whereas in recursion theory we do not place a time limit on a computation procedure, complexity theory is concerned with computations for which a time limit is given in advance. Corresponding to the question whether every r.e. set is recursive is the complexity-theoretic problem whether P = NP, which is unsolved to this day. Whatever the answer may be to this problem, however, we can be sure that it provides a counterexample to the relativization principle. We can relativize the P = NP problem by considering computations with an oracle to a given set; it turns out that there are some oracles for which P = NP and some for which $P \neq NP$. Obviously, if a relativization principle held in complexity theory, then we would have either P = NP for all oracles or $P \neq NP$ for all oracles.

A Refinement of the Gödel-Tarski Theorem.

We know from the work of Gödel and Tarski that the set of true sentences of the language of arithmetic is not itself definable in arithmetic. That is, for any formula A(x) of the language of arithmetic, there is a sentence B such that

(T) $$A(\mathbf{0}^{(m)}) \equiv B$$

does not hold, where m is the Gödel number for B (or in other words, B is a counterexample to (T)). For if there were no such B, then the above biconditional would hold for all B, and so A(x) would define the set of Gödel numbers of true statements. Our work on the arithmetical hierarchy allows us to get a refinement of this result. Specifically, if A is Σ_n (resp. Π_n), then we can choose B to be Π_n (resp. Σ_n).

To see this, suppose A(x) is Σ_n. Let B(x) be a Π_n formula that is not Σ_n; we know from our previous work that such a B(x) must exist. Now consider the function $\psi(m)$ = the Gödel number of $B(\mathbf{0}^{(m)})$. ψ is evidently recursive, so its graph is defined by some Σ_1 formula PS(x, y). Consider the formula $(\exists y)(PS(x, y) \wedge A(y))$. This formula is true iff $\psi(m)$ satisfies A(x), i.e. iff the Gödel number of the formula $B(\mathbf{0}^{(m)})$ satisfies A(x); if (T) has no Π_n counterexamples, then this holds iff $B(\mathbf{0}^{(m)})$ is true, iff m satisfies B(x). So in that case $(\exists y)(PS(x, y) \wedge A(y))$ is equivalent to B(x). Moreover, that formula is Σ_n, by our calculations. But then B(x) is equivalent to a Σ_n formula after all, which is impossible. So (T) has a Π_n counterexample, and by similar reasoning, reversing the roles of Σ and Π, if A(x) is Π_n then (T) has a Σ_n counterexample. (In that case, we use the formula $(y)(PS(x, y) \supset A(y))$ instead of $(\exists y)(PS(x, y) \wedge A(y))$.)

Looking more closely at this argument, we see that if m is a number such that $(\exists y)(PS(\mathbf{0}^{(m)}, y) \wedge A(y))$ and $B(\mathbf{0}^{(m)})$ have different truth values, then $B(\mathbf{0}^{(m)})$ is itself a Π_n counterexample to (T); otherwise $(\exists y)(PS(\mathbf{0}^{(m)}, y) \wedge A(y))$ is true iff $A(\mathbf{0}^{(q)})$ is true (where $q = \psi(m)$) iff $B(\mathbf{0}^{(m)})$ is true (since q is the Gödel number of $B(\mathbf{0}^{(m)})$). Moreover, since the only fact about B(x) we used was that it is a Π_n formula which is not Σ_n, we see that for any such formula B(x) and any Σ_n formula A(x), we can find a number m such that $B(\mathbf{0}^{(m)})$ is a counterexample to (T). However, this is not to say that we can find m *effectively* from B(x) and A(x); in fact, just as not all sets satisfy the effective form of Gödel's theorem, not all Π_n predicates B(x) are such that we can effectively find m from A(x).

It also turns out that this refinement of the Gödel-Tarski theorem is the best we can get, i.e. given a Σ_n formula A(x), there may not be a Σ_n counterexample to (T). In fact, for all $n \geq 1$, there is a Σ_n formula that defines truth for Σ_n sentences, and also a Π_n formula that defines truth for Π_n sentences. We prove this by induction on n.

First, we show that if there is a Σ_n formula that defines truth for Σ_n sentences, then there is a Π_n formula that defines truth for Π_n sentences. Suppose A(x) is such a Σ_n formula. Let ψ be a recursive function such that if m is the Gödel number of a Π_n sentence B, then

$\psi(m)$ is the Gödel number of a Σ_n sentence equivalent to ~B, and let PS(x, y) be a Σ_1 formula that defines the graph of ψ. Then a Π_n sentence B is true iff ~B is not true, so $(y)(PS(x, y) \supset \sim A(y))$ defines truth for Π_n sentences, and is equivalent to a Π_n formula.

Now we know already that there is a Σ_1 formula that defines truth for Σ_1 sentences, so the theorem holds for n = 1. Suppose it holds for n, and let A(x) be a Π_n formula that defines truth for Π_n sentences. Let χ be a recursive function such that if $(\exists x)C(x)$ is a Σ_{n+1} sentence with Gödel number m, then $\chi(m, p)$ is the Gödel number of $C(\mathbf{0}^{(p)})$, and let CH(x, y) be a Σ_1 formula that defines the graph of χ. $(\exists x)C(x)$ is true iff for some $C(\mathbf{0}^{(p)})$ is true for some p, so $(\exists y)(\exists z)(CH(x, z, y) \wedge A(y))$ defines truth for Σ_{n+1} sentences and is itself Σ_{n+1}. As we have already seen, it follows that there is a Π_{n+1} formula that defines truth for Π_{n+1} sentences, so we are done.

Lecture XXII

The ω-rule.

Recall that Gödel's theorem gives us a universally quantified statement (x)A(x) all of whose instances are provable but which is not itself provable. Thus, while intuitively it might seem like (x)A(x) follows from A(**0**), A(**0'**), ..., in fact, while all of the latter are provable, the former is not provable. However, it would be provable if we added to our formal system the following rule, known as the *ω-rule*: from A(**0**), A(**0'**), ... to infer (x)A(x). In fact, this was Hilbert's suggestion when he first heard about Gödel's result.

The ω-rule can't actually be applied in practice, since it has infinitely many premises and so a proof using the ω-rule would be infinitely long. Moreover, even if we can prove each of the instances of (x)A(x), we may not be in a position to know that they are all provable. For example, consider Goldbach's conjecture. Supposing that it is in fact true, we can easily prove each of its instances; nonetheless, we are not now in a position to know that all of its instances are provable, since we are not now in a position to prove that the statement itself is true.

Nonetheless, we can consider formal systems which contain the ω-rule, even if we cannot actually use such systems. If we add the ω-rule to an ordinary first-order deductive system (Q, for example), then not only will there be no true but unprovable Π_1 statements: all true statements will be provable. To see this, suppose we start out with a system which proves all true sentences of Lim, and which is such that every sentence of the language of arithmetic is provably equivalent to a Σ_n or Π_n sentence, for some n. If we add the ω-rule to such a system, then we will be able to prove every true Σ_n or Π_n sentence, and therefore every true sentence whatsoever. We show this by induction on n. (For the sake of the proof, we define a formula to be both Σ_0 and Π_0 if it is a formula of Lim.) We know it holds for n = 0, because by hypothesis all true sentences of Lim are provable. Suppose it holds for n, and let A be a Σ_{n+1} formula. Then A is $(\exists x)B(x)$ for some Π_n formula B(x). If A is true, then $B(\mathbf{0}^{(m)})$ is true for some m, so by the inductive hypothesis $B(\mathbf{0}^{(m)})$ is provable in the system, so A is also provable. Now let A be a Π_{n+1} formula. Then A is (x)B(x) for some Σ_n formula B(x). If A is true, then $B(\mathbf{0}^{(m)})$ is true for all m, so by the inductive hypothesis, $B(\mathbf{0}^{(m)})$ is provable for all m. Now we apply the ω-rule: from the sentences B(**0**), B(**0'**), ..., we can infer the sentence (x)B(x), i.e. the sentence A, so A is provable.

So as long as we stay within the first-order language of arithmetic, we can get around the Gödel theorem by allowing our formal systems to include the ω-rule. However, if we consider richer languages (e.g. languages with quantifiers over sets of numbers, or with extra predicates), we will not necessarily be able to get around the Gödel result in this way. In fact, there are languages richer than the first-order language of arithmetic such that, even

when we allow formal systems to contain an ω-rule, we get a Gödel-type result. This was first discovered by Rosser, but it was not until much later, when extensions of the arithmetical hierarchy were being studied in the 50's, that his ideas were taken up again.

The Analytical Hierarchy.

We have already seen how to enrich the language of arithmetic by adding extra predicates and function symbols. We can also treat these new symbols as *variables*, and even quantify over them. The resulting formulae will then have two types of variables: one type for numbers and one type for sets (or functions); if a formula has n number variables and k set variables, then it defines an n+k-place relation between numbers and sets, in which the first n places are occupied by numbers and the remaining places are occupied by sets. Similarly, if there are k function variables, then the formula defines an n+k-place relation between numbers and functions. (The formula f(x) = y, for example, defines the 3-place relation $\{<x, y, f>: x, y \in \mathbf{N}$ and $f: \mathbf{N} \to \mathbf{N}$ and $f(x) = y\}$.) When the variables are function variables, their values are always *total* functions.

We could get by with only unary predicates, reducing functions and other predicates to unary predicates via standard methods. We could also use only unary function symbols. That is, we could rewrite $f(x_1, ..., x_n)$ as $f([x_1, ..., x_n])$, and replace sets by their characteristic functions. In principle it doesn't matter what we do, but it will turn out to be convenient to require all the new variables to be unary function variables, so we shall do so. We use lower case Greek letters for function variables.

In the case of Σ^0_1 formulae, a version of the monotonicity and finiteness theorems hold. That is, if $A(x_1, ..., x_n, \alpha_1, ..., \alpha_k)$ is a Σ^0_1 formula, then $<m_1, ..., m_n, f_1, ..., f_k>$ satisfies it iff there are finite initial segments $s_1, ..., s_k$ of $f_1, ..., f_k$, such that $<m_1, ..., m_n, s_1, ..., s_k>$ satisfies it. (Unary functions on **N** can be seen as infinite sequences of numbers; an initial segment of a function f is then a sequence $<f(0), ..., f(x)>$ for some x.) Actually, this way of putting it isn't quite correct, because we require the values of the variables to be total functions, so we must restate it as follows. Let A* be the result of replacing $\alpha_i(x) = y$ by $(\exists z)(Seql(s_i, x) \wedge x < z \wedge [x, y] \in s_i)$ wherever it occurs in A. (If function variables are embedded in A, we iterate this process.) Then $<m_1, ..., m_n, f_1, ..., f_k>$ satisfies A iff for some finite initial segments $s_1, ..., s_k$ of $f_1, ..., f_k$, respectively, $<m_1, ..., m_n, \alpha_1, ..., \alpha_k>$ satisfies A*.

Now let us consider formulae which may contain quantifiers over functions; a relation between natural numbers and functions defined by such a formula is called *analytical*. In particular, a set of numbers defined by such a formula is called analytical.

A Σ^1_n formula is a formula that consists of an alternating string of function quantifiers of length n, beginning with an existential quantifier, followed by a single number quantifier of the opposite type from the last variable quantifier in the string, followed by a formula of Lim. The definition of "Π^1_n" is the same except that we require the first quantifier to be

universal. Thus, for example, the formula $(\alpha)(\exists\beta)(x)\ \alpha(x) = \beta(x)$ is a Π^1_2 formula. A relation is Σ^1_n or Π^1_n if it is defined by a Σ^1_n or Π^1_n formula, respectively; a relation is Δ^1_n if it is both Σ^1_n and Π^1_n. The hierarchy of Σ^1_n and Π^1_n sets is called the *analytical hierarchy*. This hierarchy was first studied by Kleene, who invented its name.

In general, a Σ^m_n or Π^m_n formula is an alternating string of type-m quantifiers of length n followed by a formula containing only quantifiers of type < m. Quantifiers over numbers are type-0, quantifiers over functions on **N**, sets of numbers, etc., are of type 1, quantifiers over sets of sets of numbers are of type 2, etc.

The analytical relations are not to be confused with the *analytic* relations, i.e. the Σ^1_1 relations. When Kleene first studied the analytical hierarchy, a certain class of functions had already been studied and were called "analytic"; it was only discovered later that these functions are precisely the Σ^1_1 functions. To avoid conflicting notations, the term "analytical" was chosen for the more inclusive class. Nowadays, in order to avoid confusion, the term "Σ^1_1" is generally used instead of "analytic".

Normal Form Theorems.

An arithmetical formula is a formula that does not contain any quantifiers over functions (though it may contain free function variables). We would like to show that every formula is equivalent to some Σ^1_n or Π^1_n formula (for some n), and in particular that every arithmetical formula is equivalent to some Σ^1_1 formula and to some Π^1_1 formula. At this point it should be far from obvious that this is the case, since a formula can have several number quantifiers, and a Σ^1_n or Π^1_n formula is only allowed to have a single number quantifier, and that of the opposite type from the last function quantifier. In this section we shall show how to find a Σ^1_n or Π^1_n equivalent for any formula of the language of arithmetic.

Clearly, any formula can be put into prenex form. (We consider a formula to be in prenex form if it consists of a string of unbounded quantifiers followed by a formula of Lim.) However, the initial string of quantifiers that results may not alternate, and it may also include number quantifiers. So to put the formula in the desired form, we must move the number quantifiers to the end of the string, collapse them to a single quantifier of the opposite type from the last function quantifier, and make the string of function quantifiers alternate.

First, let us work on moving the number quantifiers to the end. To do this, it suffices to show that any formula of the form $(Qx)(Q'\alpha)A$ is equivalent to a formula $(Q'\alpha)(Qx)A^*$, where Q and Q' are quantifiers and A differs from A* only in the part that is in Lim: if we have this result, then we can apply it repeatedly to any prenex formula to produce an equivalent prenex formula with all the number quantifiers at the end. This is easy to show when Q = Q': $(\exists x)(\exists\alpha)A$ is always equivalent to $(\exists\alpha)(\exists x)A$, and $(x)(\alpha)A$ is always equivalent to $(\alpha)(x)A$. So the only difficult case is when $Q \neq Q'$.

Consider a formula of the form $(x)(\exists\alpha)A$. This is true just in case for every number x there is a function α_x such that $A(x, \alpha_x)$ holds. Letting $\Phi(x) = \alpha_x$, this implies that there is a function Φ such that for all x, $A(x, \Phi(x))$ holds; conversely, if such an Φ exists, then obviously $(x)(\exists\alpha)A(x, \alpha)$ holds. Φ is a higher-order function, and the quantifiers in our formulae only range over functions from **N** to **N**, so we cannot rewrite $(x)(\exists\alpha)A$ as $(\exists\Phi)(x)A(x, \Phi(x))$. However, there is a way to get around this. Suppose Φ maps numbers onto functions; then let γ be the function from **N** to **N** such that $\gamma([x, y]) = (\Phi(x))(y)$. Let $A^*(x, \gamma)$ be the result of replacing all occurrences of $\alpha(t)$ in A by $\gamma([x, t])$, for any term t; clearly, A and A* differ only in the part that is in Lim. It is easy to see that $A^*(x, \gamma)$ holds iff $A(x, \Phi(x))$ holds. Therefore, $(x)(\exists\alpha)A$ holds iff there is a Φ such that for all x, $A(x, \Phi(x))$ holds, iff there is a γ such that for all x, $A^*(x, \gamma)$ holds, iff $(\gamma)(\exists x)A^*(x, \gamma)$ holds. So we have the desired result in this case.

There is only one remaining case, namely the case of formulae of the form $(\exists x)(\alpha)A(x, \alpha)$. But $(\exists x)(\alpha)A(x, \alpha)$ is equivalent to $\sim(x)(\exists\alpha)\sim A(x, \alpha)$, which, as we have just seen, is equivalent to $\sim(\exists\gamma)(x)\sim A^*(x, g)$, which is equivalent to $(\gamma)(\exists x)A^*(x, \gamma)$. So we have proved the following

Theorem: Any formula is equivalent to a prenex formula in which all the unbounded number quantifiers occur at the end.

Notice that, in moving from $(x)(\exists\alpha)A$ to $(\exists\Phi)(x)A(x, \Phi(x))$, we have assumed the axiom of choice: if the axiom of choice fails, then even though for every x there is an α such that $A(x, \alpha)$ holds, there may be no single function which takes x to an appropriate α.

The initial string of function quantifiers may not yet alternate. However, using the pairing function, we can collapse adjacent quantifiers of the same type into a single quantifier, and by repeating this process, we can make the initial string alternate. That is, for any formula $A(\alpha, \beta)$, let $A^*(\gamma)$ be a formula that differs from A only in the Lim part, and such that $A(\alpha, \beta)$ is equivalent to $A^*([\alpha, \beta])$ for all α, β. (Such an A* is easy to find.) Then $(\exists\alpha)(\exists\beta)A(\alpha, \beta)$ is equivalent to $(\exists\gamma)A^*(\gamma)$, and $(\alpha)(\beta)A(\alpha, \beta)$ is equivalent to $(\gamma)A^*(\gamma)$. (Here we are assuming that our pairing function is onto.) Thus, we have the following

Theorem: Any formula is equivalent to a prenex formula consisting of an alternating string of function quantifiers followed by a first-order formula.

To get the desired result, we must show how to collapse the number quantifiers into a single quantifier. We shall do this by proving that any first-order formula is equivalent to both a Σ^1_1 and a Π^1_1 formula. Once we have done this, we can prove our main result as follows. Let A be any formula, and take any prenex equivalent with all the function quantifiers in front. Suppose the last function quantifier is existential, and let B be the first-

order part of the formula. Then A is equivalent to a formula $(Q\alpha_1)...(\exists\alpha_n)B$. Now let $(\exists\alpha_{n+1})(x)C$ be a Σ^1_1 equivalent of B; A is equivalent to $(Q\alpha_1)...(\exists\alpha_n)\ (\exists\alpha_{n+1})(x)C$. We can collapse the adjacent quantifiers $(\exists\alpha_n)$ and $(\exists\alpha_{n+1})$; thus A is equivalent to $(Q\alpha_1)...(\exists\alpha_n)(x)D$, with D in Lim, i.e. A is equivalent to a Σ^1_n formula. If the last function quantifier is universal we argue similarly, this time using a Π^1_1 equivalent of B.

Theorem: Every first-order formula is equivalent to both a Σ^1_1 and a Π^1_1 formula.
Proof: Let A be any first-order formula. We know already that we can take A to be either Σ^0_n or Π^0_n, for some n. By adding vacuous quantifiers if necessary, we can assume that A is Π^0_n for some n and that n is even. Thus, A is equivalent to a formula $(x_1)(\exists y_1)...(x_m)(\exists y_m)B$ with B in Lim. Now any formula $(x)(\exists y)C(x, y)$ is equivalent to $(\exists\alpha)(x)C(x, \alpha(x))$, as we can see using the same sort of argument we used before. (If $(\exists\alpha)(x)C(x, \alpha(x))$ holds, then obviously $(x)(\exists y)C(x, y)$ holds; conversely, if $(x)(\exists y)C(x, y)$ holds, then $(\exists\alpha)(x)C(x, \alpha(x))$ holds, letting $\alpha(x)$ = the least y such that C(x, y) holds.) Iterating this, and moving the number quantifiers to the end, we see that A is equivalent to $(\exists\alpha_1)...(\exists\alpha_m)(x_1)...(x_m)B'$ for B' in Lim. We can collapse the existential function quantifiers, and we can also collapse the universal number quantifiers using a bounding trick. The result is Σ^1_1, so A is equivalent to a Σ^1_1 formula.

To see that A is also equivalent to a Π^1_1 formula, notice that the foregoing argument shows that the formula ~A is equivalent to some Σ^1_1 formula $(\exists\alpha)(x)B$, and so A itself is equivalent to the Π^1_1 formula $(\alpha)(\exists x)\sim B$.

By the foregoing remarks, we finally have our main result.

Theorem: Every formula is equivalent to some Π^1_n or Σ^1_n formula, for some n. Moreover, if A is a formula consisting of an alternating string of quantifiers of length n, the first quantifier of which is existential (universal), followed by a first order formula, then A is equivalent to a Σ^1_n (Π^1_n) formula.

(The trick of replacing $(x)(\exists y)C(x, y)$ by $(\exists\alpha)(x)C(x, \alpha(x))$ is due to Skolem. Notice that, in contrast to the previous case, we have not assumed the axiom of choice, since we defined $\alpha(x)$ to be the least y such that C(x, y). We were able to do this because we know that our domain (viz. **N**) can be well-ordered. Skolem's trick can be applied to any domain that can be well-ordered; however, if the axiom of choice fails, then there will be domains that cannot be well-ordered.)

As with the arithmetical hierarchy, we can define the *level* of an analytical relation to be the least inclusive Σ^1_n, Π^1_n, or Δ^1_n of which it is an element. The above discussion gives us ways of estimating the level of a given analytical relation.

All arithmetical relations are Δ^1_1, as we have seen. Moreover, if A is a Σ^1_n formula, then $(\exists\alpha)A$ is equivalent to a Σ^1_n formula since we can collapse $(\exists\alpha)$ with A's initial quantifier; similarly, if A is a Π^1_n formula, then $(\alpha)A$ is equivalent to a Π^1_n formula. In short, the Σ^1_n and

Π^1_n relations are closed under existential and universal functional quantification, respectively. Similarly, if A is Σ^1_n, then so are both (x)A and ($\exists$x)A, and the same is true if A is Π^1_n. This is because, as we have seen, we can always move number quantifiers inwards without affecting the variable quantifiers.

It is also not hard to see that if A and B are Σ^1_n (or Π^1_n), then so are A $\wedge$ B and A $\vee$ B. We can show this by induction on n. Since $\Sigma^1_0 = \Pi^1_0$ = {arithmetical relations}, this clearly holds for n = 0. Suppose it holds for n. If A and B are Σ^1_{n+1}, then they are ($\exists\alpha$)C and ($\exists\beta$)D for Π^1_n formula C and D. Then A $\wedge$ B and A $\vee$ B are equivalent to ($\exists\alpha$)($\exists\beta$)(C $\wedge$ D) and ($\exists\alpha$)($\exists\beta$)(C $\vee$ D), respectively, which are Σ^1_{n+1}, by the inductive hypothesis and collapsing the quantifiers ($\exists\alpha$) and ($\exists\beta$). If A and B are Π^1_{n+1}, we argue similarly.

Thus, the situation is similar to that of the arithmetical hierarchy, except that function quantifiers and unbounded number quantifiers play the role here that bounded and unbounded number quantifiers play in the arithmetical case. Using a similar argument to the one we gave there, we can see that if a relation is enumeration reducible to some Σ^1_n (resp. Π^1_n) relations, then it is Σ^1_n (resp. Π^1_n). It follows immediately that anything r.e. in a Δ^1_n relation is itself Δ^1_n; *a fortiori*, anything recursive in a Δ^1_n relation is Δ^1_n.

Exercise

1. Recall that A and B are *recursively isomorphic* (A $\equiv$ B) iff there is a 1-1 total recursive function ϕ whose range is **N**, and such that B = {ϕ(x): x $\in$ A}. Show that for all A and B, A $\equiv$ B iff A $\equiv_1$ B. The following sketches a method of proof. If A $\equiv$ B, then A $\equiv_1$ B follows easily, so suppose A $\equiv_1$ B. Let ϕ and ψ be 1-1 recursive functions such that x $\in$ A iff ϕ(x) $\in$ B and x $\in$ B iff ψ(x) $\in$ A, all x. Define, a sequence a_1, a_2, ... and a sequence b_1, b_2, ..., as follows. Suppose a_1, ..., a_n and b_1, ..., b_n have been defined (where possibly n = 0). If n is even, then let a_{n+1} be the least number distinct from a_1, ..., a_n, and let b_{n+1} be such that a_{n+1} $\in$ A iff b_{n+1} $\in$ B and b_{n+1} is distinct from all of b_1, ..., b_n. If n is odd, do the same thing in reverse (i.e. let b_{n+1} be the least number distinct from b_1, ..., b_n, etc.). Moreover, do this in such a way that the function χ such that $\chi(a_n) = b_n$ for all n $\in$ **N** is recursive. Conclude that χ is a 1-1 total recursive function whose range is **N**, and such that for all x, x $\in$ A iff χ(x) $\in$ B, and therefore that A $\equiv$ B. Hint: Informally, the problem reduces to finding an appropriate b_{n+1} *effectively* from a_1, ..., a_n, a_{n+1} and b_1, ..., b_n (or a_{n+1} from b_1, ..., b_{n+1} and a_1, ..., a_n, if n is odd). If $\phi(a_n) \notin \{b_1, ..., b_n\}$, then we can put $b_{n+1} = \phi(a_n)$. However, we may have $\phi(a_n) = b_i$ for some i = 1, ..., n; show how to get around this.

A *recursive isomorphism type* is a $\equiv$-equivalence class. Conclude that 1-degrees are therefore recursive isomorphism types, and that there is a 1-degree (which is also an m-degree and a recursive isomorphism type) which consists of the creative sets.

Comment: Dekker proposed that the notions studied by recursion theory should all be invariant under recursive isomorphism. While all the notions studied in this course are

invariant under recursive isomorphism, there is at least one notion, that of a *retraceable* set, which is not so invariant and which has been studied by recursion theorists. (Offhand, I don't know whether this notion was proved to be not recursively invariant before Dekker's proposal or only afterwards.)

Lecture XXIII

Relative Σ's and Π's.

The absolute notions Σ^0_n, Π^0_n, Σ^1_n, and Π^1_n can be relativized, just as we relativized the notions of recursiveness and recursive enumerability earlier. Let us say that a set is Σ^0_n *in* the unary functions $\alpha_1, ..., \alpha_n$ if it is definable by a Σ^0_n formula of the language of arithmetic with extra function symbols for the functions $\alpha_1, ..., \alpha_n$, and similarly for the notions Π^0_n, Σ^1_n, and Π^1_n in $\alpha_1, ..., \alpha_n$. So in particular, a relation between numbers, is Σ^0_1 in β (Δ^0_1 in β) just in case it is r.e. in β (recursive in β).

Another way of looking at this is as follows. Consider an arbitrary formula $A(x_1, ..., x_n, y_1, ..., y_m, \alpha_1, ..., \alpha_p, \beta_1, ..., \beta_q)$ of the language of arithmetic (possibly with function quantifiers), where the x's and y's are free number variables and the α's and β's are free function variables. The formula A defines an m+n+p+q-place relation, with m+n places for numbers and p+q places for functions. (Of course, any of m, n, p, and q may be 0.) Now suppose we regard the y's and β's as having *fixed* values (the numbers $k_1, ..., k_m$ and the functions $f_1, ..., f_q$, say). Relative to these fixed values, A defines an n+p-place relation. In the case of the fixed number values, we can get the same effect by considering the formula A* in which each variable y_i is replaced by the numeral $\mathbf{0}^{(k_i)}$; however, we cannot treat functions in the same way, since we do not have a term in the language for each function. (In fact, as long as we only have countably many terms in the language, we cannot have a term for each function, since there are uncountably many functions.) Ignoring the y's and k's, then, if the relation defined by A (with the β's treated as variables) is Σ^0_n, then the relation defined by A with the values of the β's fixed will be Σ^0_n in $f_1, ..., f_n$ (and similarly for Π^0_n, Σ^1_n, and Π^1_n).

Equivalently, an n+p-place relation R is Σ^0_n (or Π^0_n, etc.) in $\beta_1, ..., \beta_q$ iff there is an n+p+q-place Σ^0_n (or Π^0_n, etc.) relation R' such that $R = \{\langle x_1, ..., x_n, \alpha_1, ..., \alpha_p\rangle: \langle x_1, ..., x_n, \alpha_1, ..., \alpha_p, \beta_1, ..., \beta_q\rangle \in R'\}$. Thus, we can characterize the relative notions directly in terms of the corresponding absolute notions.

As with our other relative notions, we can reduce the general case to the case q = 1, this time using a pairing function on *functions*. There are several pairing functions that we could use. For example, we could take $[\beta_1, \beta_2]$ to be the function β such that $\beta(m) = [\beta_1(m), \beta_2(m)]$; alternatively, we could take it to be the function β such that $\beta(2n) = \beta_1(n)$ and $\beta(2n+1) = \beta_2(n)$ for all n. (The latter has the advantage of being an *onto* pairing function.) It is easy to verify that this successfully reduces the general case to the case of a single function.

We say that a relation R is $\mathbf{S}^0_n$ iff there is a function β such that R is Σ^0_n in β. We define $\mathbf{P}^0_n$, $\mathbf{S}^1_n$, and $\mathbf{P}^1_n$ similarly, and **D** is defined in the usual way. (So boldface letters are used for the notions with function parameters, lightface letters for the notions without function

parameters.) This notion is not very interesting if R is a relation between numbers, since in that case R will *always* be $\mathbf{D}^0_1$ (since R is always Δ^0_1 in its characteristic function). However, this is not the case if some of R's places are occupied by functions (i.e. if p > 0).

Let's look at $\mathbf{S}^0_1$ in the case n = 0 and p = 1 (i.e. the case of $\mathbf{S}^0_1$ sets of functions). A set S is $\mathbf{S}^0_1$ iff there is a 2-place Σ^0_1 relation R and a function β such that $S = \{\alpha: <\alpha, \beta> \in R\}$. We can also characterize the $\mathbf{S}^0_1$ sets topologically. *Baire space* is the topological space whose points are total functions from **N** to **N** and whose open sets are those sets S such that for every function ϕ in S, there is a finite initial segment s of ϕ such that every function extending s is also in S. To verify that this is indeed a topological space, we must show that if two sets satisfy our characterization of the open sets then their intersection does as well, and that if F is a family of sets satisfying that characterization then $\cup F$ also satisfies it. Alternatively, we can characterize Baire space as follows. For any finite sequence s, let $O_s = \{\phi: \phi$ is a total function which extends s$\}$; then the sets of the form O_s form a basis for Baire space.

Theorem: The $\mathbf{S}^0_1$ sets are precisely the open sets of Baire space.
Proof: First, suppose S is $\mathbf{S}^0_1$. Then there is a 2-place Σ^0_1 relation R between functions and a particular function β such that $S = \{\alpha: <\alpha, \beta> \in R\}$. Suppose $\alpha \in S$, i.e. $<\alpha, \beta> \in R$. By the monotonicity and finiteness properties of Σ^0_1 relations, there is an initial segment s of α such that $<\gamma, \beta> \in R$ for all γ extending s, and therefore $\gamma \in S$ for all such γ. Since α was arbitrary, it follows that S is open.

Next, suppose S is open. Let $F = \{s: \alpha \in S$ for all α extending s$\}$. Then $S = \{\alpha: \alpha$ extends s for some $s \in F\}$. Since F is a collection of finite sequences, we can let $G = \{n \in \mathbf{N}$: n codes some element of F$\}$, and let γ be G's characteristic function. Then S is Σ^0_1 in γ, and therefore $\mathbf{S}^0_1$, since we can define S by the Σ^0_1 formula $(\exists s)(\gamma(s) = \mathbf{0'} \wedge s \subseteq \alpha)$. (Here $s \subseteq \alpha$ abbreviates the formula $(n < s)(m < s)([n, m] \in s \pounds \alpha(n) = m)$.)

Baire space is also homeomorphic to the irrational numbers under the usual topology. The onto pairing function mentioned earlier is a homeomorphism between Baire space and its direct product with itself; since Baire space is homeomorphic to the irrationals, this shows that the irrational plane is homeomorphic to the irrational line. Thus, the situation is very different from the case of the reals.

We can set up a similar topology on sets of natural numbers by identifying these sets with their characteristic functions; if we restrict Baire space to functions into {0, 1}, the result is a space which is homeomorphic to the Cantor set. (That is, the set of all reals in the interval [0, 1] whose base-3 expansions contain no 1's.) It is also identical to the space $\mathbf{2}^\omega$, where **2** is the space {0, 1} with the discrete topology.

Notice that since the $\mathbf{S}^0_1$ sets are precisely the open sets, the $\mathbf{D}^0_1$ sets are precisely the clopen sets (i.e. sets that are both closed and open). This is another difference between the reals and the rationals: whereas the only clopen subsets of **R** are **R** itself and Ø, clopen sets

of irrationals exist in great abundance.

Another Normal Form Theorem.

Given a function α, let us define $\bar{\alpha}(n)$ to be some numerical code for the sequence $<\alpha(0), ..., \alpha(n-1)>$. It doesn't matter what particular code we choose; however, for definiteness, let us say that $\bar{\alpha}(n) = 2^{\alpha(0)+1} \cdot 3^{\alpha(1)+1} \cdot ... \cdot p_n^{\alpha(n-1)+1}$, where in general p_n is the nth prime. (This is essentially the coding scheme Gödel used.) As Quine has remarked, coding systems are not like matrimony, and we are free to switch back and forth between them as we please.

We now prove another normal form theorem, due to Kleene.

Theorem: If S is an n+p-place Σ_1^0 relation, then there is an n+p-place recursive relation R such that $S = \{<x_1, ..., x_n, \alpha_1, ..., \alpha_p>: (\exists z)R(x_1, ..., x_n, \bar{\alpha}_1(z), ..., \bar{\alpha}_n(z))\}$.
Proof: We shall prove the theorem for the case n = 0 and p = 1; the other cases are similar. Let S be a Σ_1^0 set of functions. For some relation $L(\alpha, y)$ definable in Lim, $S = \{\alpha: (\exists y)L(\alpha, y)\}$. By monotonicity and finiteness, $\alpha \in S$ iff some initial segment of α is in S, so $S = \{\alpha: (\exists z)(\exists y)L(\bar{\alpha}(z), y)\}$. In fact, $S = \{\alpha: (\exists z)(\exists y < z)L(\bar{\alpha}(z), y)\}$: if $(\exists z)(\exists y < z)L(\bar{\alpha}(z), y)$ then certainly $(\exists z)(\exists y)L(\bar{\alpha}(z), y)$, and if $L(\bar{\alpha}(k), y)$, then let $z > k, y$; $L(\bar{\alpha}(z), y)$ by monotonicity, so $(\exists z)(\exists y < z)L(\bar{\alpha}(z), y)$. Let $R'(z, s) \equiv (\exists y < z)L(s, z)$: R' is a recursive relation, and $S = \{\alpha: (\exists z)R'(z, \bar{\alpha}(z))\}$. This is almost what we want. Let $R(s) \equiv R'(lh(s), s)$, where lh(s) is the length of the sequence s; R is still recursive, and $S = \{\alpha: (\exists z)R(\bar{\alpha}(z))\}$.

This gives us a new normal form theorem for Π_1^1 relations.

Theorem: Every n+p-place Π_1^1 relation is $\{<x_1, ..., x_n, \alpha_1, ..., \alpha_p>: (\beta)(\exists z)R(x_1, ..., x_n, \bar{\alpha})_1(z), ..., \bar{\alpha}_p(z), \bar{\beta}(z))\}$ for some recursive relation R.
Proof: Let S be any n+p-place Π_1^1 relation. Then $S = \{<x_1, ..., x_n, \alpha_1, ..., \alpha_p>: (\beta)T(x_1, ..., x_n, \alpha_1, ..., \alpha_p, \beta)\}$ for some n+p+1-place Σ_1^0 relation T. By what we just proved, there is a recursive relation R such that $T(x_1, ..., x_n, \alpha_1, ..., \alpha_p, \beta)$ iff $(\exists z)R(x_1, ..., x_n, \bar{\alpha}_1(z), ..., \bar{\alpha}_p(z), \bar{\beta}(z))$; it follows that S is $\{<x_1, ..., x_n, \alpha_1, ..., \alpha_p>: (\beta)(\exists z)R(x_1, ..., x_n, \bar{\alpha}_1(z), ..., \bar{\alpha}_p(z), \bar{\beta})(z))\}$.

We can prove similar normal form theorems for the other Σ_n^1's and Π_n^1's. The main thing to note is that we have, so to speak, reduced the relation S, which may involve *functions*, to R, a recursive relation among *numbers*.

There is a related result about the various **S**'s and **P**'s.

Theorem: An n+p-place relation S is $\mathbf{S}^0_1$ iff for some β and some recursive R, $S = \{\langle x_1, ..., x_n, \alpha_1, ..., \alpha_p\rangle: (\exists z)R(x_1, ..., x_n, \overline{\alpha}_1(z), ..., \overline{\alpha}_p(z), \overline{\beta}(z))\}$, iff for some n+p-place relation R on **N** (not necessarily recursive), $S = \{\langle x_1, ..., x_n, \alpha_1, ..., \alpha_p\rangle: (\exists z)R(x_1, ..., x_n, \overline{\alpha}_1(z), ..., \overline{\alpha}_p(z))\}$.

Proof: The equivalence of the first two conditions is immediate. Suppose $S = \{\langle x_1, ..., x_n, \alpha_1, ..., \alpha_p\rangle: (\exists z)R(x_1, ..., x_n, \overline{\alpha}_1(z), ..., \overline{\alpha}_p(z), \overline{\alpha}(z))\}$, and let R' be the relation $\{\langle x_1, ..., x_n, \overline{\alpha}_1(z), ..., \overline{\alpha}_p(z)\rangle: z \in \mathbf{N}$ and $R(x_1, ..., x_n, \overline{\alpha}_1(z), ..., \overline{\alpha}_p(z), \overline{\beta}(z))\}$; then $S = \{\langle x_1, ..., x_n, \alpha_1, ..., \alpha_p\rangle: (\exists z)R'(x_1, ..., x_n, \overline{\alpha}_1(z), ..., \overline{\alpha}_p(z))\}$. Conversely, suppose $S = \{\langle x_1, ..., x_n, \alpha_1, ..., \alpha_p\rangle: (\exists z)R(x_1, ..., x_n, \overline{\alpha}_1(z), ..., \overline{\alpha}_p(z))\}$ for some relation R on **N**. Let β be the characteristic function of the set $\{[x_1, ..., x_n, y_1, ..., y_p]: R(x_1, ..., x_n, y_1, ..., y_p)\}$. Then $S = \{\langle x_1, ..., x_n, \alpha_1, ..., \alpha_p\rangle: (\exists z)\, \beta([x_1, ..., x_n, \overline{\alpha}_1(z), ..., \overline{\alpha}_p(z)]) = 1\}$, so S is Σ^0_1 in β and is therefore $\mathbf{S}^0_1$.

Similar results hold for the other **S**'s and **P**'s.

The Hyperarithmetical Hierarchy.

Consider the hierarchy 0, 0', 0", ..., $0^{(n)}$, ... of degrees. As we have seen, a set is arithmetical just in case it is recursive in one of these degrees. We also know that not all sets are arithmetical (e.g. the set of true sentences of the language of arithmetic), so there are sets which are not recursive in any of these degrees; therefore, there is a degree d which is not $\leq 0^{(n)}$ for any n. In fact, there are degrees d such that $0^{(n)} < d$ for all n: it is not too hard to see that the degree of the set of true sentences is such a degree. This suggests that we should be able to extend the hierarchy 0, 0', 0", ..., $0^{(n)}$, ... into the transfinite in some way.

In particular, it suggests that there ought to be a natural next degree, which we can call $0^{(\omega)}$, beyond all of the degrees $0^{(n)}$. But what is $0^{(\omega)}$? A natural answer would be that $0^{(\omega)}$ is the least upper bound of the degrees 0, 0', 0", ... However, by a result due to Spector, that collection of degrees does not have a least upper bound; so the most natural characterization of $0^{(\omega)}$ will not work.

However, the situation is not quite as bad as it first appears. While there is no least degree beyond 0, 0', 0", ..., there *is* a least degree a such that a = d" for some d > 0, 0', ... (This result is due to Enderton, Putnam and Sacks.) We can define $0^{(\omega)}$ to be the degree a. In fact, $0^{(\omega)}$ is the degree of the set of true sentences of the language of arithmetic.

We can use this idea to extend the hierarchy still further. In general, we say that a set is hyperarithmetical if it is recursive in $0^{(\alpha)}$ for some ordinal α for which $0^{(\alpha)}$ is defined. We can define the degrees $0^{(\omega+1)}$, $0^{(\omega+2)}$, etc. by $0^{(\omega+1)} = 0^{(\omega)'}$, $0^{(\omega+2)} = 0^{(\omega+1)'}$, etc.; in general, if $0^{(\alpha)}$ has been defined, we can define $0^{(\alpha+1)}$ to be $0^{(\alpha)'}$. We can define the next degree beyond all these, namely $0^{(\omega+\omega)}$, similarly to the way we defined $0^{(\omega)}$: there is a least degree a such that a = d" for some d > $0^{(\omega)}$, $0^{(\omega+1)}$, ..., and we can define $0^{(\omega+\omega)}$ to be that degree a. In fact, we can use this technique to define $0^{(\alpha)}$ for quite an extensive class of ordinals

(known as the *recursive ordinals*).

The resulting extended hierarchy is called the *hyperarithmetical hierarchy*. The hyperarithmetical hierarchy was first studied by Martin Davis in his Ph.D. thesis at Princeton. It was also invented independently by Mostowski and by Kleene (who coined the expression "hyperarithmetical"). Most of the basic theorems about the hierarchy were proved by Kleene and Spector.

Another approach is to define the *set* $\emptyset^{(\alpha)}$, for suitable a, and then let $0^{(\alpha)}$ be the degree of $\emptyset^{(\alpha)}$. On this approach, we can let $\emptyset^{(\omega)}$ be $\{[m, n]: m \in \emptyset^{(n)}\}$; the degree of $\emptyset^{(\omega)}$ is then $0^{(\omega)}$ as we defined it before. Obviously, this can be carried further into the transfinite. For example, we could let $\emptyset^{(\alpha+1)} = \emptyset^{(\alpha)'}$ whenever $\emptyset^{(\alpha)}$ is defined, and we could define $\emptyset^{(\omega+\omega)}$, for example, to be the set $\{[m, n]: m \in \emptyset^{(\omega+n)}\}$. To define $\emptyset^{(\omega^2)}$, we can do essentially the same thing, except this time it's a bit trickier: let $\emptyset^{(\omega^2)} = \{[m, n]: m \in \emptyset^{(\omega.n)}\}$. We could continue in this manner for quite some time, thinking of new definitions of $\emptyset^{(\alpha)}$ for limit ordinals α as we need them, but we would like to give a uniform definition of $\emptyset^{(\alpha)}$ for all of the appropriate α. We do so as follows.

An ordinal is said to be *recursive* if it is the order type of some recursive well-ordering of **N**. For example, ω is recursive because it is the order type of <0, 1, 2, ...>, and $\omega+\omega$ is recursive because it is the order type of <0, 2, 4, ..., 1, 3, 5, ...>. The recursive ordinals go up quite far. Of course, not every ordinal is recursive, since every recursive ordinal is countable but not every ordinal is countable. In fact, not all countable ordinals are recursive: since there are only countable many recursive well-orderings, there are only countably many recursive ordinals, but there are uncountably many countable ordinals. Once we have fixed a recursive well-ordering R, individual natural numbers code the ordinals less than the order type of R: specifically, we let $|n|_R$ denote the order type of the set $\{m: m\ R\ n\}$ ordered by R. (So $m\ R\ n$ iff $|m|_R < |n|_R$.)

Let S be an arbitrary recursive set, and let R be an arbitrary recursive well-ordering. We define H_n as follows, for all n. If $|n|_R = 0$, then $H_n = S$. If $|n|_R = \alpha+1$ and $|m|_R = \alpha$, then $H_n = (H_m)'$. Finally, if $|n|_R$ is a limit ordinal, let $H_n = \{[x, y]: x \in H_y \text{ and } x\ R\ y\}$. A set is said to be *hyperarithmetical* if it is recursive in H_n, for some n and some choice of R and S. (This definition is quite close to the definitions of Kleene and Spector.)

Now it might seem as though H_n depends strongly on the choice of S and of R. However, this is not really the case. Suppose R and R' are recursive well-orderings of the same order type, and S, S' are any two recursive sets; then whenever $|m|_R = |n|_{R'}$, H_m and H_n' are of the same Turing degree (where H_n' is H_n defined in terms of R' and S' rather than R and S). (The proof of this is due to Spector. The proof, by the way, is a nice illustration of the use of the recursion theorem in the study of recursive ordinals.) Thus, we may define $0^{(\alpha)}$ to be the degree of H_n, where $\alpha = |n|_R$, for any recursive ordinal α.

If R is allowed to be arithmetical, or even hyperarithmetical, then the order type of R is still a recursive ordinal; that is, while R may not itself be recursive, there is a recursive well-ordering R' which is isomorphic to R. Moreover, if R and R' are allowed to be arithmetical,

then H_m and H_n' are still of the same Turing degree, so the hierarchy is unaffected. If R is allowed to be hyperarithmetical, then the same sets get into the hierarchy, but the hierarchy may go up at a different rate.

The characterization of the hyperarithmetical sets that we have just given is *invariant*, in that while it involves S and R, which are extraneous to the hierarchy itself, the same hierarchy is given by any choice of S and R. A characterization in terms of double jumps (sketched at the beginning of this section), on the other hand, is *intrinsic* in the sense that such extraneous entities are not involved at all. This is certainly a virtue of the latter approach, although it relies on a rather more advanced result than the former approach, namely that for suitable sequences $a_1 < a_2 < ...$ of degrees there is a least d" such that $d > a_1, a_2, ...$

Another characterization of the hyperarithmetical sets is as follows. Consider those sequences $<S_\alpha:\ \alpha$ a recursive ordinal$>$ such that S_0 is recursive, $S_{\alpha+1} = S_\alpha'$ for all α, and when α is a limit ordinal, S_α is an upper bound of $\{S_\beta:\ \beta < \alpha\}$. (It doesn't matter which upper bound we choose.) Then a set will be hyperarithmetical just in case for *every* such sequence, it is recursive in some set in the sequence. There are many other equivalent characterizations of the hyperarithmetical sets.

Lecture XXIV

Hyperarithmetical and Δ^1_1 sets.

An important theorem about the hyperarithmetical sets, due to Kleene, is that they are all Δ^1_1. An even more important theorem, also due to Kleene (and whose proof is more difficult), is the converse. Thus, we have yet another characterization of the hyperarithmetical sets, this time in terms of the analytical hierarchy.

We shall prove the easier half of this theorem. In fact, we shall prove a somewhat stronger result. Let us say that a function ϕ is the *unique solution* of a formula $A(\alpha)$ if ϕ satisfies $A(\alpha)$ and is the only function that does so.

Theorem: If the characteristic function of a set S is the unique solution of an arithmetical formula, then S is Δ^1_1.
Proof: Let ϕ be the characteristic function of S, and let $A(\alpha)$ be an arithmetical formula of which ϕ is the unique solution. Then S is defined by the formula $(\exists\alpha)(A(\alpha) \wedge \alpha(x) = \mathbf{0}')$ and also by the formula $(\alpha)(A(\alpha) \supset \alpha(x) = \mathbf{0}')$. Since both $A(\alpha) \wedge \alpha(x) = \mathbf{0}'$ and $A(\alpha) \supset \alpha(x) = \mathbf{0}'$ are arithmetical formulae, the two formulae that define S are equivalent to Σ^1_1 and Π^1_1 formulae, respectively.

Notice that this argument goes through under the weaker assumption that $A(\alpha)$ is a Σ^1_1 formula.

Suppose S is hyperarithmetical; then there is a recursive well-ordering R of **N** such that S is recursive in H_n for some n, where $H_n = \emptyset$ when $|n|_R = 0$, $H_n = H_m'$ when $|n| = |m|_R+1$, and $H_n = \{[x, y]: x \in H_y \text{ and } y\ R\ n\}$ when $|n|_R$ is a limit ordinal. Let ψ be the characteristic function of $\{[m, n]: m \in H_n\}$. If ψ is the unique solution to some arithmetical formula, then that set is Δ^1_1. It follows easily (by the reasoning of the last section) that each H_n is Δ^1_1, so S is recursive in a Δ^1_1 set and is therefore itself Δ^1_1. Therefore, we need only find an arithmetical formula of which ψ is the unique solution.

Since R is r.e., there is an arithmetical formula B(x, y) that defines R, and let k be the R-least element of N. Define

$$\text{Zero}(n) =_{\text{df.}} n = \mathbf{0}^{(k)}$$
$$\text{Succ}(m, n) =_{\text{df.}} B(m, n) \wedge (y)\sim(B(m, y) \wedge B(y, n))$$
$$\text{Limit}(n) =_{\text{df.}} \sim\text{Zero}(n) \wedge \sim(\exists m)\text{Succ}(m, n).$$

These formulae hold just in case $|n|_R = 0$, $|n|_R = |m|_R+1$, and $|n|_R$ is a limit ordinal, respectively. Next, define

$$\begin{aligned}\text{Jump}(m, n, \alpha) =_{df.} (z)(\alpha([z, n]) = \mathbf{0'} \equiv \quad & (\exists e)(\exists m)(z = [e, m] \wedge (\exists s)[W(e, m, s) \wedge \\ & (k)((\mathbf{0}^{(2k)} \in s \supset \alpha([k, m]) = \mathbf{0'}) \wedge \\ & (\mathbf{0}^{(2k)}+\mathbf{0'} \in s \supset \alpha([k, m]) = \mathbf{0}))]))\end{aligned}$$

Let ϕ be a function into {0, 1} and let M and N be the sets $\{x: \phi([x, m]) = 1\}$ and $\{x: \phi([x, n]) = 1\}$, respectively. With a little work, we can see that Jump(m, n, ϕ) holds iff N = M'. We are now ready to define A(α): let A(α) be the formula

$$\begin{aligned}&(x)[(\alpha(x) = \mathbf{0} \vee \alpha(x) = \mathbf{0'}) \wedge (\alpha(x) = \mathbf{0'} \supset (\exists y)(\exists n)\, x = [y, n]) \wedge \\ &\quad (y)(n)(x = [y, n] \supset \\ &\qquad \{\text{Zero}(n) \supset \alpha(x) = \mathbf{0} \wedge \\ &\qquad \text{Limit}(n) \supset (\alpha(x) = \mathbf{0'} \equiv (\alpha(y) = \mathbf{0'} \wedge B(K_2(y), n))) \wedge \\ &\qquad (m)(\text{Succ}(m, n) \supset \text{Jump}(m, n, \alpha))\})].\end{aligned}$$

Now let us verify that ψ is the unique solution of A(α). First, we show that ψ satisfies A(α). ψ is a function into {0, 1} which only takes the value 1 on arguments that code pairs, so the first line of the formula is satisfied. Let x = [y, n] be given. If $|n|_R = 0$, then $H_n = \emptyset$, so $y \notin H_n$ and $\psi([y, n]) = 0$, so the third line is satisfied. If $|n|_R$ is a limit ordinal, then $H_n = \{[z, w]: z \in H_w \text{ and } w\ R\ n\} = \{u: \psi(u) = 1 \text{ and } K_2(u)\ R\ n\}$, so the fourth line holds. Finally, if $|n|_R = |m|_R+1$, then $H_n = H_m'$, so the last line holds as well.

Conversely, suppose ϕ satisfies A(α). Then Range(ϕ) $\subseteq$ {0, 1}, and $\phi(x) = 0$ when x is a nonpair. Let $G_n = \{y: \phi([y, n]) = 1\}$ for all n; we will show by transfinite induction on $|n|_R$ that $G_n = H_n$, from which it follows that $\phi = \psi$. If $|n|_R = 0$, then $\phi([y, n]) = 0$ for all y, so $G_n = \emptyset = H_n$. If $|n|_R = |m|_R+1$, then Jump(m, n, ϕ) holds and $G_n = G_m'$; by the inductive hypothesis, $G_m = H_m$, so $G_n = H_m' = H_n$. Finally, if $|n|_R$ is a limit ordinal, then $G_n = \{[z, w]: \phi([z, w]) = 1 \text{ and } |w|_R < |n|_R\} = \{[z, w]: z \in G_w \text{ and } |w|_R < |n|_R\}$ = (by the inductive hypothesis) $\{[z, w]: z \in H_w \text{ and } |w|_R < |n|_R\} = H_n$. This completes the proof.

The definition of A(α) is complicated, but the idea is simple. The sequence $<H_n: n \in \mathbf{N}>$ is defined in terms of itself; specifically, each H_n is defined in terms of various H_m for $|m|_R < |n|_R$. So we can define the function ψ in terms of itself in a similar way; if we do things right, the result will be an arithmetical formula A(α) with ψ as its unique solution.

$\mathbf{S}^1_n$ and $\mathbf{P}^1_n$ sets of functions are called *projective*, and the $\mathbf{S}^1_n$-$\mathbf{P}^1_n$ hierarchy is called the *projective hierarchy*. The study of the projective hierarchy and related notions is called *descriptive set theory*. Projective sets were studied years before Kleene studied the analytical hierarchy, and Suslin proved an analog of Kleene's result that Δ^1_1 = hyperarithmetical. (Specifically, he showed that the Borel sets are precisely the $\mathbf{D}^1_1$ sets.) A unified result, of which the results of Suslin and Kleene are special cases, is called the *Suslin-Kleene theorem*.

Kleene was originally unaware of this earlier work on projective sets. People then noticed analogies between this work and that of Kleene; later on, it was seen that not only

are the theories of projective sets and analytical sets analogous: in fact, they are really part of the same theory. Kleene originally called the analytical sets "analytic"; unfortunately, "analytic" was already a term of descriptive set theory for the $\mathbf{S}^1_1$ sets. To avoid confusion, Kleene's term was replaced by "analytical". Nowadays, to avoid confusion, most people say "$\mathbf{S}^1_1$" instead of "analytic".

Borel Sets.

The *Borel sets* are defined as follows: all open sets of Baire space are Borel; the complement of a Borel set is Borel; and if $<S_n: n \in \mathbf{N}>$ is any countable sequence of Borel sets, then $\cup_n S_n$ is also Borel. (It follows that $\cap_n S_n$ is Borel, since $\cap_n S_n = -\cup_n -S_n$.)

The Borel sets form a hierarchy, called the *Borel hierarchy*, defined as follows. The first level consists of the open sets and the closed sets, that is, the $\mathbf{S}^0_1$ sets and the $\mathbf{P}^0_1$ sets. The next level consists of countable unions of closed sets and countable intersections of open sets, or in other words, the $\mathbf{S}^0_2$ and $\mathbf{P}^0_2$ sets. (Countable unions of open sets are already open, and countable intersections of closed sets are already closed.) We can see that the $\mathbf{S}^0_2$ sets are precisely the countable unions of closed sets, as follows. We know already that the $\mathbf{P}^0_1$ sets are precisely the closed sets. On the one hand, suppose S is $\mathbf{S}^0_2$; then S is $\{\alpha: (\exists x)(y)R(x, y, \alpha, \beta)\}$ for some fixed β and some Π^0_1 relation R. For each n, let $S_n = \{\alpha: (y)R(n, y, \alpha, \beta)\}$; then $S = \cup_n S_n$, and each S_n is $\mathbf{P}^0_1$ and therefore closed, so S is a countable union of closed sets. Conversely, suppose $S = \cup_n S_n$, where each S_n is closed and therefore $\mathbf{P}^0_1$. For each n, $S_n = \{\alpha: (y)\, \bar{\alpha}(y) \in X_n\}$ for some set X_n of numbers, by our normal form theorem for $\mathbf{P}^0_1$. Let R be the relation $\{<x, n>: x \in X_n\}$; then $\alpha \in \cup_n S_n$ iff $(\exists n)(y)R(\bar{\alpha}(y), n)$, so $\cup_n S_n$ is $\mathbf{S}^0_2$. So the $\mathbf{S}^0_2$ sets are precisely the countable unions of closed sets, from which it follows that the $\mathbf{P}^0_2$ sets are precisely the countable intersections of open sets. In general, the $\mathbf{S}^0_n$ sets are the countable unions of $\mathbf{P}^0_{n-1}$ sets and the $\mathbf{P}^0_n$ sets are the countable intersections of $\mathbf{S}^0_{n-1}$ sets, by the same argument.

The various $\mathbf{S}^0_n$'s and $\mathbf{P}^0_n$'s do not exhaust the Borel hierarchy: we can find a countable collection of sets which contains sets from arbitrarily high finite levels of the hierarchy, and whose union does not occur in any of these finite levels. We therefore need another level beyond these finite levels. Let us call a set $\mathbf{S}^0_\omega$ if it is a countable union of sets, each of which is $\mathbf{S}^0_n$ for some n, and $\mathbf{P}^0_\omega$ if it is a countable intersection of sets, each of which is $\mathbf{P}^0_n$ for some n. In general, for countable infinite ordinals α we define a set to be $\mathbf{S}^0_\alpha$ if it is the union of a countable collection of sets, each of which is $\mathbf{P}^0_\beta$ for some $\beta < \alpha$, and $\mathbf{P}^0_\alpha$ if it is the intersection of a countable collection of sets, each of which is $\mathbf{S}^0_\beta$ for some $\beta < \alpha$. It turns out that new Borel sets appear at each level of this hierarchy. On the other hand, it is easy to see that every Borel set appears eventually in the hierarchy. For suppose not: then there is some countable family F of sets in the hierarchy such that $\cup$F is not in the hierarchy. For each $S \in$ F, let rank(S) = the least ordinal α such that $S \in \mathbf{P}^0_\alpha$. Then

{rank(S): $S \in F$} is a countable collection of countable ordinals, and it therefore has a countable upper bound α. But then $\cup S \in \mathbf{S}^0_\alpha$.

Notice that there are two equivalent ways to characterize at least the finite levels of the Borel hierarchy. One is purely topological: the $\mathbf{S}^0_1$ sets are the open sets, the $\mathbf{P}^0_1$ sets are the closed sets, the $\mathbf{S}^0_2$ sets are countable unions of closed sets, the $\mathbf{P}^0_2$ sets are countable intersections of open sets, etc. This is the way the Borel hierarchy was originally conceived, before analogies with recursion theory were noticed. The other is in terms of definability: a set is $\mathbf{S}^0_1$ iff it is definable by a $\mathbf{S}^0_1$ formula with a single function parameter, etc. The **S**, **P** notation was borrowed from recursion theory; the original notation (still quite standard outside of logic) was more baroque. Countable unions of closed sets were called F_σ, countable intersections of open sets were called G_δ, countable unions of G_δ's were called $G_{\delta\sigma}$, etc.

It is fairly easy to show that all Borel sets are $\mathbf{D}^1_1$. To prove this, it suffices to show that all open sets are $\mathbf{D}^1_1$, and that $\mathbf{D}^1_1$ is closed under complements and countable unions. That $\mathbf{D}^1_1$ is closed under complements is immediate from its definition. Suppose S is an open set; then S is $\{\alpha: R(\alpha, \beta)\}$ for some fixed β and some Σ^0_1 relation R; we know already that any Σ^0_1 relation is Δ^1_1, so S is $\mathbf{D}^1_1$. Finally, suppose $\{S_n: n \in \mathbf{N}\}$ is a countable family of $\mathbf{D}^1_1$ sets. In particular, each S_n is $\mathbf{P}^1_1$. Each S_n is $\{\alpha: (\beta)(\exists x)R_n(\overline{\alpha}(x), \overline{\beta}(x))\}$ for some relation R_n on **N**. Let R be the relation $\{<y, z, n>: R_n(y, z)\}$; $\cup_n S_n = \{\alpha: (\exists n)(\beta)(\exists x)R(\overline{\alpha}(x), \overline{\beta}(x), n))\}$. But we know already that the $\mathbf{P}^1_1$ relations are closed under number quantification, so $\cup_n S_n$ is $\mathbf{P}^1_1$. The proof that $\cup_n S_n$ is $\mathbf{S}^1_1$ is similar.

Borel sets are analogous in a number of ways to the hyperarithmetical sets. In particular, we can imitate the Borel hierarchy in the case of sets of numbers. It would not do to have the family of sets be closed under countable unions, since then as long as every singleton is included, every set whatsoever will be included. However, if we replace unions with recursive unions, we can get around this difficulty. Specifically, we can set up a system of notations for sets of numbers as follows. Let [0, m] code the set {m}; if n codes a set S, let [1, n] code the set -S; finally, if every element of W_e is already the code of some set, let [2, e] code the set $\cup\{S: S \text{ is coded by some element of } W_e\}$. We might call the sets that receive codes under this scheme the *effective Borel sets*, and the hierarchy that they form the *effective Borel hierarchy*. It turns out that the effective Borel sets are precisely the hyperarithmetical sets.

Π^1_1 Sets and Gödel's Theorem.

It turns out that there are close analogies between the Π^1_1 sets and the recursively enumerable sets (and also between the Δ^1_1 sets and the recursive sets). For example, consider the following extension of the notion of a computation procedure. We can consider, if only as a mathematical abstraction, machines which are capable of performing

infinitely many operations in a finite amount of time. (For example, such a machine might take one second to perform the first operation, half a second to perform the second one, and so on.) Such a machine will always be able to decide a Π^0_1 set, for such a set is of the form $\{x: (y)R(x, y)\}$ for some recursive relation R, and so the machine can run through all the y's, checking in each case whether R(x, y) holds, and then concluding that x is in the set or that it isn't. Using similar reasoning, we can see that any arithmetical set can be decided by such a machine. In fact, if the notion is made precise, it will turn out that the Δ^1_1 sets are precisely those sets that can be decided by such a machine, and that the Π^1_1 sets are those that can be semi-computed by one.

Another way in which the Π^1_1 sets are analogous to r.e. sets concerns representability in formal systems. Specifically, if we consider formal systems with the ω-rule, then it will turn out that all the sets weakly representable in such systems are Π^1_1, and conversely that any Π^1_1 is weakly representable in such a system.

We could also characterize the Π^1_1 sets via definability in a language: analogously to the language RE, we could set up a language with conjunction, disjunction, unbounded number quantifiers, and universal function quantifiers, in which precisely the Π^1_1 sets would be definable.

As in the arithmetical hierarchy, we have the following theorem.

Enumeration Theorem: For all n > 0 and all n and p, there is an m+1+p-place Π^1_n relation that enumerates the m+p-place Π^1_n relations, and similarly for Σ^1_n.

Proof: In what follows, we use $\vec{x}$ to abbreviate $x_1, ..., x_m$, and $\vec{\beta}$ to abbreviate $\beta_1, ..., \beta_p$. Let S be any m+p-place Π^1_1 relation. S is $\{<\vec{x}, \vec{\beta}>: (\alpha)(\exists z)R(\overline{\alpha}(z), \vec{x}, \overline{\beta}_1(z), ..., \overline{\beta}_p(z))\}$ for some recursive relation R. Since R is r.e., $R = W_e$ for some e, $S = \{<\vec{x}, \vec{\beta}>: (\alpha)(\exists z)W(e, \overline{\alpha}(z), \vec{x}, \overline{\beta}_1(z), ..., \overline{\beta}_p(z))\}$. So the relation $\{<e, \vec{x}, \vec{\beta}>: (\alpha)(\exists z)W(e, \overline{\alpha}(z), \vec{x}, \overline{\beta}_1(z), ..., \overline{\beta}_p(z))\}$ enumerates the m+p-place Π^1_1 relations. Moreover, that relation is itself Π^1_1, since it comes from an arithmetical relation by universal function quantification.

Just as we derived the general enumeration theorem for the arithmetical hierarchy from the special case of Σ^0_1, we can derive the present theorem from the case of Π^1_1. For example, consider the case of m+p-place Π^1_n relations with n odd. Any such relation is $\{<\vec{x}, \vec{\beta}>: (\alpha_1)(\exists\alpha_2)...(\exists\alpha_{n-1})S(\vec{x}, \vec{\beta}, \vec{\alpha})\}$ for some Π^1_1 relation S (where naturally $\vec{\alpha}$ abbreviates $\alpha_1, ..., \alpha_{n-1}$). But then by the enumeration theorem for Π^1_1 relations, this is $\{<\vec{x}, \vec{\beta}>: (\alpha_1)(\exists\alpha_2)...(\exists\alpha_{n-1})R(e, \vec{x}, \vec{\beta}, \vec{\alpha})\}$ for some e, where R is a Π^1_1 enumeration of the m+p+(n-1)-place $\Pi\backslash O(^1_1)$ relations. So the relation $\{<e, \backslash O(x, \vec{\ }), \backslash O(\beta, \vec{\ })>: (\alpha_1)(\exists\alpha_2)...(\exists\alpha_{n-1})R(e, \vec{x}, \vec{\beta}, \vec{\alpha})\}$ is a Π^1_n enumeration of the m+p-place Π^1_n relations. The other three cases are treated similarly.

(A similar theorem, called the *parameterization theorem*, holds for $\mathbf{S}^1_n$ and $\mathbf{P}^1_n$ relations;

in that case, relations have functions rather than numbers as indices.)

We can use the enumeration theorem to prove the following.

Hierarchy Theorem: For all n, $\Sigma^1_n \neq \Pi^1_n$.
Proof: Let R be a Π^1_n enumeration of the Π^1_n sets of numbers, and let D = {x: R(x, x)}. Then D is clearly Π^1_n, so -D is Σ^1_n. But -D is not Π^1_n, for if it were, we would have -D = {x: R(e, x)} for some e, and so $e \in$ -D iff R(e, e) iff $e \in$ D. So -D $\in \Sigma^1_n - \Pi^1_n$.

So in particular, there is a set $D \in \Sigma^1_1 - \Pi^1_1$. This D is analogous to K; we may as well call it K^{Π}.

Most of our earlier discussion of Gödel's theorem can be duplicated in the present case. (Of course, if a system has the ω-rule, or in general has Π^1_1 inference rules, it may decide every arithmetical statement. However, this is not to say that it decides every second-order statement.) Just as we showed that any system with an r.e. set of axioms and r.e. rules has an r.e. set of theorems, we want to show that the set of theorems generated by a finite set of Π^1_1 rules is Π^1_1.

First, let us associate with each rule of inference with the relation {<x, α>: x follows by the rule from premises in the set with characteristic function α}, and say that a rule is Π^1_1 if the corresponding relation is. Thus, the ω-rule is to be identified with the relation {<(x)A(x), α>: α is the characteristic function of some set that contains $A(\mathbf{0}^{(n)})$ for all n}. Let χ be a recursive function such that for all formulae A(x), if m is the Gödel number of (x)A(x), then χ(m, n) = the Gödel number of $A(\mathbf{0}^{(n)})$; then the ω-rule is Π^1_1, since the corresponding relation is defined by the formula $(y)\ \alpha(y) \leq \mathbf{0}' \wedge (n)\ \alpha(\chi(x, n)) = \mathbf{0}'$. If S is a set of sentences, then we can get the effect of taking all of the sentences in S as axioms by having the single rule *from any set of premises to infer any sentence in S*. This rule corresponds to the relation defined by $(y)\ \alpha(y) \leq \mathbf{0}' \wedge x \in S$, which is Π^1_1 if S is. Finally, if $R_1, ..., R_n$ are P^1_1 rules, then the relation $R = \{<x, \alpha>: R_1(x, \alpha) \vee ... \vee R_n(x, \alpha)\}$ is a Π^1_1 relation, and a sentence is a theorem of the formal system consisting of the rules $R_1, ..., R_n$ just in case it is a theorem of the single rule R. Thus, if we can show that the set of theorems of a single Π^1_1 rule is itself Π^1_1, it will follow that the set of theorems of a system with a Π^1_1 set of axioms, a finite number of Π^1_1 rules, and the ω-rule is Π^1_1.

Given a rule R, let ψ be the following operator on sets: ψ(S) = {x: R(x, S's characteristic function)}. Let ϕ be the corresponding operator on functions: if α is the characteristic function of a set S, then ϕ(α) = the characteristic function of ψ(S) = the characteristic function of {x: R(x, α)}. If R is a rule of inference in any reasonable sense, then ψ will be monotonic, since ψ(S) = the set of sentences that follow via R from sentences in S: if $S \subseteq S'$ and A follows from some sentences in S, then A also obviously follows from some sentences in S' as well. The set of theorems of R is the least fixed point of ψ. Recall that the least fixed point of ψ is the set $\cap\{S: \psi(S) \subseteq S\} = \{x: (S)(\psi(S) \subseteq S \supset x \in S)\}$. In terms of the operator ϕ, this set is $\{x: (\alpha)((y)[(\phi(\alpha))(y) = 1 \supset \alpha(y) = 1] \wedge \alpha$ is a

characteristic function $\supset \alpha(x) = 1)\}$. Since $(\phi(\alpha))(y) = 1$ iff $R(x, \alpha)$, this set is defined by the formula $(\alpha)([(y)(R(x, \alpha) \supset \alpha(y) = \mathbf{0}') \wedge (y)\ \alpha(y) \leq \mathbf{0}'] \supset \alpha(x) = \mathbf{0}')$. We must check that this formula is indeed Π^1_1. Since R is Π^1_1 and $R(x, \alpha)$ occurs in the antecedent of a conditional, the formula $(y)(R(x, \alpha) \supset \alpha(y) = \mathbf{0}')$ is Σ^1_1. However, that formula itself occurs in the antecedent of a conditional, so the formula $[(y)(R(x, \alpha) \supset \alpha(y) = \mathbf{0}') \wedge (y)\ \alpha(y) \leq \mathbf{0}'] \supset \alpha(x) = \mathbf{0}'$ is Π^1_1. Finally, when (α) is added, the formula remains Π^1_1. We therefore have the following

Theorem: If a formal system has Π^1_1 set of axioms and a finite number of Π^1_1 rules (possibly including the ω-rule), then the set of theorems of the system is itself Π^1_1.

The definition of "weakly represents" for such formal systems is the same as for ordinary formal systems. Let S be a set of numbers which is weakly representable in some such system. Then $S = \{n: A(\mathbf{0}^{(n)})$ is a theorem$\}$ for some formula A(x). Let χ be a recursive function such that $\chi(n)$ = the Gödel number of $A(\mathbf{0}^{(n)})$; then χ reduces S 1-1 to the set of theorems of the system, and so S is Π^1_1. So any set weakly representable in such a system is Π^1_1.

Conversely, we can find formal systems in the second-order language of arithmetic which weakly represent all the Π^1_1 sets, just as all the r.e. sets are weakly representable in Q. In particular, if Γ is such a system, then the set of theorems of the system, being Π^1_1, is weakly representable in the system itself. We can use this fact to construct a sentence that says '"Gödel heterological" is Gödel heterological', and prove that the sentence is true but unprovable if the system is consistent.

If Γ is a system all of whose theorems are true, then we can show directly that Γ is incomplete, by showing that the set of theorems of Γ is not the set of true sentences. For if it were, then the set of true sentences of the language would be Π^1_1 and therefore definable in the language itself. But then by the usual argument satisfaction would also be definable, which is impossible because the language has negation.

If Γ is Π^1_1-complete (i.e. if every true Π^1_1 sentence is provable) and consistent, then we can get a closer analog of Gödel's theorem. Let S be any Π^1_1 set of numbers that is not Σ^1_1; K^{Π} would do, for example. Then there is a Σ^1_1 formula A(x) that defines -S. Just as we did in the original Gödel theorem, we can prove that there are statements of the form $A(\mathbf{0}^{(n)})$ that are true but unprovable in the system.

Arithmetical Truth is Δ^1_1.

We have proved that all hyperarithmetical sets are Δ^1_1; since we know already that not all hyperarithmetical sets are arithmetical, it follows that there are Δ^1_1 sets that are not arithmetical. There is also a direct proof of this, due to Tarski. We know that the set of true

arithmetical sentences is not arithmetical; we can use Tarski's famous definition of truth to show that this set is Δ^1_1.

We showed that for a set to be Δ^1_1 it is sufficient that its characteristic function be the unique solution of some arithmetical formula (or even Σ^1_1 formula) $A(\alpha)$. Recall the usual inductive definition of truth:

$\mathbf{0}^{(m)} = \mathbf{0}^{(n)}$ is true iff m = n;
$A(\mathbf{0}^{(m)}, \mathbf{0}^{(n)}, \mathbf{0}^{(p)})$ is true iff m + n = p;
$M(\mathbf{0}^{(m)}, \mathbf{0}^{(n)}, \mathbf{0}^{(p)})$ is true iff $m \cdot n = p$;
$\sim A$ is true iff A is not true;
$(A \supset B)$ is true iff either A is not true or B is true;
$(x)A(x)$ is true iff for all n, $A(\mathbf{0}^{(n)})$ is true.

We can obtain $A(\alpha)$ by writing out this definition in the language of arithmetic, replacing "x is true" by "$\alpha(x) = 1$". From our previous work, we have arithmetical formulae Sent(x), At(x) and TrAt(x) which define the set of sentences of the language of arithmetic, the set of atomic sentences, and the set of true atomic sentences, respectively. We can therefore write $A(\alpha)$ as follows:

$$(x)[\alpha(x) \leq \mathbf{0}' \wedge (\alpha(x) = \mathbf{0}' \supset \mathrm{Sent}(x)) \wedge$$
$$(\mathrm{At}(x) \supset (\alpha(x) = \mathbf{0}' \equiv \mathrm{TrAt}(x))) \wedge$$
$$(y)(z)(i)\{(\mathrm{Neg}(x, y) \supset \alpha(x)+\alpha(y) = \mathbf{0}') \wedge$$
$$(\mathrm{Cond}(x, y, z) \supset [\alpha(x) = \mathbf{0}' \equiv (\alpha(y) = \mathbf{0} \vee \alpha(z) = \mathbf{0}')]) \wedge$$
$$(\mathrm{UQ}(x, y, i) \supset [\alpha(x) = \mathbf{0}' \equiv (n)(w)(\mathrm{Subst}_2(y, w, i, n) \supset \alpha(w) = \mathbf{0}')])\}]$$

Where Neg(x, y) holds iff x is the negation of y, Cond(x, y, z) holds iff x is the conditional $(y \supset z)$, and UQ(x, y) holds iff x is the result of attaching the universal quantifier (x_i) to x. We leave it to the reader to verify that this works.

Once we know that all Δ^1_1 sets are hyperarithmetical, it will turn out that the set of truths of the language of arithmetic is also hyperarithmetical. We can also give a direct proof that this set is hyperarithmetical; in fact, it turns out to be recursively isomorphic to the set H_n, where $|n|_R = \omega$, that is, it appears at the first level of the hyperarithmetical hierarchy that is beyond the arithmetical sets.

Lecture XXV

The Baire Category Theorem.

A subset S of Baire space is said to be *dense* if for any finite sequence s, there is an $\alpha \in S$ that extends s. (This definition coincides with the general definition of "dense" for topological spaces.)

Theorem: The intersection of a countable family of dense open sets is nonempty.
Proof: Let O_1, O_2, ... be dense open sets. We shall construct a function $\alpha \in \cap_n O_n$ as follows. Let s_0 be the empty sequence. If s_n has been defined, let $\alpha \in O_{n+1}$ be such that α extends s_n; this is possible because O_{n+1} is dense. Since O_{n+1} is open, there is an initial segment t of α such that every function extending t is in O_{n+1}. Let s_{n+1} be some finite sequence that properly extends both s_n and t.

We have thus defined a sequence s_0, s_1, ... of finite sequences such that $i > j$ implies that s_i properly extends s_j, and such that any function extending s_n (for $n > 0$) is an element of O_n. Let $\alpha = \cup_n s_n$; α is a total function. Moreover, since α extends each s_n, $\alpha \in O_n$ for all n, i.e. $\alpha \in \cap_n O_n$.

This is a special case of a more general theorem, known as the *Baire Category Theorem*. (The proof of the general theorem is essentially the same as the present proof.) Notice that for the theorem to go through, it suffices that each O_n *contain* some dense open set, since if for all n O_n' is a dense open subset of O_n, then we can apply the theorem to find $\alpha \in \cap_n O_n'$, whence $\alpha \in \cap_n O_n$. (Any set containing a dense set is itself dense, so if O_n contains a dense open set at all, the interior or O_n (i.e. the union of all the open sets contained in O_n) will be a dense open set. Thus, we can take O_n' to be the interior of O_n.) Notice also that O_1 need not be dense, but merely nonempty and open, since then we can let s_1 be any sequence all of whose total extensions are in O_1.)

The Baire Category Theorem turns out to have many applications in logic, and if there is a single most important principle in logic, it is probably this theorem. It is usually applied in the following way. Suppose we want to show that there is a function that satisfies a certain condition C. If we can break C down into a countable family of conditions, then we can find such a function if we can find a single function that satisfies all of those conditions simultaneously. If we can arrange things so that each of these conditions is dense and open (or contains a dense open condition), then the theorem guarantees that such a function exists.

Cohen's famous proof of the independence of the continuum hypothesis can be seen as an application of the category theorem. The theorem can also be seen as a generalization of

Cantor's diagonal argument. In particular, we can use it to show that there are uncountably many total functions on **N**. To see this, let F be any countable family of such functions, and for each $\alpha \in F$, let $O_\alpha = \{\beta: \beta \neq \alpha\}$. Each O_α is open, since two functions are different iff they disagree on some initial segment, and each O_α is dense, since any finite sequence can be extended to a function different from α. It follows that there is a function β such that $\beta \in O_\alpha$ for each $\alpha \in F$, i.e. such that $\beta \notin F$. (This application of the category theorem really boils down to Cantor's own proof, since in the latter a function outside F is constructed stage by stage in just the same way that the function α is constructed in the former.)

Incomparable Degrees.

Let us now consider an application of this theorem. For all we have said so far, the Turing degrees might be linearly ordered. It turns out that they are far from being linearly ordered; in this section, we shall construct a pair of incomparable degrees, i.e. degrees a and b such that neither $a \leq b$ nor $b \leq a$.

Call a pair of functions *recursively incomparable* if neither is recursive in the other. To find a pair of incomparable degrees, it suffices to find a pair of recursively incomparable functions, for then those functions will be of incomparable degrees. Recall that a function α is recursive in β just in case α is definable in the language RE with an extra function symbol for β. Let us define W_e^β to be the relation $\{<k, p>: (\exists s)(s \text{ is an initial segment of } \beta \text{ and } W(e, s, k, p))\}$, and let us identify functions with their graphs. Then α is recursive in β just in case $\alpha = W_e^\beta$ for some e, and α is nonrecursive in β iff $\alpha \neq W_e^\beta$ for all e. Thus, α and β will be recursively incomparable if they satisfy all of the conditions $\alpha \neq W_e^\beta$ and $\beta \neq W_e^\alpha$ simultaneously; to find such α and β, we need only show that those conditions contain dense open conditions.

Theorem: There are incomparable Turing degrees.
Proof: For any e, let $A_e = \{[\alpha, \beta]: \alpha \neq W_e^\beta\}$ and $B_e = \{[\alpha, \beta]: \beta \neq W_e^\alpha\}$. If each of the A_e's and B_e's has a dense open subset, then we can apply the Baire category theorem to obtain α and β such that $[\alpha, \beta]$ is in A_e and B_e for each e, from which it follows that α and β are recursively incomparable. We show that A_e has a dense open subset; the proof that B_e does is the same.

Let $A_e' = \{\gamma \in A_e: (\exists s)(s \text{ is an initial segment of } \gamma, \text{ and any function extending s is in } A_e\}$. A_e' is open, for let $\gamma \in A_e'$ and let $s \subseteq \gamma$ be such that any function extending s is in A_e; then any function extending s is also in A_e'. (In fact, A_e' is the interior of A_e.) We need only show that A_e' is dense.

Let s be any finite sequence, and let s_1 and s_2 be the even and odd parts of s, respectively (that is, if $s = <x_0, ..., x_n>$, with $n = 2m$, then $s_1 = <x_0, x_2, ..., x_{2m}>$ and $s_2 = <x_1, x_3, ..., x_{2m-1}>$, and similarly if $n = 2m+1$); we need to show that some function

extending s is in A_e'. It suffices to find an s' extending s (or extended by s) such that any function extending s' is in A_e. Notice that if $\gamma = [\alpha, \beta]$, then γ extends s iff α extends s_1 and β extends s_2.

Case 1: $W_e^\beta \subseteq s_1$ for all β extending s_2. In that case, $W_e^\beta \neq \alpha$ whenever α and β extend s_1 and s_2, because any such α is total and therefore properly extends s_1, so we can let s = s'.

Case 2: $W_e^\beta \not\subseteq s_1$ for some β extending s_2. Fix β, and let $<k, p> \in W_e^\beta - s_1$. Let s_2' be an initial segment of β such that $W(e, s_2', k, p)$; then $<k, p> \in W_e^{\beta'}$ for all β' extending s_2'. We can find an extension s_1' of s_1 such that $<k, p> \notin \alpha'$ for all α' extending s_1': either s_1' has a kth element that is different from p, in which we can let $s_1' = s_1$, or s_1' has no kth element, in which we can let s_1' be an extension of s_1 whose kth element is different from p. Let s' be an extension of s such that the even and odd parts of s' extend s_1' and s_2'. Then whenever $[\alpha, \beta]$ extends s', $[\alpha, \beta] \in A_e$, as required.

We can also give a direct proof that does not appeal directly to the category theorem.

Second Proof: We construct finite sequences $s_0, s_1, s_2, \ldots$ and $t_0, t_1, t_2, \ldots$ such that $\alpha = \cup_n s_n$ and $\beta = \cup_n t_n$ are total functions; we then show that α and β are recursively incomparable.

Let $s_0 = t_0 = \emptyset$. Suppose s_n and t_n have been defined. If n = 2m, we proceed as follows. If $W_m^\beta \subseteq s_n$ for all extensions β of t_n, then let s_{n+1} and t_{n+1} be any finite sequences that extend s_n and t_n. Otherwise, find an extension t_n' of t_n and a pair $<k, p>$ such that $W(e, t_n', k, p)$, and let s_n' be an extension of s_n such that $\alpha(k) \neq p$ for all extensions α of s_n', as in the first proof. Let $s_{n+1} = s_n'$ and $t_{n+1} = t_n'$. If n = 2m+1, then do exactly the same, except reversing the roles of s and t.

Now let $\alpha = \cup_n s_n$ and $\beta = \cup_n t_n$. If α is recursive in β, then $\alpha = W_m^\beta$ for some m; but α and β extend s_{2m} and t_{2m}, and it is clear from the construction of the s's and t's that $\alpha \neq W_m^\beta$ for any such α and β. So α is not recursive in β, and by same argument β is not recursive in α, i.e. α and β are recursively incomparable.

The construction of the s's and t's in this proof is not effective, since if it were, α and β would be recursive and therefore recursively comparable. In particular, we cannot effectively decide whether $W_m^\beta \subseteq s_n$ for all extensions β of t_n, since that would involve surveying all the infinitely many extensions of t_n. However, if we had an oracle which gave us the answer to this question, we could use it to effectively construct α and β, so α and β would be recursive in the oracle. We can therefore modify the proof to place an upper bound on the Turing degrees of α and β.

Theorem: There are incomparable Turing degrees below 0'.
Proof (sketch): It suffices to show that the functions α and β constructed in the above

proof are recursive in 0'. Consider the relation R = {<s, t, m>: $W_m^\beta \subseteq s$ for all β extending t}. The relation -R is r.e., since <s, t, m> ∈ -R just in case (∃t' extending t)(∃k)(∃p)(W(m, t', k, p) ∧ <k, p> ∉ s). It follows that both R and -R are recursive in 0'. Let φ be a partial function which uniformizes the r.e. relation {<s, t, [t', k, p]>: t' extends t ∧ W(m, t', k, p) ∧ <k, p> ∉ s}.

We can then construct s_n and t_n effectively in terms of R and φ. Specifically, we set s_0 = t_0 = the code of the empty sequence. If n = 2m, we set $s_{n+1} = s_n{}^\wedge\langle 0\rangle$ (i.e. the concatenation of s_n with the unit sequence <0>) and $t_{n+1} = t_n{}^\wedge\langle 0\rangle$ if R(s_n, t_n, m) holds. If R(s_n, t_n, m) doesn't hold, let [t', k, p] = φ(s_n, t_n). Then t' extends t_n, and for any β extending t', <k, p> ∈ W_m^β - s_n. We then let t_{n+1} = t' and let s_{n+1} be some extension of s_n such that the kth element of s_{n+1} exists and is different from p. (s_{n+1} can obviously be found effectively.) If n = 2m+1, we do the same, but with the roles of s and t reversed.

So we see that the maps n → s_n and n → t_n are recursive in 0'. α and β are therefore also recursive in 0', since α(n) = the nth member of the sequence $s_{2(n+1)}$ and β(n) = the nth member of the sequence $t_{2(n+1)}$.

This theorem was originally proved by Kleene and Post. Notice that it does not show that there are any incomparable r.e. degrees, since a degree can be below 0' without containing any r.e. sets. In fact, the proof that incomparable r.e. degrees exist is a souped-up version of the proof we just gave.

We can also get a refinement of these results:

Theorem: For any nonrecursive degree a, there is a degree incomparable with a.
Proof: Let α be a total function of degree a; we need to find a function β recursively incomparable with α. For all e, let A_e = {β: α ≠ W_e^β} and B_e = {β: β ≠ W_e^α}; it suffices to show that each A_e and each B_e has a dense open subset. B_e is dense and open already, as is easily seen. Let A_e' be the interior of A_e as before, i.e. A_e' = {β: (∃s ⊆ β)(β')(if β' extends s then α ≠ $W_e^{\beta'}$}. We need only show that A_e' is dense.

Let s be any finite sequence; we need to show that A_e' contains some function extending s, i.e. that there is a sequence s' extending s such that for all β extending s', α ≠ W_e^β. Suppose this is not the case; then for all s' extending s there is a β extending s' such that α = W_e^β. In that case, α = {<k, p>: (∃s')(s ⊆ s' ∧ W(e, s', k, p))}. (Suppose <k, p> ∈ α; since there is a β extending s such that α = W_e^β, there is an s' extending s such that W(e, s', k, p). On the other hand, if s' extends s and W(e, s', k, p), then <k, p> ∈ W_e^β for all β extending s'; since W_e^β = α for *some* such β, it follows that <k, p> ∈ α.) But in that case, α is an r.e. relation and is therefore a recursive function, contradicting our assumption that a is a nonrecursive degree.

We can refine this a bit further and show that for any nonrecursive degree a, there is a degree b below a' that is incomparable with a. The proof of this is like the proof that there

are incomparable degrees below 0'. (Notice that this result does not directly imply that there are incomparable degrees below 0', because we cannot take a = 0.)

The Separation Theorem for $\mathbf{S}^1_1$ Sets.

In this section, we show that every $\mathbf{D}^1_1$ set is Borel. In fact, we shall prove something stronger. Call a pair (S_1, S_2) *Borel separable* if there is a Borel set which contains S_1 and is disjoint from S_2. We shall prove a theorem due to Lusin, namely that any disjoint pair of $\mathbf{S}^1_1$ sets is Borel separable. If S is $\mathbf{D}^1_1$, then (S, -S) is a disjoint pair of $\mathbf{S}^1_1$ sets, so there is a Borel set B which separates S from -S; but then S = B, and therefore S is Borel.

Notice that a set S is Borel inseparable from a set T iff there is no Borel set B with $S \subseteq B \subseteq -T$. We begin by proving the following.

Lemma: If $S = \cup_n S_n$ and S is Borel inseparable from T, then there is an n such that S_n is Borel inseparable from T.
Proof: Suppose S_n is Borel separable from T for each n. For each n, let B_n be a Borel set such that $S_n \subseteq B_n \subseteq -T$. Then $\cup_n S_n \subseteq \cup_n B_n \subseteq -T$. But then $S = \cup_n S_n$ is Borel separable from T, since $\cup_n B_n$ is Borel.

Corollary: If the sets of two countable unions are pairwise Borel separable, then the two unions are Borel separable.

Theorem: Any two disjoint $\mathbf{S}^1_1$ sets are Borel separable.
Proof: Let S_1 and S_2 be any two $\mathbf{S}^1_1$ sets, and assume that S_1 and S_2 are Borel inseparable. We show that $S_1 \cap S_2 \neq 0$. Since S_1 and S_2 are $\mathbf{\Sigma}^1_1$, there are relations R_1 and R_2 on **N** such that

$$S_1 = \{\beta_1: (\exists\alpha_1)(x)R_1(\overline{\alpha}_1(x), \overline{\beta}_1(x))\},$$
$$S_2 = \{\beta_2: (\exists\alpha_2)(x)R_2(\overline{\alpha}_2(x), \overline{\beta}_2(x))\}.$$

We construct four infinite sequences $\langle a_1^{(n)}\rangle$, $\langle b_1^{(n)}\rangle$, $\langle a_2^{(n)}\rangle$, $\langle b_2^{(n)}\rangle$, where $a_1^{(n)}$, etc. are sequences of length n. First, set $a_1^{(0)} = b_1^{(0)} = a_2^{(0)} = b_2^{(0)}$ = the empty sequence. If s and t are finite sequences, let

$$S_1^{s,\,t} = \{\beta_1: (\exists\alpha_1)(s \subseteq \alpha_1 \wedge t \subseteq \beta_1 \wedge (x)R_1(\overline{\alpha}_1(x), \overline{\beta}_1(x)))\}$$

and define $S_2^{s,\,t}$ similarly. In particular, let $S_1^{(n)} = S_1^{a_1^{(n)},\, b_1^{(n)}}$ and $S_2^{(n)} = S_2^{a_2^{(n)},\, b_2^{(n)}}$. Then

$$S_1^{(n)} = \{\beta_1: (\exists\alpha_1)(\overline{\alpha}_1(n) = a_1^{(n)} \wedge \overline{\beta}_1(n) = b_1^{(n)} \wedge (x)R_1(\overline{\alpha}_1(x), \overline{\beta}_1(x)))\}$$

and similarly for $S_2^{(n)}$. Now suppose $a_1^{(n)}$, $b_1^{(n)}$, etc. have been defined and $S_1^{(n)}$ is Borel inseparable from $S_2^{(n)}$, and define $a_1^{(n+1)}$, etc. as follows. Notice that $S_1^{(n)} = \cup\{S_1^{s, t}$: s and t are sequences of length n+1 that extend the sequences $a_1^{(n)}$ and $b_1^{(n)}$, respectively}, so by our lemma, we can find such s and t so that $S_1^{s, t}$ is Borel inseparable from $S_2^{(n)}$. Let $a_1^{(n+1)}$ = s and $b_1^{(n+1)}$ = t for some such s and t. So $S_1^{(n+1)}$ is Borel inseparable from $S_2^{(n)}$. Similarly, we can find sequences s and t of length n+1 which extend $a_2^{(n)}$ and $b_2^{(n)}$ and such that $S_2^{s, t}$ is Borel inseparable from $S_1^{(n)}$; let $a_2^{(n)}$= s and $b_2^{(n)}$= t for some such s and t. $S_1^{(n+1)}$ and $S_2^{(n+1)}$ are therefore Borel inseparable.

Thus, the $S_1^{(n)}$'s and $S_2^{(n)}$'s are progressively narrower subsets of S_1 and S_2 that are Borel inseparable if S_1 and S_2 themselves are. Moreover, $a_1^{(n)}$ properly extends $a_1^{(m)}$ when n > m, and similarly for b_1, a_2, and b_2; so we can define $\alpha_1 = \cup_n a_1^{(n)}$, $\beta_1 = \cup_n b_1^{(n)}$, and similarly for α_2 and β_2.

Observe that $\beta_1 = \beta_2$. For suppose not; then $\beta_1(n) \neq \beta_2(n)$ for some n. Let $p = \beta_1(n)$, and let $O = \{\beta: \beta(n) = p\}$. O is open, as is easily seen, and is therefore Borel. However, $S_1^{(n+1)} \subseteq O$ and $S_2^{(n+1)} \subseteq$ -O, so $S_1^{(n+1)}$ and $S_2^{(n+1)}$ are Borel separable, contradiction.

We now show that S_1 intersects S_2 by showing that $\beta_1 \in S_1 \cap S_2$. To prove this, it suffices to show that $(x)R_1(\overline{\alpha}_1(x), \overline{\beta}_1(x))$ and $(x)R_2(\overline{\alpha}_2(x), \overline{\beta}_2(x))$. We prove the former; the proof of the latter is the same. Suppose $\sim(x)R_1(\overline{\alpha}_1(x), \overline{\beta}_1(x))$. Then for some n, $\sim R_1(\overline{\alpha})_1(n), \overline{\beta}_1(n))$. By our definition of $a_1^{(n)}$ and $b_1^{(n)}$, this just means that $\sim R_1(a_1^{(n)}, b_1^{(n)})$. It follows that $S_1^{(n)} = \emptyset$. But then $S_1^{(n)}$ is a Borel set that separates itself from $S_2^{(n)}$, which is impossible. This concludes the proof.

We have already seen that all Borel sets are $\mathbf{D}_1^1$; we have therefore established Suslin's theorem: $\mathbf{D}_1^1$ = Borel. This is an unusually simple proof for such a sophisticated result. Notice that it is similar in flavor to the proof of the existence of incomparable degrees; in particular, the function β_1 is constructed via a stage-by-stage process.

Exercises

1. Consider the language of arithmetic with one extra predicate P(x).

(a) Consider a system in this languaßge whose axioms are an r.e. set of the language of arithmetic containing Q. Add the axioms $P(\mathbf{0}^{(n)})$ for each $n \in S$, where S is any set. No axioms except these contain the extra predicate P. Assume that the resulting system is ω-consistent. On these assumptions, characterize the sets weakly representable in the system and prove the characterization.

(b) Make the same assumptions as in (a) except that now we have $P(\mathbf{0}^{(n)})$ for each $n \in S$ and $\sim P(\mathbf{0}^{(n)})$ for each $n \notin S$. Characterize the weakly representable sets and prove the answer.

(c) Under the assumptions of (b), characterize the strongly representable (binumerable) sets, and prove the answer.

2. Consider a system in the second order language of arithmetic (i.e., the language of arithmetic supplemented with variables and quantifiers for 1-place number theoretic functions), with a Π^1_1 set of axioms containing at least the axioms of Q, and with the ω-rule added to the usual logical rules.

(a) Under the assumption that all the axioms are true, define a statement analogous to "'Gödel heterological" is Gödel heterological' and show that it is true but undecidable.

(b) Show that every true Π^1_1 sentence is provable. Use this to show that if all of the axioms are true, then the sets weakly representable in the system are precisely the Π^1_1 sets. (Hint: Prove the contrapositive (i.e., that if a Π^1_1 sentence is not provable, then it is not true) by a method similar to the proof of the $\mathbf{S}^1_1$ separation theorem and its corollary the Suslin characterization of the $\mathbf{D}^1_1$ sets. You may assume in your proof that the system contains any reasonable axioms, over and above those in the language of arithmetic, to handle function quantifiers; in particular, relevant axioms might include $(m)((n)(\alpha(m)=n\supset A(\alpha)))\supset A(\alpha)$.)

(c) Let A(x) be a Σ^1_1 formula with one free number variable. Using (b), show that if the system is consistent and the set defined is not Π^1_1, then some sentence of the form $A(\mathbf{0}^{(n)})$ is true but unprovable.

(d) Show that if the system is consistent (not necessarily true), then the statement "'Gödel heterological" is Gödel heterological' of part (a) must be true but unprovable.

3. Let R be any binary relation on the natural numbers. Suppose that for any partial recursive function ϕ there is a total recursive function ψ such that $R(\psi(x),\phi(x))$ whenever $\phi(x)$ is defined. Prove that, under this hypothesis, for any total recursive function χ there is a number m such that $R(m,\chi(m))$. Show that immediate consequences of this principle for suitable choices of R are: the self-reference lemma, that every maximal enumeration has the fixed-point property, and that there are two disjoint r.e. sets without recursive separation.