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:

$$\begin{tikzcd}
S_1 \leq_1 S_2 \arrow[r, Rightarrow] & S_1 \leq_m S_2 \arrow[r, Rightarrow] \arrow[d, Rightarrow] & S_1 \leq_T S_2 \arrow[d, Rightarrow] \\
 & S_1 \leq_e S_2 \arrow[r, Rightarrow] & S_1 \text{ r.e. in } S_2
\end{tikzcd}$$

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,

$$\begin{tikzcd}
S_1 \leq_1 S_2 \arrow[r, Leftrightarrow] \arrow[d, Rightarrow] & -S_1 \leq_1 -S_2 \arrow[d, Rightarrow] \\
S_1 \leq_m S_2 \arrow[r, Leftrightarrow] \arrow[d, Rightarrow] & -S_1 \leq_m -S_2 \arrow[d, Rightarrow] \\
S_1 \leq_T S_2 \arrow[r, Leftrightarrow] & -S_1 \leq_T -S_2
\end{tikzcd}$$

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:

$$\begin{tikzcd}
S_1 \leq_1 S_2 \arrow[r, Rightarrow] & S_1 \leq_m S_2 \arrow[r, Rightarrow] & S_1 \leq_{ee} S_2 \arrow[r, Rightarrow] \arrow[d, Rightarrow] & S_1 \leq_T S_2 \arrow[d, Rightarrow] \\
 & & S_1 \leq_e S_2 \arrow[r, Rightarrow] & S_1 \text{ r.e. in } S_2
\end{tikzcd}$$

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^n_1] (resp. RE[f^n_1]) have the same expressive power, where P^n_1 is interpreted as the relation R (resp. f^n_1 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^2_1], let A* be the formula of RE[P^1_1] that comes from A by replacing all occurrences of $P^2_1(x, y)$ 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^2_1] 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 $x \to 2x$, and in the case of S_2 by the map $x \to 2x + 1$. 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 $G = \cup\{S'\colon S'$ 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 $S_n \subseteq G$. $S_0 = Ø \subseteq G$. 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. $R_0 = Ø$ 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 $S_1 = \{n: (\exists \text{ finite } S_0)(S_0 \subseteq S_2 \wedge R(n, S_0))\}$. 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 $S_1 = \{n: (\exists \text{ finite } S_0)(S_0 \subseteq S_2 \wedge R(n, S_0))\}$. 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 $\mathrm{deg}_e(A)$ ($\mathrm{deg}_T(A)$, 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 $\mathrm{deg}_T(A) \leq \mathrm{deg}_T(B)$ iff $A \leq_T B$ (and similarly for other kinds of degrees). It is easy to check that ≤ is well-defined, and that it partially orders the degrees. When "≤" 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 ≤ 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 $A \leq_e B \cup C$, so in particular, A is r.e. in B iff $A \leq_e B, -B$ iff $A \leq_e B \cup -B$. Recall also that $A \cup B \leq_e C$ iff $A \leq_e C$ and $B \leq_e C$. It follows that $A \leq_T B$ iff $A \cup -A \leq_e B \cup -B$.

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 $(S \cup -S)^*$. 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: $(S \cup -S)^* \leq_e S \cup -S$ by the properties of *, i.e. $(S \cup -S)^*$ 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 $-(S \cup -S)^* \leq_e S \cup -S$, 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 *, $(S \cup -S) \leq_e (S \cup -S)^*$, 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 $A \leq_e S \cup -S$, this follows directly from our equivalences for * by replacing S by $S \cup -S$.

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 $A \cup -A$ and $B \cup -B$, 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. Deg(B') = 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 <{T-degrees}, ≤> onto a subset of the structure <{1-degrees}, ≤>. More precisely, the map $\text{Deg}_T(A) \rightarrow \text{Deg}_1(A')$ is such an embedding. This is simply because $\text{Deg}_T(A) \leq \text{Deg}_T(B)$ iff $A \leq_T B$ iff $A' \leq_T B'$ iff $\text{Deg}_1(A') \leq \text{Deg}_1(B')$. In fact, the same argument shows that the map $\text{Deg}_e(A) \rightarrow \text{Deg}_1(A^*)$ 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 $A \cup -A \leq_e B \cup -B$; it follows that the map $\text{Deg}_T(A) \rightarrow \text{Deg}_e(A \cup -A)$ 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 $A \cup -A$. An enumeration degree is also total iff it contains the graph of a total function (hence the name).

Let f be the embedding $\text{Deg}_T(A) \rightarrow \text{Deg}_e(A \cup -A)$, and let g be the embedding $\text{Deg}_e(A) \rightarrow \text{Deg}_1(A^*)$. 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 $\text{Deg}_T(A) \rightarrow \text{Deg}_1(A')$ 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 $\mathbf{U}$ 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 $\mathbf{U}$ 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 $\phi: A \leq_m A_1$ and $\psi: B \leq_m B_1$. Let χ be the recursive function such that $\chi(2n) = 2\phi(n)$ and $\chi(2n+1) = 2\psi(n)+1$; $\chi: A \mathbf{U} B \leq_m A_1 \mathbf{U} B_1$. 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 $\mathbf{U}$ B is an upper bound of A and B in all of our reducibility notions, $\text{Deg}_r(A)$ and $\text{Deg}_r(B)$ are $\leq \text{Deg}_r(A \mathbf{U} B)$ 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 $Ø' = (Ø \cup \mathbf{N})^* = \{odds\}^* = \{[e, m]: (\exists s)(s \subseteq \{odds\} \wedge W(e, m, s))\}$; 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^0_n) 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 ~A defines -S; but ~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.

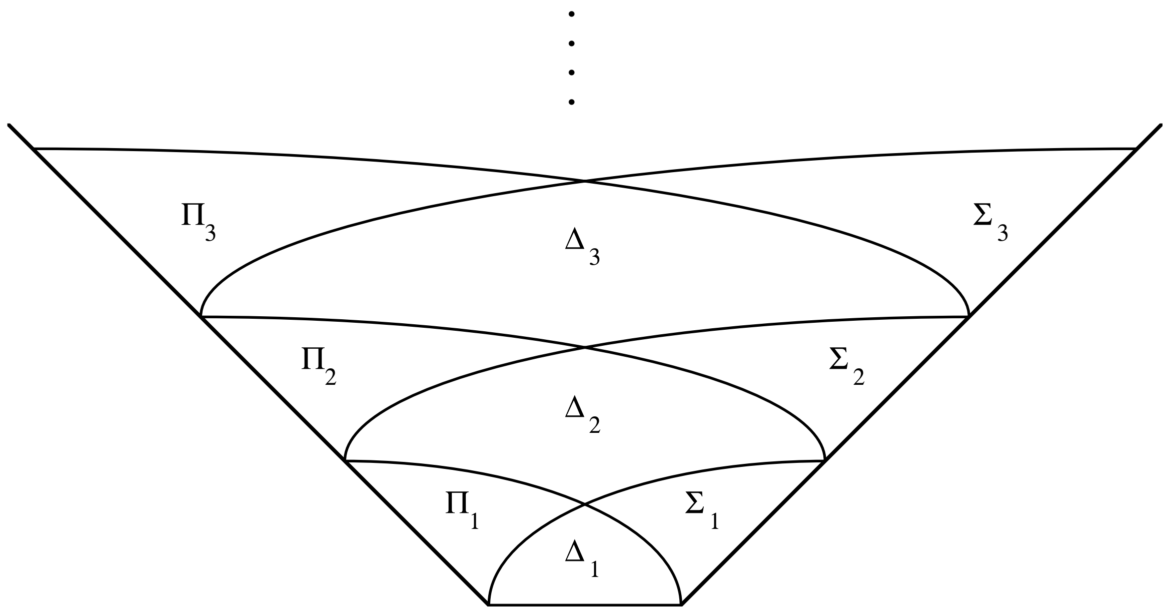
(There is also a Σ_n (Π_n) enumeration of the Σ_n (Π_n) k-place relations, for all k; we could either generalize the proof in the case k = 1, or use the pairing function.)

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 ~(A $\wedge$ B) is equivalent to ~A $\vee$ ~B, where ~A and ~B are Σ_n; so by the theorem, ~A $\vee$ ~B, and hence ~(A $\wedge$ B), is Σ_n, and therefore A $\wedge$ B is Π_n. Similarly, if A is Π_n, then ~(x)A is equivalent to ($\exists$x)~A, and ~A is Σ_n, so by the theorem ($\exists$x)~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. ~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 $Ø^{(n)}$ 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 Ø. But a set is r.e. in Ø 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 $Ø^{(n)}$. By the properties of the jump operator, $Ø^{(n)} = Ø^{(n-1)'}$ is r.e. in $Ø^{(n-1)}$. By the inductive hypothesis, then, $Ø^{(n)}$ 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 $Ø^{(n-1)}$. By the jump properties, $S_1 \leq_1 Ø^{(n-1)'} = Ø^{(n)}$, so *a fortiori* $S_1 \leq_T Ø^{(n)}$. By the weak transitivity property of *r.e. in*, S is r.e. in $Ø^{(n)}$.

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 $Ø^{(n)}$.

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 ≠ NP. Obviously, if a relativization principle held in complexity theory, then we would have either P = NP for all oracles or P ≠ 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

$$\text{(T)} \qquad A(\mathbf{0}^{(m)}) \equiv B$$

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.