THE BULLETIN OF SYMBOLIC LOGIC
Volume 1, Number 3, Sept. 1995

HOW TO COMPUTE ANTIDERIVATIVES

CHRIS FREILING

This is *not* about the symbolic manipulation of functions so popular these days. Rather it is about the more abstract, but infinitely less practical, problem of the primitive. Simply stated:

Given a derivative $f : \mathbb{R} \to \mathbb{R}$, how can we recover its primitive?

The roots of this problem go back to the beginnings of calculus and it is even sometimes called "Newton's problem". Historically, it has played a major role in the development of the theory of the integral. For example, it was Lebesgue's primary motivation behind his theory of measure and integration. Indeed, the Lebesgue integral solves the primitive problem for the important special case when $f(x)$ is bounded. Yet, as Lebesgue noted with apparent regret, there are very simple derivatives (e.g., the derivative of $F(0) = 0$, $F(x) = x^2 \sin(1/x^2)$ for $x \neq 0$) which cannot be inverted using his integral.

The general problem of the primitive was finally solved in 1912 by A. Denjoy. But his integration process was more complicated than that of Lebesgue. Denjoy's basic idea was to first calculate the definite integral $\int_a^b f(x)\,dx$ over as many intervals (a, b) as possible, using Lebesgue integration. Then, he showed that by using these results, the definite integral could be found over even more intervals, either by using the standard improper integral technique of Cauchy, or an extension technique developed by Lebesgue (see appendix for details). By proving that at least one of these techniques would always succeed, the process could be continued until the definite integral over all possible intervals was obtained. At this point, the antiderivative $F(x) = \int_0^x f(x)\,dx$ (up to a constant) becomes apparent. The trouble with Denjoy's procedure is that it needs to be continued transfinitely and, in fact, may require arbitrarily large countable ordinals to complete. He called his process "totalization". The question was immediately raised (for example in Lusin's thesis) as to whether such use of transfinite numbers was really necessary. Could perhaps a different approach avoid these countable ordinals (or at least arbitrarily large ones) and still recover the primitive?

Received March 15, 1995.
Research supported by the National Science Foundation.
The author would like to thank the referee for an unusually helpful and thorough report.

1079-8986/95/0103-0002/$4.80

In 1915, H. Bauer, using an integral introduced by O. Perron a year earlier, proposed a brand new solution. By utilizing the concept of "major" and "minor" functions, introduced by de la Vallee-Poussin, it was able to avoid any mention of transfinite ordinals. Compared to totalization, this new integral was incredibly simple and much easier to understand and work with (see appendix for details). Accordingly, the Perron-Bauer solution was enthusiastically received by many and gradually became the standard for further investigation. Still, some people (e.g., Looman [11]) complained that by avoiding transfinite ordinals something crucial was lost. Denjoy himself was extremely critical of the Perron-Bauer approach and over the years seemed to become increasingly bitter. Some of his later writings contain scathing attacks on Perron personally as well as his integral.

To understand Denjoy's complaint (about the integral), it helps to consider a third solution to the problem, which is rather silly and reminiscent of a famous joke by the comedian Steve Martin:

> You can be a millionaire and never pay taxes. That's right! You can be a millionaire and never pay taxes. How? It's easy. Two simple steps: First, get a million dollars. Then when they ask you why didn't you pay taxes say "I forgot" and when they say "You forgot!?" say "Well excuuuuuuuuuuse me".

The Steve Martin solution to the primitive problem might go something like this: First, get the right function. Then, show it has the derivative you were searching for.

Now try to ignore the fact that this "Martin integral" is probably the most powerful method of integration known and, in practice, has successfully computed more antiderivatives than all other solutions combined. Consider instead how hard it is to "guess" the antiderivative. For example, try to find the integral of $\sin x/x$. Well, ok you might be able to "guess" some sort of infinite series solution. But things get worse than this. Dougherty and Kechris [8] have shown (using Y. Matiyasevich's work on diophantine representation of recursively enumerable sets [12]) that there are derivatives which are analytically expressible (in terms of an explicit formula using the basic elementary functions sin, cos, exponents, absolute values, etc., and the elementary operations of multiplication, division, composition, and infinite sums) but whose primitive is immensely complicated, so that for example, there is no way to analytically express the primitive. The Martin solution, therefore, cannot seriously be considered a solution at all. But it does seem to illustrate what is the real problem of the primitive and that is—the problem itself. Maybe, the Denjoy-Perron controversy would not have been so bitter had the problem been proposed in a more precise manner. What does it mean to be "given a derivative"? or to "recover a primitive"? This is where the logicians come in.

Let's call a solution "nonconstructive" (which is a polite way of saying that as far as solving the primitive problem, it is just as silly as the Martin integral) if it involves a search over all continuous functions, or something which is morally equivalent to this (e.g., a search over all real numbers). The Perron integral is nonconstructive (see appendix). Nearly a half century later, the Riemann-complete integral was introduced by J. Kurzweil and then rediscovered and developed by R. Henstock. The name comes because it is just a slight variation of the classical Riemann integral, yet it is strong enough to invert derivatives (see appendix). Unlike the Perron integral, it has a natural appeal and really has something substantial to say. Sadly, it too is nonconstructive. In fact, so is just about every other solution which has ever been proposed (see [3]) except those which are based on the original totalization procedure of Denjoy. But does totalization also contain something which makes it nonconstructive? A negative answer was given in the mid-1980s by M. Ajtai. He showed that totalization (or at least a variation of it) follows a very strict and very precise notion of definability (perhaps if Denjoy had known about this he could have expressed his criticism in a more "constructive" manner). Ajtai's result is unpublished but it is referenced in an article by Dougherty and Kechris [8], who outline their own proof of this result. They call this type of definability "Δ_1^1 on the set of derivative codes". But their main thrust was to show that no substantial improvement to this will ever be possible for *any* solution to the primitive problem. So, in a sense, they classified the complexity of the operation of antidifferentiation, answered the question from Lusin's thesis, and proved that Denjoy's solution is and always will be in some sense the best! (something Denjoy probably suspected all along).

But what does any of this have to do with constructiveness? The problem said to "recover" the primitive, not "redefine" it. Well, these two notions can sometimes be very closely related. It follows from the pioneering work of S. Kleene, that there is always a computer type algorithm for carrying procedures which are "hyperarithmetically" defined. It seems natural therefore to try to exhibit such an algorithm. That way, the true essence of both the primitive problem and its solution could be grasped without first taking a year off to study Descriptive Set Theory [14]. Furthermore, since constructiveness is really the whole point, a computer program would be philosophically the most direct way of understanding it.

To accomplish this project, we first need a computer language for carrying out these Kleene-type algorithms. This has been provided by the work of Harel and Kozen and we will discuss it below.

Secondly, because of the sophistication of the techniques of Dougherty and Kechris, our project won't be feasible using their proof, and so we will need a new proof of their result, utilizing only monotone inductions (also explained below). Dougherty and Kechris used non-monotone inductions but they were able to get around this by carefully calculating bounds (using an argument of W. H. Woodin) for the ordinal lengths of their inductions.

Before we begin, let's be clear that we are not talking about a real computer here. Indeed, ordinary finite computers are useless even for the most elementary questions about real numbers. But that's all right. Our goal is not to present an algorithm which can be physically carried out. Rather, it is to use computer programs to illustrate and explain the constructive nature of antidifferentiation. Unfortunately, we will be forced to rely on the mind of the reader to act as our CPU and to do this for a program which we have no way of debugging. The solution may appear at first a little long as computer programs often do, but this is only because we are trying to give all the details. It is hoped that the reader will patiently follow these details in the beginning and that in this way, the notion of constructiveness becomes so ingrained that many of the details in the last part of the program can be easily skipped. We will assume familiarity with Lebesgue measure and integration, the Baire Category theorem, the Heine-Borel Theorem, and the notion of transfinite induction. For a reader who is already well versed in Descriptive Set Theory and only wishes to see the new proof of Ajtai's theorem, see §4.

§1. Warm up. If we wish to exhibit an algorithm for finding antiderivatives, ordinary computers are clearly inadequate. For one thing, an ordinary computer can only talk about integers and we need to talk about real numbers and functions. For another, every ordinary computation must be finite. But this does not mean that there can't be a high degree of constructiveness. For example, suppose we are given an infinite sequence of zeros and ones and wish to know if the sequence ever contains a one. Everyone agrees that there is a simple algorithm for doing this, even though it may be impossible to physically carry out. It seems that it would take a computer with a special ability (sometimes called the "infinite mind") to accomplish this task. The "infinite mind" of the computer would allow it to go through an infinite sequence of steps which are already known to be computable and then report if a certain event ever occurred. For reasons which will become apparent later, it is more convenient to imagine that this "infinite mind" checks each of these steps simultaneously rather than sequentially.

The "hyperarithmetical" sets of integers are intuitively the sets whose membership can be decided by such an "infinite minded" computer, and the result of Ajtai, Dougherty, and Kechris (combined with the work of Kleene) surprisingly says that this "infinite mind" is the only extra thing we need to calculate antiderivatives! But first, we have to mention a couple of more things. Since our computer is only allowed to talk about integers (rational numbers and finite strings of integers, etc., are also ok) we have to do all computations using only these objects. For example, we will say that a definite integral is "computed" if we have a program which can tell which rationals lie below the integral and which lie above it. Secondly, we need to be able to use the special "infinite mind" capability in ordinary ways naturally associated with computer programs. For example, using a simple subroutine, the "infinite mind" will also be able to tell if an infinite sequence of zeros and ones has infinitely many ones. But the real power of these computations comes from constructing loops using "**go to**" statements. Harel and Kozen [9] have developed the following computer programming language (called "IND") to make things more explicit. In its simplest form, the language consists of only three types of allowable statements.

l_1: $y \leftarrow \forall$ (or $y \leftarrow \exists$)
l_2: **accept** (or **reject**)
l_3: **if** $R(x)$ **go to** l_4

(where $R(x)$ is a relation which can be determined with an ordinary computer, i.e., "recursive".)

A program in this language consists of a finite sequence of such labeled statements. Statements of the first type access the "infinite mind" of the computer. Informally, if the statement has the form "$y \leftarrow \forall$" then the computer runs the remainder of the program for all values y of the appropriate data type (see next paragraph). If all of them lead to "**accept**" then the computer is said to accept at this step. If one of them leads to "**reject**" then the computer is said to reject at this step. In all other cases the program is said to have an infinite loop (∞) at this step. The semantics of $y \leftarrow \exists$ are similar and statements of the second and third type are self-explanatory.

The variables in the program will represent either natural numbers, integers, rationals, or finite sequences of these objects. The domain of a variable should be clear from the context. However, just in case, we will use i, j, k, m, n, M in the program to represent positive integers, ε will represent the reciprocal of a positive integer and all other lower case letters will represent rationals. When a variable appears with a bar over it then it will represent a finite sequence. For example, $\bar{p}$ will represent a finite sequence of rationals, $\langle p_1, p_2, \dots, p_{l(\bar{p})}\rangle$ where $l(\bar{p})$ is the length of the sequence. We make no distinction between numbers and sequences of length one. We also use $\bar{p} \wedge \bar{q}$ to represent the concatenation of two sequences, $\langle p_1, \dots, p_{l(\bar{p})}, q_1, \dots, q_{l(\bar{q})}\rangle$. It is clear that all of these objects can be effectively coded by integers.

For convenience, we will often list similar commands on a single line, separated by commas; e.g.,

l_{10}: $x \leftarrow \forall$, $y \leftarrow \forall$, $z \leftarrow \forall$.

Also, we usually put labels only on the statements which are referred to in a different part of the program. In addition, we will use the following "macros" from [9] which are easily computable from the allowable statements:

l_4: **go to** l_i,
l_5: **if** $R(x)$ **accept** (or **reject**), and
l_6: $y \leftarrow x$

which will be considered abbreviations respectively of:

l_4: **if** $0 = 0$ **go to** l_i,
l_5: **if** $R(x)$ **go to** l_7
  go to l_i
l_7: **accept** (or **reject**) (where l_i is the label of the next statement), and
l_6: $y \leftarrow \exists$
  if $y \neq x$ **reject**.

As a quick example, suppose we are "given" an infinite sequence of zeros and ones, i.e., $\alpha(1), \alpha(2), \ldots$. In the program we reflect that this sequence is "given" to us by allowing the $R(x)$ in the third type of statement to also be replaced by the relation $\alpha(x) = n$. Now, we can easily design a program which will accept if and only if the given sequence contains infinitely many ones. For example,

l_1: $n \leftarrow \forall$
  $m \leftarrow \exists$
  if $m < n$ **reject**
  if $\alpha(m) = 1$ **accept**
  reject

The IND programs provide us with a natural framework in which to do transfinite inductions and also provide the strictest notion of the word "constructive" which allows us to compute antiderivatives. It is at least in some sense possibly much stronger than what Denjoy had in mind when he developed his process of totalization. For example, while our special computer must have an infinite amount of memory available, we are only allowed to use finitely many variables, so that we are only able to store a finite amount of information. Therefore, not every transfinite induction on sets of integers can be programmed this way.

There are several ways to measure the time it takes to run a program in this language. For example, we might try to measure the time elapsed since the beginning of the computation, which we call "forward time", or instead we might try to measure the time from the end of the computation, which we call "backward time". In general these will be different. For example, if we use forward time and we imagine that the branching statements ($x \leftarrow \forall$ and $x \leftarrow \exists$) are handled in parallel then we might say that the running time is the supremum of the lengths of the branches in the resulting computation tree, or at least the part of the tree necessary to determine the output of the program. Since all of our programs are finite in length, the branches of such a tree, and hence the running time, will always be less than or equal to ω. This is not very useful for comparison purposes since most programs will have a running time equal to ω.

Backwards time is defined inductively on the same tree, working from the leaves to the root, assigning either a countable ordinal or else infinity (∞) to each node (see [9] for precise details). Roughly speaking, all the end nodes (accept or reject) are given a time value of 1. The time to completion at a "**go to**" statement is the time at the following node (either the next statement of the program or the destination of the "**go to**"), plus 1. Nodes of the tree associated with an "$x \leftarrow \forall$" statement will have infinitely many immediate successors. The time to completion is then the minimum time it takes for one of these successors to be rejected (in the case that there is such a successor), plus 1, or the supremum of the completion times at all immediate successors (in the case that they all lead to "**accept**"), plus 1, or else infinity (∞) (if neither of the other two cases holds). Nodes associated with statements of the form "$x \leftarrow \exists$" are handled similarly. When we talk about time (we only do this in Proposition 1) we prefer to use this backwards notion. For us it has the advantage of being able to say that if we are in the middle of a program then it takes less time to finish the program than it would to just start the whole thing over from the beginning.

Another convenient thing to remember is that the programming of an inductive procedure is also always backwards. To illustrate this, we consider another example: "Determine the perfect part P of a closed set C". (Recall that a set is "perfect" if it is closed and has no isolated points. The perfect part of a closed set is the same as the set of condensation points, that is, the points in C which have uncountably many elements of C in any neighborhood.) Since we don't allow direct computations with real numbers, let's consider the closed set C as "given" by the relation:

$$R(p,q) \iff (p,q) \text{ is a rational interval in the complement of } C$$

which may be substituted for the $R(x)$ in statements of the third kind (i.e., "**if** $R(x)$ **go to** l_4" can be replaced with "**if** $R(p,q)$ **go to** l_4"). Let P' denote the collection of rational intervals contained in the complement of P. To "produce" the perfect part, P, we mean that we have a program which "accepts" exactly when the input is in P'.

The simplest way to get to the perfect part is through the following induction, which is essentially the same as that of Cantor-Bendixson (see for example [14]): Start with $G^0 =$ the set of rational intervals in the complement of C. Then let

$$(*) \qquad (p,q) \in G^{\alpha+1} \iff (\forall\varepsilon)(\exists r,s)\ 0 < s - r < \varepsilon \\ \& \big(p = r \text{ or } (p,r) \in G^{\alpha}\big) \\ \& \big(s = q \text{ or } (s,q) \in G^{\alpha}\big).$$

If λ is a limit ordinal then $G^{\lambda} = \bigcup_{\alpha<\lambda} G^{\alpha}$. Since every time G is increased it will contain a new rational interval (this is sometimes called the Cantor-Baire stationary principle), there must be some countable ordinal γ where $G^{\gamma+1} = G^{\gamma}$ at which point it is not hard to check that $G^{\gamma} = P'$.

To construct a program for this process, first note that the definition $(*)$ "arithmetically" defines $G^{\alpha+1}$ from G^{α} (i.e., all variables represent integers, rationals, etc., and all relations are computable with an ordinary computer, with the exception of membership in G^{α}). Notice also that the references to G^{α} are "positive" since the complement of G^{α} is not mentioned. Because of this, the induction is "monotone", which means that an increase in G^{α} would cause $G^{\alpha+1}$ to either increase or remain the same. Monotonicity is extremely helpful in programming the induction as we will explain below.

Notice that we can't simply start with G^0 then compute G^1, G^2, etc. Such a process would require both an infinite amount of storage and a computation tree with transfinite branches, neither of which are allowed in our strict notion of constructiveness. Instead, we program it "backwards" using methods outlined in [9]. First, reword $(*)$ as follows:

$$(**) \quad (p,q) \in G \iff (\forall\varepsilon)(\exists r,s)\ 0 < s - r < \varepsilon \\ \& \big(\text{either } p = r \text{ or } (p,r) \text{ is already known to be in } G\big) \\ \& \big(\text{either } s = q \text{ or } (s,q) \text{ is already known to be in } G\big).$$

We now write a program with references to the "old" G (see statements l_2 and l_4) represented by a loop back to the first statement. The role of the variable n is simply to insure that both (p,r) and (s,q) get checked for membership in the "old" G.

l_1: **if** $R(p,q)$ **accept**
 $\varepsilon \leftarrow \forall$, $r \leftarrow \exists$, $s \leftarrow \exists$
 if $s - r \geq \varepsilon$ **or** $s \leq r$ **reject**
 $n \leftarrow \forall$
 if $n = 1$ **go to** l_3
 if $p = r$ **accept**
 $q \leftarrow r$
l_2: **go to** l_4
l_3: **if** $s = q$ **accept**
 $p \leftarrow s$

l_4: **go to** l_1

We would like to now check that this program does what it is supposed to do, i.e., it accepts (p, q) if and only if (p, q) is in P'. However, since we will be using this same technique repeatedly, let us talk in a little more generality. Note that in the program above, we combined the two loops into one. That is, in l_2 we didn't say directly to go back to l_1, we saved that for the end.

Therefore, let $\mathcal{P}$ be a program in the language IND, with a finite sequence $\bar{x}$ of input variables, and with an initial statement labeled say "l_1" and with final statement "**go to** l_1". Let P'' be the collection of $\bar{x}$ which are accepted by $\mathcal{P}$. Suppose we change the last statement of $\mathcal{P}$ to the "illegal" statement "**if** $\bar{x} \in A$ **accept**" where A is a set variable and where this command has the obvious meaning. Let's call this "illegal" program $\mathcal{P}_A$. Define $\mathcal{Q}(A) = \{ \bar{x} \mid \mathcal{P}_A \text{ accepts } \bar{x} \}$. Then let $Q^0 = \emptyset$, $Q^{\alpha+1} = \mathcal{Q}(Q^\alpha)$, and when λ is a limit ordinal, $Q^\lambda = \bigcup_{\alpha<\lambda} Q^\alpha$. Note that $\mathcal{Q}$ is a "monotone" operator meaning that if A is increased, $\mathcal{Q}(A)$ must increase or remain the same. Therefore at some (countable) ordinal γ we will have $Q^{\gamma+1} = Q^\gamma$. It is immediate from the definitions that $\mathcal{Q}(P'') = P''$ and that $\mathcal{Q}(Q^\gamma) = Q^\gamma$.

PROPOSITION 1. *If P'' and Q^γ are as above then $P'' = Q^\gamma$.*

PROOF. Let α be the smallest ordinal (if there is one) such that there is some $\bar{x}$ in $Q^\alpha \setminus P''$. By its nature, α must be a successor ordinal. Then $\bar{x} \in \mathcal{Q}(Q^{\alpha-1})$. But $Q^{\alpha-1} \subset P''$ by hypothesis. Therefore, by monotonicity, $\bar{x} \in \mathcal{Q}(P'') = P''$ which is a contradiction.

In the other direction, let α be the least amount of time it takes for $\mathcal{P}$ to accept some $\bar{x}$ which is not in Q^γ (using some backwards notion of time, explained above). Let $A = \{ \bar{y} \mid \mathcal{P} \text{ accepts } \bar{y} \text{ in time less than } \alpha \}$. Then $\bar{x} \in \mathcal{Q}(A)$ and by the induction hypothesis, $A \subset Q^\gamma$. But then by monotonicity, $\bar{x} \in \mathcal{Q}(Q^\gamma) = Q^\gamma$ which is a contradiction. ⊣

The above proposition can be used to check that a program works. Take, for example, our perfect set program given above. Observe that if we replaced the last statement "l_4: **go to** l_1" with the illegal statement "l_4: **if** $(p, q) \in G$ **accept**" then the program becomes an exact translation of $(**)$. Therefore, Q^γ is the same as G^γ and so $P'' = P'$.

In the above program, we have shown that $\mathcal{P}$ accepts all the right inputs and only the right inputs. The algorithm, however, still has a sort of defect. If we input a rational interval which is not in P' then the program will never report this to us, but instead will go into an infinite loop (check this). We might say then, that the perfect part P is really only "half-calculated". In such a situation, the operation is still referred to as "inductive". When we are assured the benefit of an answer for all inputs then it is called "hyperarithmetical". This is analogous to the "semi-recursive" vs. "recursive" relations in ordinary computability theory. It is because of this phenomenon that the use of monotone induction becomes so vital: Suppose for example, in defining $G^{\alpha+1}$ we referred to membership in the complement of G^{α}. Then the program would need a loop which referred back to this complement. This, in turn, would cause all inputs (even those in P') to enter the infinite loop.

Note however, that even though a program may halt on all inputs, this does not mean that it is totally free of infinite loops. This is because we say that a statement of the form "$y \leftarrow \exists$" is accepted if just one value of y leads to acceptance, even though others may lead to an infinite loop. The idea here is that the infinite loops never need to be completely run as long as we think of the "infinite mind" as having the ability to check all the values of y simultaneously.

Harel and Kozen analyzed which relations can be computed by a program in the language IND. Recall that a set of integers is "arithmetical" if and only if it can be defined in the language of arithmetic (using only $+$, $-$, $\times$, $\div$, exp, 1, $=$) (or equivalently, using only recursive relations) and using the usual logical symbols where quantifiers are over integers. It is not hard to check (see [9]) that any arithmetical relation can be programmed in the IND language and in fact can be programmed without using any "loops" (i.e., "**go to**" statements are always forward). For example, suppose $B(x)$ and $C(x)$ are computable from programs starting with labels l_2 and l_3 respectively. Then putting the following three statements before these programs yields a program for the relation $A(x) \leftrightarrow B(x)\,\&\,C(x)$:

l_1: $m \leftarrow \forall$
  if $m = 1$ **go to** l_2
  go to l_3

Harel and Kozen prove that the relations which can be programmed in an "inductive" way as defined above, are exactly the ones which can be defined using a transfinite induction $G = \bigcup G^{\alpha}$ where at limit ordinals we take unions and where $G^{\alpha+1}$ is defined "arithmetically" and "positively" from G^{α} as in the previous example. (Work of Kleene classifies these inductive sets as the "Π^1_1 sets", i.e., sets which can be defined using an arithmetical formula preceded by one "$\forall$" quantifier over infinite sequences of integers, so that the hyperarithmetical sets are the same as the "Δ^1_1" sets, i.e., sets which are both Π^1_1 and whose complements are Π^1_1.)

It has long been known that the classes of "inductive" and "hyperarithmetical" sets reflect this kind of algorithmic behavior. This may, in fact, have been Kleene's motivation in introducing them [10]. According to Rogers [17] this behavior has even been used by certain constructivists to claim that the "hyperarithmetical" sets are in some sense the only ones which really exist!

The work of Harel and Kozen makes more explicit the constructive nature of these classes. It also provides an intuitive and philosophically direct method for showing that a relation is in one of these classes—just program it. The naturalness of these classes is also reflected in the many closure properties they enjoy, properties which one would naturally associate with a notion of "computable" (see Rogers [17]).

§**2. Given a derivative.** Before we start our "antiderivative" program we must face the controversial issue of how the function f will be "given" to us. One natural way (and this is the way we will consider) is in terms of the inverse images of rational open intervals. For a derivative, each such inverse image is a countable union of closed sets, each of which may be represented by the collection of (non-degenerate) rational intervals in its complement. This leads us to the following relation:

$$\mathrm{INVIM}(a, b, x, y, n) \iff (x, y) \text{ is in the complement of the } n\text{th closed set in the inverse image of the function on } (a, b).$$

We will therefore consider our derivative as "given" by allowing this relation to be a replacement for $R(x)$ in statements of the type "If $R(x)$ go to l_n". Similarly, we will say that the integral will be "produced" by a program which can compute the relation:

$$\begin{aligned} \mathrm{INT}(p, q, r, n) \iff & n = 1 \,\&\, r > F(q) - F(p) \\ & \text{or } n = 2 \,\&\, r < F(q) - F(p), \\ & \quad \text{where } F \text{ is any antiderivative of } f. \end{aligned}$$

The reader may now have a valid complaint. Although the relation INT seems non-controversial, perhaps the relation INVIM is giving away too much information about the function f. For example, consider that in Calculus we usually represent a function as a "machine" or a "black box" which when a value of x is inserted the machine spits out the value of $f(x)$. This sort of representation emphasizes that the inner workings of a function may not be known, and indeed, there may not be any rule or reason as to why a particular function does what it does. It should seem natural then to use a similar idea to represent our derivative. Why then don't we try to use a "black box" instead of the more revealing "inverse images of intervals"? Because doing so would render impossible any sort of countable constructive procedure for calculating the antiderivative. To see this, note that in any such procedure, only countably many values of x would ever be inserted into the "black box". Furthermore, which values to plug in would completely depend on the results of previous inquiries. Therefore, for each derivative there would be a countable set which would completely determine its antiderivative. But this is impossible! For any derivative and any countable set there is another derivative which has exactly the same values on the specified countable set. To see this, just add a derivative which is zero on the given set but not identically zero (such a derivative can be constructed directly, or its existence can be inferred from more general work of Zahorski, see [2], [19]). Nevertheless, classifying the operation of obtaining antiderivatives from a "black box" may still be interesting to study from the standpoint of recursion in higher types.

The use of inverse images can also be partially justified by the fact that in classical calculus, the integrands are really given by a formula. Of course it's not true that every derivative has such a formula, but the idea here is that the function should be presented in a manner which gives insight into how the function values are determined, rather than just as individual function values. The relation INVIM then seems natural because it provides an overall picture of the structure of a function, but in a way which can be used to represent any derivative. It may even seem reasonable to take the opposite point of view. Perhaps the derivative should be given to us in the strongest possible (nontrivial) way! Suppose for example, that method A gives away more information than method B and we know how to compute antiderivatives from method A, then the problem of computing antiderivatives from method B is reduced to the problem of how to get from B to A, which is no longer an integration problem.

Finally, it should be remarked that Dougherty and Kechris [8] consider a derivative given in a slightly different way; as an infinite sequence of continuous functions. We instead use inverse images because it seems more natural and more in line with what Denjoy was doing. Nevertheless, IND calculations with our "Borel code" or with their "Baire code" are completely equivalent. We skip the details. Other methods are also possible. Darji, Evans, and O'Malley have just recently given another very natural way to be "given" a derivative using what they call "first return continuity" (see [5]). It seems likely that their method, as well as others, will turn out to be computationally equivalent. But it is by no means certain that this will be true for every method of representing a derivative conceived in the future.

In the next section we will start our IND program for computing antiderivatives, proving that the operation of antidifferentiation is "inductive". Obtaining a "hyperarithmetical" program (which halts on *all* inputs) is impossible by a result of Dougherty and Kechris [8] who proved that the operation is not Borel. Nevertheless, it will be easy to get a program which halts on all inputs at least when the function given to us is guaranteed to be a derivative. The reason is that in this case, $\mathrm{INT}(p, q, r, 1)$ fails exactly when $\mathrm{INT}(p, q, s, 2)$ holds for every $s < r$, and $\mathrm{INT}(p, q, r, 2)$ fails exactly when $\mathrm{INT}(p, q, s, 1)$ holds for every $s > r$. This allows us to convert a program for $\mathrm{INT}(p, q, r, n)$ into a program for $\neg\mathrm{INT}(p, q, r, n)$. Then running the programs simultaneously, (see [9] for technical details of how to create a single IND program to combine two others) we have a computation which will halt on all derivatives. This level of complexity is what Dougherty and Kechris call "Δ^1_1 on the set of [codes of] derivatives" since although it is not strictly "hyperarithmetical", it always halts as long as we are given the code of a derivative. In other words, since the primitive problem says "Given a derivative ..." we have to assume that they are not tricking us by giving us something which may look like a derivative but really isn't.

§3. Recover the primitive (part 1). We are now ready to start the antiderivative program. We begin with the relation:

$$
\begin{aligned}
(1)\qquad & \text{“Cover}(\bar{p},\bar{q},s,t)\text{”} \\
& \iff \text{“The sequences } (\bar{p},\bar{q}) \text{ of rational intervals} \\
& \qquad \text{form a sequential cover of the rational} \\
& \qquad \text{interval } [s,t]\text{”} \\
& \iff (\exists k) l(\bar{p}) = l(\bar{q}) = k \mathbin{\&} (\forall i \le k) p_i < q_i \\
& \qquad\qquad \mathbin{\&} (\forall i < k) p_{i+1} < q_i \mathbin{\&} p_1 < s < t < q_k.
\end{aligned}
$$

Since the relation is arithmetical, it is easy to program both it and its complement. We give the programs here. (From this point on, we will be building our antiderivative program in blocks. Each block will end in an "**accept**", "**reject**", or an unconditional "**go to**" statement. It therefore won't matter which order we assemble the blocks except for the first one, which will begin with the label "l_1" and will be the start of the entire program. This first block will actually be the last one presented. This is why we start here with the label "l_2".)

l_2: $k \leftarrow \exists$
  if $l(\bar{p}) \neq k$ or $l(\bar{q}) \neq k$ **reject**
  $i \leftarrow \forall$
  if $i > k$ **accept**
  if $p_i \geq q_i$ **reject**
  if $i = k$ **accept**
  if $p_{i+1} \geq q_i$ **reject**
  if $p_1 < s < t < q_k$ **accept**
  reject

When a program halts on all inputs, then its complement can be programmed by merely interchanging "$\exists$" with "$\forall$" and also "**accept**" with "**reject**". Thus the following is a program for "$\neg$ Cover$(\bar{p},\bar{q},s,t)$":

l_3: $k \leftarrow \forall$
  if $l(\bar{p}) \neq k$ or $l(\bar{q}) \neq k$ **accept**

$i \leftarrow \exists$
if $i > k$ **reject**
if $p_i \geq q_i$ **accept**
if $i = k$ **reject**
if $p_{i+1} \geq q_i$ **accept**
if $p_1 < s < t < q_k$ **reject**
accept

Our program will involve two transfinite inductions. The first will be monotone and will serve to organize the reals into a chain of open sets, with the property that f is bounded on each "new part" of the chain. The second induction will be monotone with respect to the first and will calculate the actual integrals. Each induction will have two stages. We now describe the first stage of the first induction. Let S be a collection of rational intervals. We define

$$(2) \quad (x, y) \in U(S) \iff (\forall s, t)[s, t] \subset (x, y) \rightarrow \\ [(\exists \bar{p}, \bar{q})\text{"Cover}(\bar{p}, \bar{q}, s, t)\text{"} \,\&\, (\forall i \leq l(\bar{p}))(p_i, q_i) \in S].$$

The operation $U(S)$ merely closes S under unions and subintervals, using compactness. The definition is "positive arithmetical" in S and so it is easy to write a program for $U(S)$ where we use the label l_8 to denote the start of a program (not yet given), which accepts an interval (a, b) if and only if $(a, b) \in S$.

l_4: $s \leftarrow \forall$, $t \leftarrow \forall$
if $\neg(s < t \,\&\, [s, t] \subset (x, y))$ **accept**
$\bar{p} \leftarrow \exists$, $\bar{q} \leftarrow \exists$, $m \leftarrow \forall$
if $m = 1$ **go to** l_2
l_5: $i \leftarrow \forall$
if $i > l(\bar{p})$ **accept**
$a \leftarrow p_i$, $b \leftarrow q_i$
go to l_8

Let T be another collection of rational intervals. We now give a program which accepts (x, y) if and only if $(x, y) \notin U(T)$. Here, l_8 is the start of a program (not yet given) which accepts (c, d) if and only if $(c, d) \notin T$.

l_6: $s \leftarrow \exists$, $t \leftarrow \exists$
if $\neg(s < t \,\&\, [s, t] \subset (x, y))$ **reject**
$\bar{p} \leftarrow \forall$, $\bar{q} \leftarrow \forall$, $m \leftarrow \exists$
if $m = 1$ **go to** l_3
l_7: $i \leftarrow \exists$
if $i > l(\bar{p})$ **reject**
$c \leftarrow p_i$, $d \leftarrow q_i$
go to l_8

One may wonder at this point why we are using the same label l_8 to represent both that $(a,b) \in S$ and that $(c,d) \notin T$. Actually, the program starting at l_8 is going to calculate one or the other, depending on an additional parameter $n \in \{1,2\}$. The parameter n will always be 1 when l_4 is reached and will always be 2 when l_6 is reached.

We now proceed to the second stage of the first induction. We define

$$(3)\quad (a,b) \in \mathcal{B}(S) \iff (\exists M)(\exists n)(\forall x)(\forall y) \\ \big[\mathrm{INVIM}(-M,M,x,y,n)\,\&\,(x,y) \subset (a,b)\big] \to (x,y) \in U(S).$$

Note that the operation $\mathcal{B}$ adds to S rational intervals in which $\bigcup S$ covers the complement of one of the closed sets which make up the inverse image of $(-M,M)$ for some M. Thus if $(a,b) \in \mathcal{B}(S)$ then the range of f on $(a,b) \setminus \bigcup S$ is bounded. The operator $\mathcal{B}$ is monotone since $U(S)$ is and membership in $U(S)$ is only mentioned in a "positive" way. It can be iterated in the usual manner: $B^0 = \emptyset$, $B^{\alpha+1} = \mathcal{B}(B^\alpha)$, and if λ is a limit ordinal then $B^\lambda = \bigcup_{\alpha<\lambda} B^\alpha$.

Suppose that S does not contain all rational intervals. Then $\mathcal{B}(S)$ is strictly larger than S. To see this, notice that $\mathbb{R}$ is the union of all of the closed sets making up the inverse images of the intervals $(-M,M)$. Let $C = \mathbb{R} \setminus \bigcup S$. The Baire Category Theorem says that in some nontrivial portion, $(a,b)\cap C$, of C one of these closed sets is dense. But then, since the set is closed, it must contain $(a,b)\cap C$. Therefore, its complement in (a,b) must be covered by $\bigcup S$, and so $(a,b) \in \mathcal{B}(S)$. A similar argument shows that $\bigcup \mathcal{B}(S)$ is dense.

Therefore, $\mathcal{B}$ cannot become stationary until it contains everything, so at some countable ordinal γ, B^γ must be the entire collection of rational intervals. For this reason, it won't be helpful to program membership in B^γ. It will be more useful to have a program to tell in which order the intervals are admitted to B^γ. This is an instance of the Stage Comparison Theorem of Moschovakis, see ([13], p. 27) where the argument for it used below is credited to Kunen.

For each interval (p,q) let $\mathrm{ord}(p,q)$ be the least ordinal α such that $(p,q) \in B^\alpha$. By its nature, $\mathrm{ord}(p,q)$ is a successor ordinal. We abbreviate $(a,b) \le (c,d)$ if and only if $\mathrm{ord}(a,b) \le \mathrm{ord}(c,d)$ and $(a,b) < (c,d)$ if and only if $\mathrm{ord}(a,b) < \mathrm{ord}(c,d)$. To program these relations, notice first that

$$(4)\qquad (a,b) \le (c,d) \iff (a,b) \in \mathcal{B}\big(\{\,(a',b') \mid (a',b') < (c,d)\,\}\big) \\ \text{and } (a,b) < (c,d) \iff (c,d) \notin \mathcal{B}\big(\{\,(c',d') \mid \neg(a,b) \le (c',d')\,\}\big).$$

Both of the above equivalences hold because of the monotonicity of $\mathcal{B}$. The second is just the negation of the first with the roles of (a,b) and (c,d)

interchanged. Define the relation

$$(5)\qquad D(a,b,c,d,n) \iff n=1 \text{ and } (a,b)<(c,d) \text{ or } n=2 \text{ and } (a,b)\leq(c,d).$$

Using (4), the relation D can be built by transfinite induction using the following operator. Let A be a subset of $\mathbb{Q}^4\times\{1,2\}$. Then let

$$(6)\quad (a,b,c,d,n)\in\mathcal{D}(A) \iff \\ n=1 \text{ and } (c,d)\notin\mathcal{B}\big(\{\,(c',d')\mid (a,b,c',d',2)\notin A\,\}\big) \\ \text{or } n=2 \text{ and } (a,b)\in\mathcal{B}\big(\{\,(a',b')\mid (a',b',c,d,1)\in A\,\}\big).$$

Note that because of the double negation in the first half of the right side, the monotonicity of $\mathcal{D}$ follows from the monotonicity of $\mathcal{B}$. As before, let $D^0=\emptyset$, $D^{\alpha+1}=\mathcal{D}(D^\alpha)$, and for limit ordinals λ, $D^\lambda=\bigcup_{\alpha<\lambda}D^\alpha$. Again, this forms an increasing sequence of sets which becomes stationary at some countable ordinal γ .

Proposition 2 (Moschovakis). *If D and D^γ are as above then $D=D^\gamma$.*

Proof. Let α be the smallest ordinal such that there exists an (a,b,c,d,n) in D where $\operatorname{ord}(a,b)=\alpha$ and $(a,b,c,d,n)\notin D^\gamma$. Let $n=2$ if possible and otherwise let $n=1$.

Case 1. $n=1$.

Since $(a,b,c,d,1)\in D$, we have that $(a,b)<(c,d)$. By induction hypothesis we have that $(a,b,c',d',2)\in D^\gamma$ for every $(c',d')\geq(a,b)$. But

$$\begin{aligned}(c,d)\notin\mathcal{B}(\{\,(c',d')\mid (c',d')<(c,d)\,\}) \\ =\mathcal{B}(\{\,(c',d')\mid \neg(c',d')\geq(a,b)\,\}) \\ \supset\mathcal{B}(\{\,(c',d')\mid (a,b,c',d',2)\notin D^\gamma\,\})\end{aligned}$$

by monotonicity. Therefore by (6), $(a,b,c,d,1)\in\mathcal{D}(D^\gamma)=D^{\gamma+1}=D^\gamma$ which is a contradiction.

Case 2. $n=2$.

Since $(a,b,c,d,2)\in D$, we have that $(a,b)\leq(c,d)$. By induction hypothesis, we have that $(a',b',c,d,1)\in D^\gamma$ for every $(a',b')<(a,b)$. But

$$\begin{aligned}(a,b)\in\mathcal{B}(\{\,(a',b')\mid (a',b')<(a,b)\,\}) \\ \subset\mathcal{B}(\{\,(a',b')\mid (a',b',c,d,1)\in D^\gamma\,\})\end{aligned}$$

by monotonicity. Therefore by (6), $(a,b,c,d,2)\in\mathcal{D}(D^\gamma)=D^{\gamma+1}=D^\gamma$ which is a contradiction.

For the other direction, (4) says that $D = \mathcal{D}(D)$, then induction on α using the monotonicity of $\mathcal{D}$ gives $D^\alpha \subset D$ for all α. ⊣

Using the technique already displayed in a previous example and taken from Harel and Kozen [9], we can now transform (3) and (6) into a program which accepts (a, b, c, d, n) if and only if $D(a, b, c, d, n)$. The reason the program does what it is supposed to follows as before, using Propositions 1 and 2:

l_8: **if** $n = 1$ **go to** l_9
$M \leftarrow \exists, n \leftarrow \exists, x \leftarrow \forall, y \leftarrow \forall$
if $\neg$INVIM$(-M, M, x, y, n)$ **or** $\neg(x, y) \subset (a, b)$ **accept**
$n \leftarrow 1$
go to l_4
l_9: $M \leftarrow \forall, n \leftarrow \forall, x \leftarrow \exists, y \leftarrow \exists$
if $\neg$INVIM$(-M, M, x, y, n)$ **or** $\neg(x, y) \subset (c, d)$ **reject**
$n \leftarrow 2$
go to l_6

§4. **Intermission.** In the previous section we showed how to generate a transfinite sequence of collections of finite open rational intervals, $\{B^\alpha\}$, with the properties:

(1) If $I \in B^\alpha$ and J is an open rational subinterval of I then $J \in B^\alpha$,
(2) every finite rational open interval in $\bigcup B^\alpha$ is in $B^{\alpha+1}$,
(3) If $I \in B^{\alpha+1}$ then $f(x)$ is bounded on $I \setminus \bigcup B^\alpha$,
(4) $B^\lambda = \bigcup_{\alpha<\lambda} B^\alpha$ if λ is a limit ordinal,
(5) $\bigcup_\alpha B^\alpha$ contains all open rational intervals,
(6) B^0 is empty and for $\alpha > 0$, $\bigcup B^\alpha$ is dense.

We also showed how to effectively compute the relations $\mathrm{ord}(I) < \mathrm{ord}(J)$ (meaning I got into the sequence before$J$) and $\mathrm{ord}(I) \leq \mathrm{ord}(J)$.

Before we continue with the second half of the program, where the integrals are actually calculated, we pause to discuss the overall plan of totalization, why it works, and how we plan to make it programmable.

Notation. We denote by $|G|$ the Lebesgue measure of the set G. All of our intervals are finite. For an interval $J = (a, b)$, $(a, b]$, $[a, b)$, or $[a, b]$ we let $F(J)$ denote the integral $F(b) - F(a)$. If $G = \bigcup G_i$ is a countable union of disjoint intervals, then denote by $F(G)$ the sum $\sum F(G_i)$, but we only do this when the sum is known to be absolutely convergent, since in that case, the order of the summation is irrelevant. The relative integral is then defined to be $F(G)/|G|$. If S denotes a bounded measurable set on which f is bounded, then the Lebesgue integral of f over S exists and we call it $L(S)$, and we call $L(S)/|S|$ the relative Lebesgue integral of S.

Suppose now that we have a subrelation $A \subset \mathrm{INT}$. Recall that $\mathrm{INT} = \{\,(p,q,r,n) \mid p < q$ are rational and either $n = 1$ and $r > F(q) - F(p)$ or $n = 2$ and $r < F(q) - F(p)\,\}$ which is the relation we are trying to determine. We denote by D the domain of A so that $D = \{\,(p,q) \mid (\exists n,r)(p,q,r,n) \in A\,\}$ and suppose that the following properties hold:

(i) If $I \in D$ and J is an open rational subinterval of I then $J \in D$.
(ii) If $I \in D$ then $(p,q,r,n) \in A$ if and only if either $n = 1$ and $r > F(I)$ or $n = 2$ and $r < F(I)$.

These properties obviously hold for our first such A, which is the empty set. Our goal is to find a way to expand any such relation, whose domain D is not yet all rational intervals, taking care to preserve the above properties. Once this is established, we can iterate the process knowing that at some closure ordinal, we will arrive at the relation INT. At the limit stages of this iteration we will simply take unions of the previously defined A's. It is easy to see that if the properties (i) and (ii) hold at each previous stage, they will also hold at limit stages. So we only need to concentrate on the successor stages. There are three methods which will help us to add a new rational open interval to D.

The first approach is very easy. If we "know" the integrals on (a_1,a_2), (a_2,a_3), …, (a_{n-1},a_n) then we may find the integral on (a_1,a_n) by simply adding these together. This fact is immediate from the definition.

The second approach is the improper integration technique of Cauchy. If the sequence of open intervals $\{I_n\}$ converges to I, then $F(I) = \lim F(I_n)$. This, of course, is an immediate consequence of the continuity of F.

The third approach is due to Lebesgue. If G is an open subset of I then $F(I) = F(G) + L(I \setminus G)$. This presupposes of course that these terms are well defined, so in particular the Lebesgue integral of f over $I \setminus G$ must exist and the sum $F(G)$ must be absolutely convergent. This equivalence is an elementary consequence of Lebesgue integration and was proved by Lebesgue in 1904 (see [16]). We will refer to it as Lebesgue's Theorem. It will be applied as follows: If $B^\alpha \subset D$ and $I \in B^{\alpha+1}$ then (using (3)) f is bounded on $I \setminus \bigcup D$ so the Lebesgue integral on this set exists. If we are lucky enough to also have a bound, say M, for the relative integrals of the components, J_i, of $I \cap (\bigcup D)$, then $\sum |F(J_i)|$ is no bigger than $M \cdot |I|$. Therefore, in this case, we may apply Lebesgue's Theorem to calculate $F(I) = F\big(I \cap (\bigcup D)\big) + L\big(I \setminus (\bigcup D)\big)$.

Later accounts of totalization usually combine the first and third approaches (e.g., Saks [18], Natanson [15]). Indeed, the first approach is actually a special case of the third one. However, it will be more convenient for us to treat them separately. Since our purpose is to write a program for this "totalization" process, the first step is to carefully define each of the three extension techniques using only arithmetical quantifiers, and using A

only in a positive way. We start here now the first one:

$$(7) \qquad (p, q, r, n) \in \mathcal{H}(A) \iff$$
$$\text{there is a partition, } (a_1, \ldots, a_j) \text{ of } (p, q) \text{ such that for each } i < j, \text{ there is a rational } r_i \text{ with } (a_i, a_{i+1}, r_i, n) \in A \text{ and either } n = 1 \text{ and } \sum r_i < r \text{ or } n = 2 \text{ and } \sum r_i > r.$$

The operator $\mathcal{H}$ is easily defined in an arithmetical way and since it only mentions A in a positive way, it will be easy to program. Notice that $\mathcal{H}(A)$ contains A, and that the properties (i) and (ii) still hold, where of course, D refers to the new domain, the domain of $\mathcal{H}(A)$. In fact, a slightly stronger form of (i) now holds:

(i′) Any open rational interval which is covered by finitely many elements of D, is in D.

We now assume that (i′) holds and apply the next operator $\mathcal{C}(A)$ defined as follows:

$$(8) \qquad (p, q, r, n) \in \mathcal{C}(A) \iff$$
$$(\exists s, t)(\forall \varepsilon) \text{ there is a rational } u, \text{ and a rational subinterval } (a, b) \text{ of } (p, q) \text{ such that } a - p + q - b < \varepsilon, \; s < r < t, \; (a, b, u, n) \in A, \text{ and either } n = 1 \text{ and } u < s \text{ or } n = 2 \text{ and } u > t.$$

Once again this is easily arithmetical and monotone in A, so it will be easily programmed. It is also readily checked that $\mathcal{C}(A)$ contains A and that the properties (i′) and (ii) still hold when A is expanded to $\mathcal{C}(A)$. In fact, a stronger form of (i′) now holds:

(i″) Any rational interval in $\bigcup D$ is also in D.

Obtaining a suitable definition for the Lebesgue extension will be trickier. The problem is that we have to find a way to talk about components of $\bigcup D$ without mentioning D in a negative way. The solution is to take advantage of the fact that we have already shown how to compute membership in B^α. So when we are tempted to mention D in a negative way, we will try to get by with using B^α instead. To make things easier, we will split this into two propositions. The first of these, Proposition 3, shows how to define the new domain and the second, Proposition 4, tells how to compute the integrals on this new domain. Since we will only apply this third technique after the operators $\mathcal{H}$ and $\mathcal{C}$ have been applied, we can assume that the property (i″) holds. We will first present an easy but useful lemma:

LEMMA. *Let* $\{x_m^n\}$ *be a bounded collection of real numbers, defined whenever* $m \le n$ *are positive integers. Then for some increasing function* $g : \mathbb{Z}^+ \to \mathbb{Z}^+$ *we have for each* m *that* $\lim_{n\to\infty} x_m^{g(n)}$ *exists.*

PROOF. Let $g_0(n) = n$. Given g_{m-1}, let g_m be a subsequence of g_{m-1} such that $\lim_{n\to\infty} x_m^{g_m(n)}$ exists and g_m, g_{m-1} agree on the first m entries. Since g_n is a subsequence of g_0, it is increasing. Also, for each n, $g_m(n)$ is eventually constant and so $\lim_{m\to\infty} g_m(n)$ exists. Call this limit $g(n)$. Then for each m, $g(n)$ is a subsequence of g_m. Therefore, $g(n)$ is also increasing and since $\lim_{n\to\infty} x_m^{g_m(n)}$ exists, so does $\lim_{n\to\infty} x_m^{g(n)}$. $\dashv$

PROPOSITION 3. *Suppose* (i$''$) *and* (ii) *hold,* $D \subset B^\alpha$*, and let* $I \in B^{\alpha+1}$*. Then the following are equivalent*:

(3a) *There is a bound for the relative integrals of the components of* $I \cap (\bigcup D)$.

(3b) *For some finite* M*, every finite collection of rational intervals in* $I \cap (\bigcup B^\alpha)$ *can be covered by a finite collection of disjoint intervals in* $I \cap (\bigcup D)$ *whose relative integrals are in* $(-M, M)$.

PROOF. (3a) $\to$ (3b): Suppose every component of $I \cap (\bigcup D)$ has a relative integral between $-M$ and M. Since D contains all the intervals in B^α, any finite collection of rational intervals in $I \cap (\bigcup B^\alpha)$ is covered by a finite collection of components of $I \cap (\bigcup D)$. Each such component can be reduced, if necessary, to a rational subinterval which will then be in D, preserving the property that each has a relative integral between M and $-M$, and also the fact that the original collection of rational intervals remains covered.

(3b) $\to$ (3a): Suppose I satisfies property (3b) for some M. Let $M' > M$ so that the range of f on $I \setminus \bigcup B^\alpha$ is also bounded by M' (this is possible by the definition of $B^{\alpha+1}$). Now let J be a component of $I \cap (\bigcup D)$. By ignoring the (two) end components (this won't affect boundedness), we may assume that J is a component of $\bigcup D$. We will finish the proof by showing that the relative integral of J is in $(-M', M')$. By property (3b), any finite number of rational subintervals of $J \cap (\bigcup B^\alpha)$ can be covered by a finite number of disjoint intervals in D whose relative integrals are in $(-M, M)$. But since J is a component of $\bigcup D$ the intervals in such a cover can be assumed to be subintervals of J.

Let $K_1, K_2, \ldots$ be the components of $J \cap (\bigcup B^\alpha)$. Let R_m^n be a rational subinterval of K_m with $|K_m| - |R_m^n| < 1/n$. Let $S^n = \bigcup_{m\le n} R_m^n$, and C^n be a cover of S^n as promised by (3b). For each $m \le n$, let C_m^n denote the component of C^n which covers R_m^n. Using the lemma, there is an increasing sequence $g(n)$ such that for each m, the interval $Z_m = \lim_{n\to\infty} C_m^{g(n)}$ exists. Then Z_m must contain all of K_m and so $Z = \bigcup Z_m$ contains all of $J \cap (\bigcup B^\alpha)$. Furthermore, each Z_m has a relative integral in $[-M, M]$ since each $C_m^{g(n)}$ does. Finally, any two intervals Z_i and Z_j are either disjoint or equal: If there is a region of overlap, then there is a region of overlap for $C_i^{g(n)}$ and $C_j^{g(n)}$ for large enough n, causing $C_i^{g(n)} = C_j^{g(n)}$, which forces $Z_i = Z_j$. Therefore, the components of Z are the intervals Z_m, and these have bounded relative integrals, and so Lebesgue's Theorem applies and $F(J) = F(Z) + L(J \setminus Z)$. But the relative integral of Z is in $(-M', M')$ since $M' > M$. Also, by choice of M', the relative Lebesgue integral of $J \setminus Z$ is also in $(-M', M')$. Therefore, so is the relative integral of J. ⊣

PROPOSITION 4. *Suppose properties* (i″) *and* (ii) *hold,* $B^\alpha \subset D$, $I \in B^{\alpha+1}$, $r \in \mathbb{Q}$, *and* $n \in \{1, 2\}$. *If I satisfies the property from Proposition* 3*, then the following are equivalent*:

(4a) *Either* $r > F(I)$ *and* $n = 1$ *or* $r < F(I)$ *and* $n = 2$.

(4b) ($\exists$ *finite number* M) ($\exists s < r,\ t > r$)
($\exists$ *finite union* $A = \bigcup_i A_i$ *of rational subintervals of* I *in* $\bigcup B^\alpha$)
($\forall$ *finite union* $B = \bigcup_i B_i$ *of rational subintervals of* I *in* $\bigcup B^\alpha$ *with* $A \subset B$)
($\exists$ *finite union* $C = \bigcup_i C_i$ *of disjoint rational intervals in* $I \cap (\bigcup D)$ *with* $B \subset C$ *and with the relative integral of each* C_i *in* $(-M, M)$)
($\forall$ *finite union,* J, *of components of* C *with* $A \subset J$)
$F(J) + L(I \setminus J \setminus \bigcup B^\alpha) < s$ *and* $n = 1$ *or* $F(J) + L(I \setminus J \setminus \bigcup B^\alpha) > t$ *and* $n = 2$.

PROOF. Since both cases $n = 1$ and $n = 2$ are similar, we prove only the case $n = 1$. So that we don't have to keep saying it, all intervals (except those denoted by K or Z) are rational subintervals of I.

(4a) $\rightarrow$ (4b): First check the case $\alpha = 0$: M may be anything and s may be any rational between $F(I)$ and r. Then A and B will be empty and we let C also be empty forcing J to be empty. Then $F(J) = 0$, f is bounded (hence Lebesgue integrable) on I, $F(I) = L(I) = L(I \setminus J \setminus \bigcup B^\alpha)$ and (4b) follows.

Now let $\alpha > 0$ so that $\bigcup B^\alpha$ is dense. Let M be the bound from property (3a), and increase M so that it is also a bound for the range of f on $I \setminus \bigcup B^\alpha$ (this is possible by definition of $B^{\alpha+1}$), and also larger than $|f(x)|$ calculated at each endpoint of I. Let $r > F(I)$ and $s = (r + F(I))/2$. Choose $A = \bigcup_i A_i$ so that each A_i is in B^α and so that the components of $I \cap (\bigcup D)$ disjoint from A have total measure less than $(r - s)/4M$. Let $B = \bigcup_i B_i$ so that $A \subset B \subset \bigcup B^\alpha$. Choose $C = \bigcup_i C_i$ to be a rational approximation for the union of components of $I \cap (\bigcup D)$ which intersect B, so that:

(a) $B \subset C \subset \bigcup D$, with at most one C_i in each component of $\bigcup D$,
(b) $|I \cap (\bigcup D) \setminus C| < (r - s)/4M$,
(c) the relative integral of each C_i is in $(-M, M)$, and

(d) for each i, if Z_i is the component of $\bigcup D$ containing C_i, then the relative integral of each $Z_i \setminus C_i$ is in $(-M, M)$. This is possible, since for each component (a, b) of $\bigcup D$, there is a rational interval $(c, d) \subset (a, b)$ with $c - a$ and $b - d$ arbitrarily small, and with the relative integrals of (a, c) and (d, b) arbitrarily close to $f(a)$, $f(b)$ respectively. But both $f(a)$ and $f(b)$ are in $(-M, M)$.

Let J be a finite union of components of C, with $A \subset J$. Then:

$$\begin{aligned} &F(J) + L(I \setminus J \setminus \textstyle\bigcup B^\alpha) \\ &\qquad = F(J) + L(I \setminus C \setminus \textstyle\bigcup B^\alpha) + L(C \setminus J \setminus \textstyle\bigcup B^\alpha), \end{aligned}$$

since C contains J,

$$\begin{aligned} &= F(J) + F(I \setminus C) + F(C \setminus J) - F(I \setminus C) - F(C \setminus J) \\ &\qquad + L(I \setminus C \setminus \textstyle\bigcup B^\alpha) + L(C \setminus J \setminus \textstyle\bigcup B^\alpha) \\ &= F(I) - F(I \setminus C) - F(C \setminus J) + L(I \setminus C \setminus \textstyle\bigcup B^\alpha) \\ &\qquad + L(C \setminus J \setminus \textstyle\bigcup B^\alpha) \\ &= F(I) - F\big((I \setminus C) \cap (\textstyle\bigcup D)\big) - L\big(I \setminus C \setminus (\textstyle\bigcup D)\big) \\ &\qquad - F(C \setminus J) + L(I \setminus C \setminus \textstyle\bigcup B^\alpha) + L(C \setminus J \setminus \textstyle\bigcup B^\alpha), \end{aligned}$$

by Lebesgue's Theorem on $I \setminus C$,

$$\begin{aligned} &\le F(I) + \big|F\big((I \setminus C) \cap (\textstyle\bigcup D)\big)\big| \\ &\qquad + \big|L(I \setminus C \setminus \textstyle\bigcup B^\alpha) - L\big(I \setminus C \setminus (\textstyle\bigcup D)\big)\big| \\ &\qquad + |F(C \setminus J)| + |L(C \setminus J \setminus \textstyle\bigcup B^\alpha)| \\ &= F(I) + \big|F\big((I \setminus C) \cap (\textstyle\bigcup D)\big)\big| \\ &\qquad + \big|L\big(I \setminus C \setminus (\textstyle\bigcup B^\alpha) \cap (\textstyle\bigcup D)\big)\big| \\ &\qquad + |F(C \setminus J)| + |L(C \setminus J \setminus \textstyle\bigcup B^\alpha)|, \end{aligned}$$

since $B^\alpha \subset D$,

$$\begin{aligned} &< F(I) + M|(I \setminus C) \cap (\textstyle\bigcup D)| \\ &\qquad + M\big|I \setminus C \setminus (\textstyle\bigcup B^\alpha) \cap (\textstyle\bigcup D)\big| + M|C \setminus J| \\ &\qquad + M|C \setminus J \setminus \textstyle\bigcup B^\alpha|, \end{aligned}$$

by (d), (c) and the fact that the range of f on $I \setminus \bigcup B^\alpha$ is bounded by M,

$$\begin{aligned} &\le F(I) + 2M\big|(I \setminus C) \cap (\textstyle\bigcup D)\big| + 2M|C \setminus J| \\ &\le F(I) + \frac{r-s}{2} + \frac{r-s}{2}, \end{aligned}$$

by (b) and since any component of $C \setminus J$ is contained in a component of $\bigcup D$ which is disjoint from A,

$$= F(I) + r - s = s < r.$$

(4b) $\to$ (4a): Given M and $s < r$ which make (4b) hold, choose $M' > M$ such that the range of f on $I \setminus \bigcup B^\alpha$ is bounded by M' (possible by definition of $I \in B^{\alpha+1}$). Let $K_1, K_2, \ldots$ be an enumeration of the components of $I \cap (\bigcup B^\alpha)$. After $A = \bigcup_i A_i$ is given, let R^n_m be a rational subinterval of K_m, containing $A \cap K_m$, and such that $|K_m \setminus R^n_m| < 1/n$. Let $S^n = R^n_1 \cup R^n_2 \cup \cdots \cup R^n_n$. When n is large enough so that S^n contains A, let C^n be a cover of S^n as promised by (4b) with $B = S^n$, and let C^n_m be the component of C^n which contains R^n_m.

Using the lemma, for some sequence $g(n)$ and each m, the limit interval $Z_m = \lim_{n\to\infty} C^{g(n)}_m$ exists. As before, any two intervals Z_i and Z_j are either disjoint or equal. Also, for each m, Z_m contains K_m and has a relative integral in $[-M, M]$. Then $Z = \bigcup Z_m$ contains $I \cap (\bigcup B^\alpha)$ and it follows that Lebesgue's Theorem may be applied to get $F(I) = F(Z) + L(I \setminus Z)$.

Let $Z' = Z_1 \cup \cdots \cup Z_k$ be a finite collection of components of Z so that Z' contains A and $|Z \setminus Z'| < (r-s)/2M'$. It follows that $|F(Z) - F(Z')| < (r-s)/2$. Then let $J = \bigcup \{\, C^p_m \mid m \le k \,\}$ for some p in the range of g, large enough so that:

(1) $A \subset J$

(2) $\big|(Z \setminus J) \cup (J \setminus Z)\big| < (r-s)/2M'$

(3) $|F(Z) - F(J)| < (r-s)/2$.

Applying (4b) (with $C = C^p$) we get that$F(J) + L(I \setminus J \setminus \bigcup B^\alpha) < s$. Then,

$$\begin{aligned} F(I) &= F(Z) + L(I \setminus Z) \\ &= F(Z) + L(I \setminus Z \setminus \textstyle\bigcup B^\alpha), \end{aligned}$$

since Z contains $I \cap (\bigcup B^\alpha)$,

$$\begin{aligned} &= F(Z) + L(I \setminus J \setminus \textstyle\bigcup B^\alpha) - L(Z \setminus J \setminus \bigcup B^\alpha) \\ &\qquad + L(J \setminus Z \setminus \textstyle\bigcup B^\alpha) \\ &\le F(Z) + L(I \setminus J \setminus \textstyle\bigcup B^\alpha) \\ &\qquad + \big|L\big((Z \setminus J) \cup (J \setminus Z) \setminus \textstyle\bigcup B^\alpha\big)\big| \\ &\le F(Z) + L(I \setminus J \setminus \textstyle\bigcup B^\alpha) \\ &\qquad + M'\big|(Z \setminus J) \cup (J \setminus Z) \setminus \textstyle\bigcup B^\alpha\big|, \end{aligned}$$

by choice of M',

$$\begin{aligned}&\leq F(Z) + L(I \setminus J \setminus \bigcup B^{\alpha})\\&\qquad + M'\big|(Z \setminus J) \cup (J \setminus Z)\big|\\&< F(J) + \frac{r-s}{2} + L(I \setminus J \setminus \bigcup B^{\alpha}) + \frac{r-s}{2},\end{aligned}$$

by (2) and (3)

$$< s + (r-s) = r,$$

and this concludes the proof of Proposition 4. ⊣

We can now use these two propositions to define a Lebesgue extension operator. Since (4b) easily implies (3b) we only need to mention (4b):

(9) $(p, q, r, n) \in \mathcal{L}(A) \iff$
$(p, q, r, n) \in A$, or else $(p, q) \in B^{\alpha+1}$ for some α
with $B^{\alpha} \subset D$ and (4b) holds (with references to F
replaced by the corresponding references to A).

We see from the propositions that this definition correctly applies the third extension technique. We also see that the operator $\mathcal{L}$ is monotone in A, that $A \subset \mathcal{L}(A)$ and that although (i′) and (i″) may no longer hold after $\mathcal{L}$ is applied, certainly (i) and (ii) still hold. The process of iterating the operators can therefore continue. To carry out the totalization, we therefore define a new operator $\mathcal{I}(A) = \mathcal{L}(\mathcal{C}(\mathcal{H}(A)))$ and iterate this in the usual way. In the next section we will show how to program this induction. But for now, it is still incumbent upon us to show that the process succeeds in extending the domain of A. This argument is due to Denjoy.

Suppose $A \subset \mathrm{INT}$, $A \neq \mathrm{INT}$, and A satisfies (i) and (ii). It may be that none of the operators $\mathcal{H}$, $\mathcal{C}$, and $\mathcal{L}$ individually extend the domain D, but the combination $\mathcal{I} = \mathcal{L}(\mathcal{C}(\mathcal{H}(A)))$ certainly will. To see this, suppose $\mathcal{H}(A) = A$. Then (i′) holds. Suppose also that $\mathcal{C}(A) = A$. Then (i″) also holds. Let P be the complement of $\bigcup D$. If P were empty then A would be INT and we would be done. So P is a nonempty closed set. Let J be in $B^{\alpha+1}$ with α minimal such that $J \cap P$ is nonempty. We will show under these conditions that the operator $\mathcal{L}$ will add a new subinterval to D.

Now by the definition of derivative, given any $\varepsilon < 1$, for each $x \in P$ there is a maximum number $\delta(x) > 0$ such that the difference quotient of F (i.e., relative integral) over any interval of size less than $\delta(x)$ which contains x must be within ε of $f(x)$. By the Baire Category Theorem, there must be a rational interval $K \subset J$ and a number $\delta > 0$ such that $K \cap P$ is nonempty and $\delta(x) > \delta$ on a dense subset $P' \subset K \cap P$. By shrinking K if necessary, we can also assume that $|K| < \delta$. Now since $K \in B^{\alpha+1}$ there is an interval $(-M, M)$ which contains the range of $f(x)$ on $K \setminus (\bigcup B^\alpha)$. Since P is disjoint from $\bigcup B^\alpha$, we have $K \cap P \subset K \setminus \bigcup B^\alpha$. Then any subinterval of K containing an element of P' must have a relative integral which is in $(-M - \varepsilon, M + \varepsilon)$. Since any component of $K \cap (\bigcup D)$ will have elements of P' arbitrarily close by, its relative integral must be in $[-M - \varepsilon, M + \varepsilon]$. Therefore, K satisfies Proposition (3a) and is therefore in Dom $\mathcal{L}(A)$. Finally since $K \cap P \neq \emptyset$, K can't be in D and we are done.

§5. Recover the primitive (part 2). We will now continue with the program started in §3 where now it is our goal to write a program for the induction $\mathcal{I} = \mathcal{L}(\mathcal{C}(\mathcal{H}(A)))$ mentioned in the last section. This is easily accomplished using the methods already established. We will do it in blocks, making sure everything is either arithmetical or uses a positively arithmetically defined operator, and also perhaps references to previous programs. The proof that each program does what it is supposed to is either trivial or (if it contains a loop) follows easily from Proposition 1. Each block will be given a name in quotation marks, and a definition. Then the relation will be programmed and, when possible, a program for its negation will follow.

"UB$(\bar{c}, \bar{d}, p, q)$"

$$\begin{aligned}
&\iff (\bar{c}, \bar{d}) \text{ is a collection of rational subintervals of } (p,q) \cap (\textstyle\bigcup B^{\alpha-1}), \\
&\qquad \text{where } \alpha = \operatorname{ord}(p,q), \\
&\iff (\exists j) l(\bar{c}) = l(\bar{d}) = j \;\&\; (\forall i \le j) p \le c_i < d_i \le q \\
&\qquad\qquad \&\; (\forall i \le j)(\forall s,t)(c_i < s < t < d_i) \to \\
&\qquad\qquad\qquad \big[(\exists \bar{p}, \bar{q}) \text{ "Cover}(\bar{p}, \bar{q}, s, t)\text{"} \\
&\qquad\qquad\qquad\qquad \&\; \big(\forall i \le l(\bar{p})\big) \operatorname{ord}(p_i, q_i) < \operatorname{ord}(p,q)\big]
\end{aligned}$$

l_{10}: $j \leftarrow \exists$
if $\neg(l(\bar{c}) = l(\bar{d}) = j)$ **or** $\neg(\forall i \le j) p \le c_i < d_i \le q$ **reject**
$i \leftarrow \forall$, $s \leftarrow \forall$, $t \leftarrow \forall$
if $i > j$ **accept**
if $\neg(c_i < s < t < d_i)$ **accept**
$\bar{p} \leftarrow \exists$, $\bar{q} \leftarrow \exists$
$m \leftarrow \forall$
if $m = 1$ **go to** l_2
$i \leftarrow \forall$
if $i > l(\bar{p})$ **accept**
$n \leftarrow 1$, $a \leftarrow p_i$, $b \leftarrow q_i$, $c \leftarrow p$, $d \leftarrow q$
go to l_8

$\neg$UB$(\bar{c}, \bar{d}, p, q)$

l_{11}: $j \leftarrow \forall$
if $\neg(l(\bar{c}) = l(\bar{d}) = j)$ **or** $\neg(\forall i \le j) p \le c_i < d_i \le q$ **accept**

$i \leftarrow \exists$, $s \leftarrow \exists$, $t \leftarrow \exists$
if $i > j$ **reject**
if $\neg(c_i < s < t < d_i)$ **reject**
$\bar{p} \leftarrow \forall$, $\bar{q} \leftarrow \forall$
$m \leftarrow \exists$
if $m = 1$ **go to** l_3
$i \leftarrow \exists$
if $i > l(\bar{p})$ **reject**
$n \leftarrow 2$, $a \leftarrow p$, $b \leftarrow q$, $c \leftarrow p_i$, $d \leftarrow q_i$
go to l_8

"WIT(M, m, p, q)"

$\iff$ (M, m) witnesses that (p, q) belongs to B^α, where $\alpha = \text{ord}(p, q)$,
$\iff$ the mth closed set in the inverse image of $(-M, M)$ contains $(p, q) \setminus \bigcup\{\,(x, y) \mid \text{ord}(x, y) < \text{ord}(p, q)\,\}$,
$\iff$ $(\forall c, d)\big[(c, d) \subset (p, q)\ \&\ \text{INVIM}(-M, M, c, d, m)\big] \to$ "UB(c, d, p, q)".

l_{12}: $c \leftarrow \forall$, $d \leftarrow \forall$
if $\neg(p \le c < d \le q)$ **accept**
if INVIM$(-M, M, c, d, m)$ **go to** l_{13}
accept
l_{13}: $\bar{c} \leftarrow c$, $\bar{d} \leftarrow d$
go to l_{10}

"$\neg$WIT(M, m, p, q)"

l_{14}: $c \leftarrow \exists$, $d \leftarrow \exists$
if $\neg(p \le c < d \le q)$ **reject**
if INVIM$(-M, M, c, d, m)$ **go to** l_{15}
reject
l_{15}: $\bar{c} \leftarrow c$, $\bar{d} \leftarrow d$
go to l_{11}

Our next block characterizes an important open set which we call $G(n, a, b, p, q, \bar{e}, \bar{g})$ which is defined to be the union of the following:

(i) the complement of the union of the first n closed sets in the inverse image under f of the interval (a, b),
(ii) the union of the intervals in $B^{\alpha-1}$ where $\alpha = \text{ord}(p, q)$, and
(iii) the union of the sequence $(\bar{e}, \bar{g})$ of rational intervals.

"$G(s, t, n, a, b, p, q, \bar{e}, \bar{g})$"

$\iff$ "$[s, t]$ is a subset of G",
$\iff$ $(\exists \bar{p}, \bar{q})$ "Cover$(\bar{p}, \bar{q}, s, t)$" & $\big(\forall i \le l(\bar{p})\big)$ either $(\forall k \le n)$INVIM(a, b, p_i, q_i, k), or "UB(p_i, q_i, p, q)" or $(\exists j)(p_i, q_i) \subset (e_j, g_j)$.

l_{16}: $\bar{p} \leftarrow \exists, \bar{q} \leftarrow \exists$
$m \leftarrow \forall$
if $m = 1$ **go to** l_{17}
go to l_2
l_{17}: $i \leftarrow \forall$
if $i > l(\bar{p})$ **accept**
$k \leftarrow \forall$
if $k > n$ **accept**
if $\mathrm{INVIM}(a, b, p_i, q_i, k)$ **accept**
$j \leftarrow \exists$
if $(p_i, q_i) \subset (e_j, g_j)$ **accept**
$\bar{c} \leftarrow p_i, \bar{d} \leftarrow q_i$
go to l_{10}

"$\neg G(s, t, n, a, b, p, q, \bar{e}, \bar{g})$"

l_{18}: $\bar{p} \leftarrow \forall, \bar{q} \leftarrow \forall$
$m \leftarrow \exists$
if $m = 1$ **go to** l_{19}
go to l_3
l_{19}: $i \leftarrow \exists$
if $i > l(\bar{p})$ **reject**
$k \leftarrow \exists$
if $k > n$ **reject**
if $\mathrm{INVIM}(a, b, p_i, q_i, k)$ **reject**
$j \leftarrow \forall$
if $(p_i, q_i) \subset (e_j, g_j)$ **reject**
$\bar{c} \leftarrow p_i, \bar{d} \leftarrow q_i$
go to l_{11}

Let $\mathbb{Q}(m)$ denote some canonical bijection from the positive integers to the rationals, and let the "mth component" of an open set be the one (if any) which includes $\mathbb{Q}(m)$ but does not include $\mathbb{Q}(j)$ for any $j < m$. The "mth component" of an open set may not exist, but at least if we list them this way, no component is counted twice. Our next block estimates the size of the mth component of $G \cap (p, q)$.

"$\mathrm{LEN}(r, n, a, b, p, q, \bar{e}, \bar{g}, m)$"
$\iff$ "$r = 0$ or else the mth component of $G(n, a, b, p, q, \bar{e}, \bar{g}) \cap (p, q)$ exists and has length greater than r",
$\iff$ $r = 0$ or $(\exists s, t) r < t - s \;\&\; p \le s < \mathbb{Q}(m) < t \le q$
$\&$ "$G(s, t, n, a, b, p, q, \bar{e}, \bar{g})$"
$\&$ $(\forall j < m)$ if $\mathbb{Q}(j) < \mathbb{Q}(m)$
then "$\neg G(\mathbb{Q}(j), \mathbb{Q}(m), n, a, b, p, q, \bar{e}, \bar{g})$"

and if $\mathbb{Q}(m) < \mathbb{Q}(j)$
then "$\neg G\big(\mathbb{Q}(m), \mathbb{Q}(j), n, a, b, p, q, \bar{e}, \bar{g}\big)$".

l_{20}: **if** $r = 0$ **accept**
$k \leftarrow \forall$
if $k = 1$ **go to** l_{21}
$s \leftarrow \exists$, $t \leftarrow \exists$
if $r \geq (t - s)$ **or** $\neg\big(p \leq s < \mathbb{Q}(m) < t \leq q\big)$ **reject**
go to l_{16}
l_{21}: $j \leftarrow \forall$
if $j \geq m$ **accept**
if $\mathbb{Q}(j) > \mathbb{Q}(m)$ **go to** l_{22}
$s \leftarrow \mathbb{Q}(j)$, $t \leftarrow \mathbb{Q}(m)$
go to l_{18}
l_{22}: $s \leftarrow \mathbb{Q}(m)$, $t \leftarrow \mathbb{Q}(j)$
go to l_{18}

"$\neg\mathrm{LEN}(r, n, a, b, p, q, \bar{e}, \bar{g}, m)$"
l_{23}: **if** $r = 0$ **reject**
$k \leftarrow \exists$
if $k = 1$ **go to** l_{24}
$s \leftarrow \forall$, $t \leftarrow \forall$
if $r \geq (t - s)$ **or** $\neg\big(p \leq s < \mathbb{Q}(m) < t \leq q\big)$ **accept**
go to l_{18}
l_{24}: $j \leftarrow \exists$
if $j \geq m$ **reject**
if $\mathbb{Q}(j) > \mathbb{Q}(m)$ **go to** l_{25}
$s \leftarrow \mathbb{Q}(j)$, $t \leftarrow \mathbb{Q}(m)$
go to l_{16}
l_{25}: $s \leftarrow \mathbb{Q}(m)$, $t \leftarrow \mathbb{Q}(j)$
go to l_{16}

"$\mathrm{LEBM}(r, n, a, b, p, q, \bar{e}, \bar{g})$"
$\iff$ "$r >$ the Lebesgue measure of $(p, q) \setminus G(n, a, b, p, q, \bar{e}, \bar{g})$",
$\iff$ "the Lebesgue measure of $(p, q) \cap G(n, a, b, p, q, \bar{e}, \bar{g})$ is $> q - p - r$",
$\iff$ "some finite union of components of $(p, q) \cap G(n, a, b, p, q, \bar{e}, \bar{g})$ has measure greater than $q - p - r$",
$\iff$ $(\exists i)(\exists \bar{r})\big[l(\bar{r}) = i \;\&\; \sum \bar{r} > q - p - r \;\&\; (\forall j \leq i)\ \mathrm{LEN}(r_j, n, a, b, p, q, \bar{e}, \bar{g}, j)\big]$.

l_{26}: $i \leftarrow \exists$, $\bar{r} \leftarrow \exists$
if $l(\bar{r}) \neq i$ **or** $\sum \bar{r} \leq q - p - r$ **reject**
$j \leftarrow \forall$
if $j > i$ **accept**

$r \leftarrow r_j, m \leftarrow j$
go to l_{20}

"$\neg$LEBM$(r, n, a, b, p, q, \bar{e}, \bar{g})$"

l_{27}: $i \leftarrow \forall, \bar{r} \leftarrow \forall$
if $l(\bar{r}) \neq i$ **or** $\sum \bar{r} \leq q - p - r$ **accept**
$j \leftarrow \exists$
if $j > i$ **reject**
$r \leftarrow r_j, m \leftarrow j$
go to l_{23}

"LEBI$(r, p, q, \bar{e}, \bar{g})$"

$\iff$ "$r >$ the Lebesgue integral of f on $(p, q) \setminus \bigcup B^{\alpha-1} \setminus \bigcup\{(\bar{e}, \bar{g})\}$, where $\alpha = \operatorname{ord}(p, q)$",

$\iff$ $(\exists M, m)$ "WIT(M, m, p, q)"
& $(\exists \bar{p}, \bar{q}, \bar{u}, k) l(\bar{p}) = l(\bar{q}) = l(\bar{u}) = k$
& $p_1 = -M$, $q_k = M$,
& $(\forall i < k)[p_i < q_i = p_{i+1} < q_{i+1}]$
& $(\forall i \leq k)(\exists n)$ "$\neg$LEBM$(u_i, n, p_i, q_i, p, q, \bar{e}, \bar{g})$"
& $(\exists x)(\forall n)$ "LEBM$(x, n, -M, M, p, q, \bar{e}, \bar{g})$"
& $M \cdot x + \sum \bar{u}(\bar{q} - M) < r$.

The idea here is that we break the range $(-M, M)$ into small intervals (p_i, q_i). The inverse image of each of these intervals on our domain $(p, q) \setminus (\bigcup B^{\alpha-1}) \setminus \bigcup\{(\bar{e}, \bar{g})\}$ is then under-estimated (using $\bar{u}$) by considering only the first j closed sets of this inverse image. We then subtract M from the function, causing it to be negative on its domain. When u_i is multiplied by $(q_i - M)$ this estimate is negative, but not as negative as it should be, and so when these are added we get an over-estimate, $\sum \bar{u}(\bar{q} - M)$, for the integral of $f(x) - M$. Then we add back in $M \cdot x$ where x is an over-estimate for the entire domain, finally obtaining an over-estimate for the integral of f. We then check to see if this can be made less than r.

l_{28}: $M \leftarrow \exists, m \leftarrow \exists$
$z \leftarrow \forall$
if $z = 1$ **go to** l_{12}
$\bar{p} \leftarrow \exists, \bar{q} \leftarrow \exists$
$\bar{u} \leftarrow \exists, k \leftarrow \exists$
if $\neg\big(l(\bar{p}) = l(\bar{q}) = l(\bar{u}) = k \;\&\; p_1 = -M \;\&\; q_k = M\big)$ **reject**
if $\neg(\forall i < k)[p_i < q_i = p_{i+1} < q_{i+1}]$ **reject**
$z \leftarrow \forall$
if $z = 1$ **go to** l_{29}
$i \leftarrow \forall$
if $i > k$ **accept**
if $\neg(p_i < q_i = p_{i+1} < q_{i+1})$ **reject**

$n \leftarrow \exists$
$r \leftarrow u_i,\ a \leftarrow p_i,\ b \leftarrow q_i$
go to l_{27}
l_{29}: $x \leftarrow \exists$
$n \leftarrow \forall$
if $M \cdot x + \sum \bar{u}(\bar{q} - M) \geq r$ **reject**
$r \leftarrow x,\ a \leftarrow -M,\ b \leftarrow M$
go to l_{26}

"$\neg$LEBI$(r, p, q, \bar{e}, \bar{g})$"

l_{30}: $M \leftarrow \forall,\ m \leftarrow \forall$
$z \leftarrow \exists$
if $z = 1$ **go to** l_{14}
$\bar{p} \leftarrow \forall,\ \bar{q} \leftarrow \forall,\ \bar{u} \leftarrow \forall,\ k \leftarrow \forall$
if $\neg(l(\bar{p}) = l(\bar{q}) = l(\bar{u}) = k \,\&\, p_1 = -M \,\&\, q_k = M)$ **accept**
if $\neg(\forall i < k)[p_i < q_i = p_{i+1} < q_{i+1}]$ **accept**
$z \leftarrow \exists$
if $z = 1$ **go to** l_{31}
$i \leftarrow \exists$
if $i > k$ **reject**
if $\neg(p_i < q_i = p_{i+1} < q_{i+1})$ **accept**
$n \leftarrow \forall$
$r \leftarrow u_i,\ a \leftarrow p_i,\ b \leftarrow q_i$
go to l_{26}
l_{31}: $x \leftarrow \forall$
$n \leftarrow \exists$
if $M \cdot x + \sum \bar{u}(\bar{q} - M) \geq r$ **accept**
$r \leftarrow x,\ a \leftarrow -M,\ b \leftarrow M$
go to l_{27}

"$(p, q, r, n) \in \mathcal{H}(A)$"

$$\begin{aligned} \iff (\exists \bar{a}, \bar{r}, j) l(\bar{a}) = l(\bar{r}) = j \,\&\, j > 1 \\ \&\, a_1 = p,\ a_j = q \text{ and } (\forall i < j)(a_i < a_{i+1}) \\ \&\, \big(n = 1 \,\&\, \textstyle\sum \bar{r} < r \text{ or } n = 2 \,\&\, \textstyle\sum \bar{r} > r\big) \\ \&\, (\forall i < j)(a_i, a_{i+1}, r_i, n) \in A. \end{aligned}$$

Here, A denotes the subrelation of INT which is "already" determined ("already" means in a backwards sense; A hasn't actually been mentioned in the program, but represents those "simpler" integrals which now need to be determined). It is in this block then, that the loop for the induction occurs. Notice the final statement "go to l_1". This refers to the start of the entire program which will appear in our last block.

l_{32}: $\bar{a} \leftarrow \exists,\ \bar{r} \leftarrow \exists,\ j \leftarrow \exists$
if $\neg(l(\bar{a}) = l(\bar{r}) = j \,\&\, j > 1 \,\&\, a_1 = p \,\&\, a_j = q)$ **reject**

if $(\exists i < j) a_i \geq a_{i+1}$ **reject**
if $\neg(n = 1 \,\&\, \sum \bar{r} < r$ or $n = 2 \,\&\, \sum \bar{r} > r)$ **reject**
$i \leftarrow \forall$
if $i \geq j$ **accept**
$p \leftarrow a_i,\, q \leftarrow a_{i+1},\, r \leftarrow r_i$
go to l_1

"$(p, q, r, n) \in \mathcal{C}(\mathcal{H}(A))$"

$$\begin{aligned} \iff (\exists s, t)(\forall \varepsilon)(\exists u, a, b)\ & p < a < b < q,\ a - p + q - b < \varepsilon,\ s < r < t, \\ & (\text{either } n = 1 \,\&\, u < s \text{ or } n = 2 \,\&\, u > t), \\ & \&\, \text{“}(a, b, u, n) \in \mathcal{H}(A)\text{”}. \end{aligned}$$

l_{33}: $s \leftarrow \exists,\, t \leftarrow \exists$
$\varepsilon \leftarrow \forall$
$u \leftarrow \exists,\, a \leftarrow \exists,\, b \leftarrow \exists$
if $\neg(p < a < b < q \,\&\, a - p + q - b < \varepsilon \,\&\, s < r < t)$ **reject**
if $\neg(n = 1 \,\&\, u < s$ or $n = 2 \,\&\, u > t)$ **reject**
$p \leftarrow a,\, q \leftarrow b,\, r \leftarrow u$
go to l_{32}

The next relation "COVER" is capitalized to distinguish it from our previously defined "Cover":

"COVER$(\bar{a}, \bar{b}, \bar{c}, \bar{d})$"

$$\begin{aligned} \iff\ & (\bar{c}, \bar{d}) \text{ covers } (\bar{a}, \bar{b}), \\ \iff\ & (\exists j, k) l(\bar{a}) = l(\bar{b}) = j,\ l(\bar{c}) = l(\bar{d}) = k, \\ & \quad \&\, (\forall i \leq j)(\exists l \leq k)(c_l \leq a_i < b_i \leq d_l). \end{aligned}$$

l_{34}: $j \leftarrow \exists,\, k \leftarrow \exists$
if $\neg(l(\bar{a}) = l(\bar{b}) = j \,\&\, l(\bar{c}) = l(\bar{d}) = k)$ **reject**
if $(\forall i \leq j)(\exists l \leq k)(c_l \leq a_i < b_i \leq d_l)$ **accept**
reject

"$\neg$COVER$(\bar{a}, \bar{b}, \bar{c}, \bar{d})$"

l_{35}: $j \leftarrow \forall,\, k \leftarrow \forall$
if $\neg(l(\bar{a}) = l(\bar{b}) = j \,\&\, l(\bar{c}) = l(\bar{d}) = k)$ **accept**
if $\neg(\forall i \leq j)(\exists l \leq k)(c_l \leq a_i < b_i \leq d_l)$ **reject**
accept

"DISJ$(\bar{e}, \bar{g}, p, q)$"

$$\begin{aligned} \iff\ & (\bar{e}, \bar{g}) \text{ are disjoint subintervals of } (p, q), \\ \iff\ & (\exists j)(l(\bar{e}) = l(\bar{g}) = j) \\ & \quad \&\, (\forall k, m \leq j)\big(e_m \notin (e_k, g_k) \,\&\, g_m \notin (e_k, g_k) \\ & \quad \&\, p \leq e_m < g_m \leq q\big) \end{aligned}$$

l_{36}: $j \leftarrow \exists$
if $\neg(l(\bar{e}) = l(\bar{g}) = j)$ **reject**

if $(\forall k, m \le j)\big(e_m \notin (e_k, g_k)\ \&\ g_m \notin (e_k, g_k)$
$\quad\&\ p \le e_m < g_m \le q\big)$ **accept**
reject

"$\neg$DISJ$(\bar{e}, \bar{g}, p, q)$"

l_{37}: $j \leftarrow \forall$
if $\neg(l(\bar{e}) = l(\bar{g}) = j)$ **accept**
if $\neg(\forall k, m \le j)\big(e_m \notin (e_k, g_k)\ \&\ g_m \notin (e_k, g_k)$
$\quad\&\ p \le e_m < g_m \le q)$ **reject**
accept

"SUBCOLL$(\bar{u}, \bar{v}, \bar{e}, \bar{g})$"
$\iff$ the collection of intervals $(\bar{u}, \bar{v})$ is a subset of the collection $(\bar{e}, \bar{g})$, with left endpoints in increasing order and with no duplication,
$\iff (\exists j, k)\big(l(\bar{u}) = l(\bar{v}) = j \le l(\bar{e}) = l(\bar{g}) = k\big)$
$\quad\&\ (\forall i \le j)(\exists m \le k)(u_i = e_m, v_i = g_m)$
$\quad\&\ (\forall i < j) u_i < u_{i+1}$.

l_{38}: $j \leftarrow \exists, k \leftarrow \exists$
if $\neg(l(\bar{u}) = l(\bar{v}) = j \le l(\bar{e}) = l(\bar{g}) = k)$ **reject**
if $(\forall i \le j)(\exists m \le k)(u_i = e_m, v_i = g_m)\ \&\ (\forall i < j) u_i < u_{i+1}$ **accept**
reject

"$\neg$SUBCOLL$(\bar{u}, \bar{v}, \bar{e}, \bar{g})$"

l_{39}: $j \leftarrow \forall, k \leftarrow \forall$
if $\neg(l(\bar{u}) = l(\bar{v}) = j \le l(\bar{e}) = l(\bar{g}) = k)$ **accept**
if $\neg(\forall i \le j)(\exists m \le k)(u_i = e_m, v_i = g_m)$ **or** $\neg(\forall i < j) u_i < u_{i+1}$
reject
accept

"PROP4(p, q, r, n)"
$\iff (\exists M, s, t)(s < r < t)\ \&\ (\exists \bar{a}, \bar{b})$ "UB$(\bar{a}, \bar{b}, p, q)$"
$\quad\&\ (\forall \bar{c}, \bar{d})\big($"UB$(\bar{c}, \bar{d}, p, q)$" & "COVER$(\bar{a}, \bar{b}, \bar{c}, \bar{d})$" $\big) \rightarrow$
$\quad(\exists \bar{e}, \bar{g})\big[$"DISJ$(\bar{e}, \bar{g}, p, q)$ & "COVER$(\bar{c}, \bar{d}, \bar{e}, \bar{g})$"
$\qquad\&\ \big(\forall k \le l(\bar{e})\big)$
$\qquad\quad\big[(\exists x, y)\big(-M(g_k - e_k) < y < x < M(g_k - e_k)\big)$
$\qquad\qquad\&$ "$(r_k, s_k, x, 1) \in C(H(A))$"
$\qquad\qquad\&\ (r_k, s_k, y, 2) \in C(H(A))$"$\big]$
$\qquad\&\ (\forall \bar{u}, \bar{v})\big($"COVER$(\bar{a}, \bar{b}, \bar{u}, \bar{v})$"
$\qquad\qquad\&$ "SUBCOLL$(\bar{u}, \bar{v}, \bar{e}, \bar{g})$"$\big) \rightarrow$
$\qquad\qquad\big[(\exists \bar{x}, m)(l(\bar{x}) = l(\bar{u}) = m)$
$\qquad\qquad\quad\&\ (\forall i \le m)$ "$(u_i, v_i, x_i, n) \in C(H(A))$"
$\qquad\qquad\quad\&\ (\exists y)\big(n = 1\ \&$ "LEBI$(y, p, q, \bar{u}, \bar{v})$"

$$\begin{aligned}&\& \, y + \textstyle\sum \bar{x} < s)\\ \text{or } \big(n = 2 \,\&\, &\text{“}\neg\text{LEBI}(y, p, q, \bar{u}, \bar{v})\text{”}\\ &\& \, y + \textstyle\sum \bar{x} > t)\big]\big].\end{aligned}$$

l_{40}: $M \leftarrow \exists$, $s \leftarrow \exists$, $t \leftarrow \exists$
if $\neg(s < r < t)$ **reject**
$\bar{a} \leftarrow \exists$, $\bar{b} \leftarrow \exists$
$z \leftarrow \forall$
if $z = 1$ **go to** l_{41}
$\bar{c} \leftarrow \bar{a}$, $\bar{d} \leftarrow \bar{b}$
go to l_{10}
l_{41}: $\bar{c} \leftarrow \forall$, $\bar{d} \leftarrow \forall$
$z \leftarrow \exists$
if $z = 1$ **go to** l_{35}, **if** $z = 2$ **go to** l_{42}
go to l_{11}
l_{42}: $\bar{e} \leftarrow \exists$, $\bar{g} \leftarrow \exists$
$z \leftarrow \forall$
if $z = 1$ **go to** l_{43}, **if** $z = 2$ **go to** l_{44}, **if** $z = 3$ **go to** l_{46}
go to l_{36}
l_{43}: $\bar{a} \leftarrow \bar{c}$, $\bar{b} \leftarrow \bar{d}$, $\bar{c} \leftarrow \bar{e}$, $\bar{d} \leftarrow \bar{g}$
go to l_{34}
l_{44}: $k \leftarrow \forall$
if $k > l(\bar{e})$ **accept**
$x \leftarrow \exists$, $y \leftarrow \exists$
if $\neg(-M(g_k - e_k) < y < x < M(g_k - e_k))$ **reject**
$z \leftarrow \forall$
if $z = 1$ **go to** l_{45}
$p \leftarrow r_k$, $q \leftarrow s_k$, $r \leftarrow x$, $n \leftarrow 1$
go to l_{33}
l_{45}: $p \leftarrow r_k$, $q \leftarrow s_k$, $r \leftarrow y$, $n \leftarrow 2$
go to l_{33}
l_{46}: $\bar{u} \leftarrow \forall$, $\bar{v} \leftarrow \forall$
$z \leftarrow \exists$
if $z = 1$ **go to** l_{39}, **if** $z = 2$ **go to** l_{47}
$\bar{c} \leftarrow \bar{u}$, $\bar{d} \leftarrow \bar{v}$
go to l_{35}
l_{47}: $\bar{x} \leftarrow \exists$, $m \leftarrow \exists$
if $\neg(l(\bar{x}) = l(\bar{u}) = m)$ **reject**
$z \leftarrow \forall$
if $z = 1$ **go to** l_{48}
$i \leftarrow \forall$
if $i > m$ **accept**
$p \leftarrow u_i$, $q \leftarrow v_i$, $r \leftarrow x_i$

go to l_{33}
l_{48}: $y \leftarrow \exists$
if $\neg(n = 1$ **or** $n = 2)$ **reject**
if $n = 2$ **go to** l_{49}
if $\neg(y + \sum \bar{x} < s)$ **reject**
$r \leftarrow y, \bar{e} \leftarrow \bar{u}, \bar{g} \leftarrow \bar{v}$
go to l_{28}
l_{49}: **if** $\neg(y + \sum \bar{x} > t)$ **reject**
$r \leftarrow y, \bar{e} \leftarrow \bar{u}, \bar{g} \leftarrow \bar{v}$
go to l_{30}

$$\text{“}(p,q,r,n) \in \mathcal{L}(\mathcal{C}(\mathcal{H}(A)))\text{”} \iff \text{“}(p,q,r,n) \in \mathcal{C}(\mathcal{H}(A))\text{” or} \\ \big[\big[(\forall c,d)\text{“UB}(c,d,p,q)\text{”} \rightarrow (\exists m,u)\text{“}(c,d,u,m) \in \mathcal{C}(\mathcal{H}(A))\text{”}\big] \\ \&\,\text{“PROP4}(p,q,r,n)\text{”}\big].$$

l_1: $z \leftarrow \exists$
if $z = 1$ **go to** l_{33}
$t \leftarrow \forall$
if $t = 1$ **go to** l_{40}
$c \leftarrow \forall, d \leftarrow \forall$
$z \leftarrow \exists$
if $z = 1$ **go to** l_{50}
$\bar{c} \leftarrow c, \bar{d} \leftarrow d$
go to l_{11}
l_{50}: $m \leftarrow \exists, u \leftarrow \exists$
$p \leftarrow c, q \leftarrow d, r \leftarrow u, n \leftarrow m$
go to l_{33}

Appendix §A. The solutions of Denjoy, Perron-Bauer, and Kurzweil-Henstock. We outline here the three most famous solutions to the problem of the primitive, mentioned in the introduction.

A.1. Denjoy's solution. Let D be a collection of open real intervals I, on which the definite integral $F(I)$ is known. Suppose also that whenever $J \subset I \in D$ then $J \in D$. There are three techniques which may be used to find a definite integral $F(J)$ where $J \notin D$:

(a) (trivial) If J is partitioned by the intervals $I_1, \ldots, I_n \in D$ then $F(J) = \sum F(I_n)$.
(b) (Cauchy) If $J = \lim_{n\to\infty} I_n$ where each $I_n \in D$ then $F(J) = \lim_{n\to\infty} F(I_n)$. This is a direct consequence of the continuity of F.
(c) (Lebesgue) If f is bounded on $J \setminus \bigcup D$ then the Lebesgue integral $L(J \setminus \bigcup D)$ exists. If we also have a bound for the difference

quotients $F(I)/|I|$ over all components I of $J \cap (\bigcup D)$ then the corresponding sum $\sum F(I)$ over these components is absolutely convergent, and will be denoted by $F(J \cap (\bigcup D))$. It then follows by a theorem of Lebesgue that $F(J) = F(J \cap (\bigcup D)) + L(J \setminus \bigcup D)$. Application of this technique, of course, will require that every component of $J \cap (\bigcup D)$ be already in D.

First note that if $F(J)$ can be found by any of the above techniques, then so can $F(K)$ for any $K \subset J$. Therefore, each provides a method of expanding D.

Next we show that at least one of the three techniques will properly expand D unless, of course, D already contains all open intervals. To this end, suppose neither (a) nor (b) expands D. Then $\mathbb{R} \setminus \bigcup D$ could have no isolated points. Furthermore, any interval covered by finitely many elements of $\bigcup D$ would already be in D. If in addition, (b) fails then any component of $\bigcup D$ is already in D. Under this assumption we show that (c) will not fail.

Using $\varepsilon = 1$ in the definition of derivative, we know that for each x there is a $\delta(x) > 0$ such that the difference quotient $F(I)/|I|$ is within 1 of $f(x)$ whenever I contains x and $|I| < \delta(x)$. An easy argument using the continuity of F shows that the function $\delta(x)$ can be chosen to be uppersemicontinuous, meaning that each of the sets $\{ x \mid \delta(x) \geq \delta \}$ is closed. It follows by the Baire Category Theorem that for some $\delta > 0$, the set $\{ x \mid \delta(x) \geq \delta \}$ contains a nontrivial portion, J, of $\mathbb{R} \setminus \bigcup D$. Assume without loss of generality that $|J| < \delta$. Then for each x in $J \setminus \bigcup D$, $f(x)$ is within 1 of $F(J)/|J|$ implying that f is bounded on $J \setminus \bigcup D$. Furthermore, if I is a component of $J \cap (\bigcup D)$ and x is an endpoint of I in J, then the difference quotient $F(I)/|I|$ is within 1 of $f(x)$, hence within 2 of $F(J)/|J|$. Therefore such difference quotients are also bounded. It follows that $F(J)$ can be calculated by (c).

Finally, since one of the techniques always succeeds, we may iterate them, taking unions at limit ordinals, until D contains all open intervals. At this point, the definite integral for any interval is known and hence so is the primitive (up to a constant).

Note: Several variations of this technique are possible. For example, Dougherty and Kechris first obtain a transfinite sequence $D_0, D_1, \ldots$ of open sets, with f being bounded on each $D_{\alpha+1} \setminus D_\alpha$. Then they apply Denjoy's procedure to find the integral on each piece. Our computer program is modeled after this Dougherty-Kechris variation.

A.2. Perron-Bauer solution. Let f be a derivative and $\varphi(x)$, $\psi(x)$ be continuous functions with $\varphi(0) = \psi(0) = 0$. Then φ is called a "majorant" (or "major function") of f if

$$\underline{D}\varphi(x) = \liminf_{h \to 0} \frac{\varphi(x+h) - \varphi(x)}{h}$$

is always at least as big as $f(x)$. Similarly, $\psi(x)$ is called a "minorant" of f if

$$\bar{D}\psi(x) = \limsup_{h\to 0} \frac{\psi(x+h)-\psi(x)}{h}$$

is always at least as small as $f(x)$. We show that when $x \geq 0$, $\varphi(x) \geq \psi(x)$. To see this note that $\underline{D}(\varphi - \psi) \geq \underline{D}(\varphi) - \bar{D}(\psi) \geq 0$. Hence, $\varphi - \psi$ is monotonically increasing. Then since $\varphi(0) - \psi(0) = 0$, $\varphi(x) - \psi(x) \geq 0$ when $x \geq 0$.

Let F be the unique primitive of f with $F(0) = 0$. Since F is both a majorant and a minorant, we have for any majorant φ, and any minorant ψ, and any $x \geq 0$, that $\varphi(x) \geq F(x) \geq \psi(x)$. It follows that for $x \geq 0$, $F(x)$ can be found by taking the infimum of the majorants or the supremum of the minorants.

Note: Denjoy ([7], pp. 677–683) likened this to the following situation: Suppose two people wanted to know how to get an airplane to fly twice the speed of sound. The first person carefully details the design of the wings, the construction of the engine, etc. The second person finds a much easier approach. Just take the infimum of all the airplanes which go more than twice the speed of sound and the supremum of all the ones which go less than twice the speed of sound, and there it is!

A.3. Kurzweil-Henstock solution. Let us first recall from elementary calculus, the definitions behind the Riemann integral:

Definition. A finite sequence of points $x_0 < x_1 < \cdots < x_n$ is called a partition of $[a, b]$ if $x_0 = a$ and $x_n = b$. The mesh of the partition is the maximum value of $x_i - x_{i-1}$. A set of tags for a partition is a collection of elements c_i with $c_i \in [x_{i-1}, x_i]$. Given a function f and a tagged partition, the corresponding Riemann sum is $\sum_{i=1}^{n} f(c_i)(x_i - x_{i-1})$. Finally, I is called the Riemann integral of f on $[a, b]$ if and only if $(\forall\varepsilon > 0)(\exists\delta > 0)(\forall$ tagged partition of $[a, b]$ with mesh $< \delta)$ the corresponding Riemann sum is within $\varepsilon(b - a)$ of I.

The Riemann integral cannot be used to invert every derivative, but it will invert a continuous one. To see this, let $F'(x) = f(x)$ where f is continuous. By the definition of derivative, given x and $\varepsilon > 0$ there is a $\delta > 0$ such that $|F(y) - F(z) - f(x)(y - z)| < \varepsilon(y - z)$ whenever $z \leq x \leq y$ and $y - z < \delta$. Fix ε and for each x, let $\delta(x)$ denote the largest value of δ for which this definition holds. Also, let $\delta'(x)$ be similarly defined using $\varepsilon/2$. Then by the continuity of F and f we have that for all x' in some neighborhood of x, $\delta(x') \geq \delta'(x)$. Such neighborhoods form an open covering of $[a, b]$ and so by compactness, there is some δ such that $\delta(x) \geq \delta > 0$ for each x in $[a, b]$. Then, if $x_0 < x_1 < \cdots < x_n$ is a partition of $[a, b]$ with mesh less than δ, and if $\{c_i\}$ is a collection of tags, with $c_i \in (x_{i-1}, x_i)$ then $F(x_i) - F(x_{i-1})$

is within $\varepsilon(x_i - x_{i-1})$ of $f(c_i)(x_i - x_{i-1})$. Summing over i, we get that $\sum f(c_i)(x_i - x_{i-1})$ is within $\varepsilon(b-a)$ of $F(b) - F(a)$ and we are done.

If $f(x)$ is not continuous, then we can't always find a uniform δ. Kurzweil (and later Henstock) made the observation that in this case we can just leave $\delta(x)$ a function of x (called a "gauge" function) and then replace the condition that the mesh be less than δ to the requirement that $x_i - x_{i-1} < \delta(c_i)$ (this requirement on a tagged partition is called "δ-fine"). The same proof now shows that this new definition of integral will invert any derivative. In fact, the proof is much simpler because we no longer have to show that $\delta(x)$ is bounded above zero. This improved version is sometimes called the Riemann-complete integral.

Note: It may be hard to believe that Riemann himself was not aware of this "improvement". But if he was, why didn't he state his integral in this more powerful way? Perhaps one explanation is that despite the strong similarity, there is a profound difference between the original Riemann integral and the Riemann-complete version. Riemann's integral can be effectively used to obtain approximations to the increment $F(b) - F(a)$. One just chooses partitions with smaller and smaller mesh, and is guaranteed that the corresponding Riemann sums will get closer and closer to the correct value. In the Riemann-complete integral we don't know (without prior knowledge of the gauge function) if our tagged partitions are ever going to be δ-fine, and without this knowledge we are totally at a loss. There is no way to tell if our approximations will be getting better and better or if they are getting worse and worse.

REFERENCES

[1] A. Blass and D. Cenzer, *Cores of Π^1_1 sets of reals*, ***Journal of Symbolic Logic***, vol. 39 (1974), pp. 649–654.

[2] A. Bruckner, ***Differentiation of real functions***, CRM Monograph Series, no. 5, American Mathematical Society, Providence, 1994.

[3] P. S. Bullen, *Non-absolute integrals: a survey*, ***Real Analysis Exchange***, vol. 5 (1979–80), pp. 195–259.

[4] D. Cenzer and R. D. Mauldin, *Inductive definability: measure and category*, ***Advances in Mathematics***, vol. 38 (1980), no. 1, pp. 55–90.

[5] U. Darji, M. Evans, and R. O'Malley, *First return path systems: differentiability, continuity, and orderings*, to appear.

[6] A. Denjoy, *Calcul de la primitive de la fonction derivée la plus genérale*, ***Comptes rendus hebdomadaires des séanses de l' Académie des sciences Paris, Series I, Mathematiques***, vol. 154 (1912), pp. 1075–1078.

[7] ———, *Leçons sur le calcul des coefficients d'une série trigonométrique, i-iv*, Paris, 1941–1949.

[8] R. Dougherty and A.S. Kechris, *The complexity of antidifferentiation*, ***Advances in Mathematics***, vol. 88 (1991), pp. 145–169.

[9] D. Harel and D. Kozen, *A programming language for the inductive sets, and applications*, ***Information and Control***, vol. 63 (1984), pp. 118–139.

[10] S. Kleene, *Arithmetical predicates and function quantifiers*, ***Transactions of the American Mathematical Society***, vol. 79 (1955), pp. 405–428.

[11] H. Looman, *Über die Perronsche Integraldefinition*, ***Mathematische Annalen***, vol. 93 (1925), pp. 153–156.

[12] Y. Matiyasevich, *A new proof of the theorem on exponential diophantine representation of enumerable sets*, ***Journal of Soviet Mathematics***, vol. 14 (1980), pp. 1475–1486.

[13] Y. N. Moschovakis, ***Elementary induction on abstract structures***, North-Holland, Amsterdam, 1975.

[14] ———, ***Descriptive set theory***, North-Holland, Amsterdam, 1980.

[15] I. P. Natanson, ***Theory of functions of a real variable***, Ungar, New York, 1955 and 1959.

[16] I. Pesin, ***Classical and modern integration theories***, Academic Press, New York and London, 1970.

[17] H. Rogers, ***Theory of recursive functions and effective computability***, McGraw-Hill Series in Higher Mathematics, McGraw-Hill, New York, 1967.

[18] S. Saks, ***Theory of the integral***, Hafner Publishing Company, Warsaw, 1937.

[19] Z. Zahorski, *Sur la primière dérivée*, ***Transactions of the American Mathematical Society***, vol. 69 (1950), pp. 1–54.

MATHEMATICS DEPARTMENT
CALIFORNIA STATE UNIVERSITY
SAN BERNARDINO, CALIFORNIA 92407

E-mail: cfreilin@wiley.csusb.edu