REMARKS ON THE THEORY OF GEOMETRICAL CONSTRUCTIONS

ERWIN ENGELER*

It is astonishing that a field historically as close to the foundations of mathematics as the theory of geometrical constructions has received so little attention by logicians and has been left so largely untouched by the methods of formalization and axiomatization. From the point of view of contemporary mathematics, this has left the field with some gaping ambiguities and inadequacies in the formulation of the most basic notions and results.

Quite apart from these feelings of regret in the state of one of the oldest and historically most important branches of mathematics, there are several reasons for devoting renewed interest to this field. First, in connection with the topic of this conference, the theory allows and motivates interesting applications of infinitary logic without which a formalization could probably not have been obtained. The second reason is psychological: Our intuition in the constructive theory of real numbers is not as weak as we might be led to believe by recursive analysis, for example. We feel, on the contrary, that an adequate (and more nearly standard) constructive analysis could be developed on the basis of computationally closed fields (section 2) rather than on the set of computable real numbers. Finally, there is a didactic reason: In this era of programmed computers, a treatment of geometrical construction programs might serve as a very welcome didactic tool at an early stage of the training of our students.

For people unsympathetic with infinitely long formulas, let it be remarked that the kind of formulas used in our formalization can easily be encoded in expressions of finite length. The detour through general infinitely long formulas is merely a useful technical device.

*Work supported in part by NSF Grant GP-5434.

1. Construction programs in plane Euclidean geometry.

The need for a programming language in which to formulate directions for geometrical constructions has not arisen in the past because these directions were addressed to budding mathematicians rather than to stupid machines. Also, for the rather narrow part of theory of geometrical constructions that has received algebraic treatment as an application of Galois theory, the description of constructions could indeed be left informal. For a more comprehensive approach, however, we do need a clarification of several of the fundamental notions. The groundwork for doing this was laid in [1] to which we refer for the proof of Theorem 1.4 and for the clarification of some of the features below that are not self-explanatory.

Consider the Euclidean plane as a collection of three kinds of entities: points (for which we will use variables $P_0, P_1, P_2, \ldots$), lines (variables $\ell_0, \ell_1, \ell_2, \ldots$), and circles (variables $\gamma_0, \gamma_1, \gamma_2, \ldots$). Between these entities there are defined certain relations, among them equality ($P_i = P_j, \ell_i = \ell_j, \gamma_i = \gamma_j$), incidence ($I(P_i,\ell_j), I(P_i,\gamma_j), I(\ell_i,\gamma_j), I(\gamma_j,\gamma_k)$), betweeness ($B(P_i,P_j,P_k)$), parallelity ($\ell_i \| \ell_j$), and equidistance ($E(P_i,P_j;P_r,P_s)$). In addition, we assume that two lines, x and y, have been singled out such that x and y are perpendicular (a notion that is definable), intersect in a point 0, and that we have selected points E_x on x, E_y on y such that $0,E_x$ and $0,E_y$ are equidistant and $E_x \neq 0 \neq E_y$.

The basic capabilities of the Euclidean constructor can be expressed as follows: $(i,j,k,s = 0,1,2,\ldots)$:

A. Operating capabilities.

(1) $\ell_i := (P_j,P_k)$, draw a line ℓ_i through two distinct points P_j,P_k;

(2) $P_i := (\ell_j,\ell_k)$, find the intersection P_i of two non-parallel lines;

(3) $\gamma_i := [P_j,P_k]$, draw the circle γ_i of center P_j and passing through $P_k, P_j \neq P_k$;

(4) $P_i := (\gamma_j,\ell_k)$, choose an intersection point P_i of a circle γ_j and an intersecting line ℓ_k;

(5) $P_i := (P_s,\gamma_j,\ell_k)$, given a line through a peripheral point of a circle, find the other intersection point;

(6) $P_i := (\gamma_j,\gamma_k)$, choose an intersection point of two intersecting circles;

(7) $P_i := (P_s,\gamma_j,\gamma_k)$, given an intersection point of two circles, find the other intersection point.

(8) $P_i := 0$, $P_i := E_x$, $P_i := E_y$, $\ell_i := x$, $\ell_i := y$, $P_i := P_j$, $\ell_i := \ell_j$, $\gamma_i := \gamma_j$.

B. Decision-making capabilities.

(1) $P_i = P_j$, $\ell_i = \ell_j$, $\gamma_i = \gamma_j$, decide whether two points, lines, or circles are equal;

(2) $\ell_i \| \ell_j$, decide whether two lines are parallel;

(3) $I(P_i, \ell_j)$, decide whether P_i lies on ℓ_j;

(4) $I(P_i, \gamma_j)$, decide whether P_i lies on γ_j;

(5) $I(\ell_i, \gamma_j)$, decide whether ℓ_i intersects γ_j;

(6) $I(\gamma_i, \gamma_j)$, decide whether γ_i and γ_j intersect;

(7) $B(P_i, P_j, P_k)$, decide whether P_j lies between P_i and P_k;

(8) $E(P_i, P_j; P_k, P_s)$, decide whether P_i, P_j and P_k, P_s are equidistant.

On the basis of these operations and decisions, we can write programs as in [1], composing them of individual labelled instructions of the following kind.

k: do ψ then go to p, (for operations ψ);

k: if φ then go to p else go to q, (for decision-conditions φ).

There is a slight complication here due to the fact that some of the operations are not toal, i.e., not defined for some of the arguments. However, for each such operation there is a condition among B(1) - B(8) which decides whether the operation is geometrically performable. Thus we agree to replace each instruction

s: do $\ell_i := (P_j, P_k)$ then go to p

by a subroutine

s: if $P_j = P_k$ then go to s else go to s';

s': do $\ell_i := (P_j, P_k)$ then go to p

where s' is a label that did not occur before in the program. Similarly for the other partial operations. The end effect is that we may consider all operations as total; the program does not terminate if at some point a geometrically non-performable operation is called upon. We describe this situation by saying that the partial operations in A(1) - A(8) are definite partial operations.

Clearly, the list of operations and decisions above is highly redundant, and it is an easy exercise in plane Euclidean geometry to reduce this list. This means to provide subroutines to replace some operational instructions or to provide subroutines to replace some conditional instructions. For example, a conditional instruction

k: <u>if</u> $I(P_i,\ell_j)$ <u>then go to</u> p <u>else go to</u> q

can be replaced by

k: <u>if</u> $\ell_j \parallel x$ <u>then go to</u> k_1 <u>else go to</u> k_5;

k_1: <u>do</u> $P_0 := (y,\ell_j)$ <u>then go to</u> k_2;

k_2: <u>if</u> $P_0 = P_i$ <u>then go to</u> p <u>else go to</u> k_3;

k_3: <u>do</u> $\ell_0 := (P_0,P_i)$ <u>then go to</u> k_4;

k_4: <u>if</u> $\ell_0 = \ell_j$ <u>then go to</u> p <u>else go to</u> q;

k_5: <u>do</u> $P_0 := (x,\ell_j)$ <u>then go to</u> k_2.

(We assume that the variables P_0,ℓ_0 do not occur in the original program and that the labels $k_1,\ldots,k_5$ don't occur either, otherwise we simply rename the appropriate variables and labels.)

THEOREM 1.1. <u>The set of all points on the line</u> x <u>forms an euclidean ordered field</u> $\mathfrak{a}$ <u>in which the operations</u> $+$, $\cdot$, $-$, $^{-1}$, $\sqrt{\ }$ <u>and the relation</u> $\le$ <u>are definable by programs in terms of the capabilities</u> A(1) - B(8).

This theorem is due, in essence, to Hilbert. For an operation, say $+$, to be definable by a program we mean that there is a program π^+ which contains the variables P_i, P_j and P_k and has the property that π^+ terminates for each assignment of points of x to the variables P_i, P_j. The value of P_k at termination is the result of the operation $+$. A relation, say $\le$, is definable by a program, if there is a program $\pi^{\le}$ with two exits and containing the variables P_i, P_j such that $\pi^{\le}$ terminates for any assignment of points of x to P_i, P_j either in one exit (if the relation holds), or in the other (if it doesn't). The details for writing these programs are well-known.

Conversely, suppose that an euclidean ordered field is given, say $\mathfrak{a} = \langle A, \le, +, \cdot, -, ^{-1}, 0, 1\rangle$, with respect to which we have the following capabilities (using $x_0,x_1,\ldots$ for variables over A):

(a) $x_i := x_j$, $x_i := 0$, $x_i := 1$, $x_i := x_j + x_k$, $x_i := x_j \cdot x_k$, $x_i := -x_j$, $x_i := (x_j)^{-1}$ (for $x_j \neq 0$); $x_i := \sqrt{x_j}$ (for $x_j \geq 0$); $i,j,k = 0,1,2,\ldots$.

(b) $x_i \leq x_j$, $x_i = x_j$, $x_i = 0$, $x_i = 1$; $i,j = 0,1,2,\ldots$.

Consider the plane analytic geometry $G(\mathcal{a})$ over $\mathcal{a}$ in which points, lines, and circles, incidence, etc. are defined in the usual way.

THEOREM 1.2. $G(\mathcal{a})$ _is a geometry in which the operation_ A(1) - A(8) _and the relations_ B(1) - B(8) _are definable by programs in terms of the capabilities_ (a) _and_ (b).

This theorem may be ascribed to Descartes and is well-known.

COROLLARY 1.3. _If_ $\mathcal{a}_1$ _and_ $\mathcal{a}_2$ _are euclidean ordered fields such that each program over_ (a), (b) _terminates in_ $\mathcal{a}_1$ _if and only if it terminates in_ $\mathcal{a}_2$, _then every program over_ A(1) - B(8) _terminates in_ $G(\mathcal{a}_1)$ _if and only if it terminates in_ $G(\mathcal{a}_2)$. _Conversely if_ $\mathcal{G}_1$, $\mathcal{G}_2$ _are euclidean planes such that each program over_ A(1) - B(8) _terminates in_ $\mathcal{G}_1$ _if and only if it terminates in_ $\mathcal{G}_2$, _then every program over_ (a), (b) _terminates in the field corresponding to_ $\mathcal{G}_1$ _if and only if it terminates in the field corresponding to_ $\mathcal{G}_2$. _(We say that a program terminates if it terminates for all assignments of initial values to the variables._)

The proof is obvious from 1.1 and 1.2.

Loosely speaking, a _construction problem_ is the problem of finding a program over A(1) - B(8) which constructs certain points, lines, and circles from given data in such a way that the configuration that is obtained has certain geometrical properties. The crucial question is: what kind of property? The most natural answer, in our opinion, is that the property be _verifiable_ by the constructive means at our disposal.[1] This means that we have a program whose favorable outcome is a necessary and sufficient condition for the property to hold. It is clear that by a slight change in the program (leading non-favorable exits into non-terminating subroutines), we can normalize the notion of favorable termination to simply termination. Properties that are in this sense verifiable by programs are called _algorithmic properties_, and we have the following result:

THEOREM 1.4. _To every program_ π _with free variables_ $x_1,\ldots,x_n$ _we can effectively find a_

---

[1] See Remark 2 in Section 3 for a discussion of other views of this notion.

quantifier-free formula $\varphi(x_1,\dots,x_n)$ in $L_{\omega_1,\omega}$ such that the termination of π for an assignment is equivalent with φ holding for this assignment; [1].

We call two euclidean geometries <u>algorithmically equivalent</u> if, for every program π over A(1) - B(8), π terminates in one geometry for all initial assignments if and only if it does in the other. Similarly for euclidean ordered fields.

From Corollary 1.3 follows at once:

COROLLARY 1.5. <u>Two geometries are algorithmically equivalent if and only if their corresponding fields are algorithmically equivalent.</u>

Thus, if our goal is to axiomatize constructive geometry, we need a characterization of the class of all euclidean fields that are algorithmically equivalent to the field of reals. Once an axiomatic characterization of this concept is found the remainder is straightforward.

2. <u>An axiomatization of the algorithmic theory of real numbers</u>.

The algorithmic basis $\mathfrak{B}$ for the field $\mathfrak{R}$ of real numbers, as realized by the theory of geometrical constructions, consists of the following:

(a) $x_i := x_j$, $x_i := 0$, $x_i := 1$, $x_i := x_j + x_k$, $x_i := x_j \cdot x_k$, $x_i := -x_j$, $x_i = (x_j)^{-1}$ (for $x_j \neq 0$); $i,j,k = 0,1,2,\dots$.

(b) $x_i \leq x_j$, $x_i = x_j$, $x_i = 0$, $x_i = 1$; $i,j = 0,1,2,\dots$.

Note that $x_i := (x_j)^{-1}$ is a partial operation, undefined for $x_j = 0$. If we make the operation total by defining arbitrarily $0^{-1} = 0$ we also have to convert each program into a new one by replacing each instruction

k: <u>do</u> $x_i := (x_j)^{-1}$ <u>then go to</u> p

by the subroutine

k: <u>if</u> $x_j = 0$ <u>then go to</u> k <u>else go to</u> k';

k': <u>do</u> $x_i := (x_j)^{-1}$ <u>then go to</u> p

where k' is a label that did not occur in the original program.

With this slight change the work of [1] applies to the field of real numbers. In particular, we have an effective method for formulating for each program π over $\mathfrak{B}$ a sentence in (a fragment of)

$L_{\omega_1,\omega}$ that expresses exactly the property of $\mathfrak{R}$ verified by the termination of π. The algorithmic theory of $\mathfrak{R}$ is the set of all such sentences (or their negations) holding in $\mathfrak{R}$. Our goal is to axiomatize this theory in the framework of $L_{\omega_1,\omega}$.

For this purpose we first collect a few outstanding properties, formulated in $L_{\omega_1,\omega}$, which a structure

$$\mathfrak{A} = \langle A, \leq, +, \cdot, -, ^{-1}, 0, 1\rangle$$

must have in order to be algorithmically equivalent to $\mathfrak{R}$.

LEMMA 2.1. $\mathfrak{A}$ is an archimedian ordered field.

It is easy to write programs over $\mathfrak{R}$ that verify the finitely-many axioms of archimedian ordered fields. For example, the following program verifies the archimedian property:

1: do $x_0 := 0$ then go to 2;

2: do $x_3 := x_2$ then go to 3;

3: if $x_1 \leq x_0$ then go to 7 else go to 4;

4: if $x_2 \leq x_0$ then go to 7 else go to 5;

5: if $x_1 \leq x_2$ then go to 7 else go to 6;

6: do $x_2 := x_2 + x_3$ then go to 5.

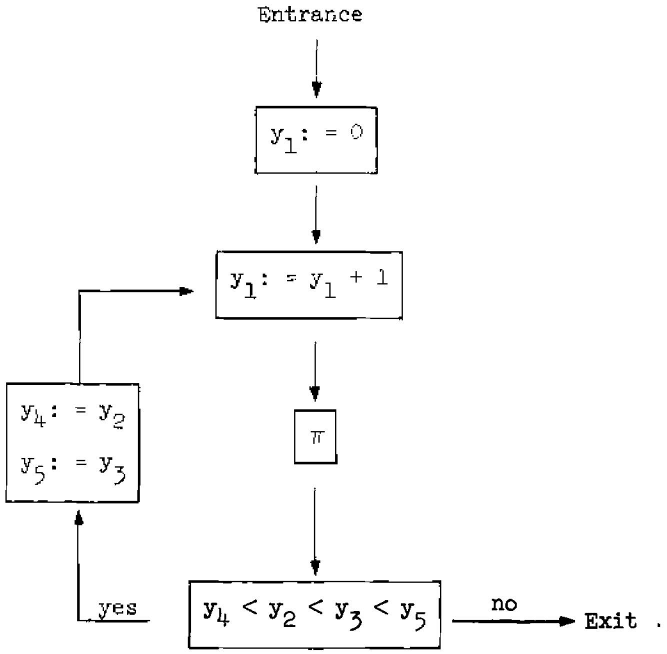
The axioms for an ordered field are first-order and hence a fortiori formulable in $L_{\omega_1,\omega}$; the archimedian property is

$$(\forall x)(\forall y)(x > 0 \wedge y > 0 \rightarrow \bigvee_{i=1}^{\infty} (x \leq \underbrace{y + y + \cdots + y}_{i \text{ times}}))$$

which is a formula in $L_{\omega_1,\omega}$.

LEMMA 2.2. $\mathfrak{A}$ is real-closed.

There are well-known numerical procedures that approximate the square root of a nonnegative real number or approximate a root of a polynomial of odd degree (with given coefficients). These procedures can easily be transformed into programs that compute a sequence of nested intervals (by computing the endpoints) that converge towards the real root. Such a program will look as follows:

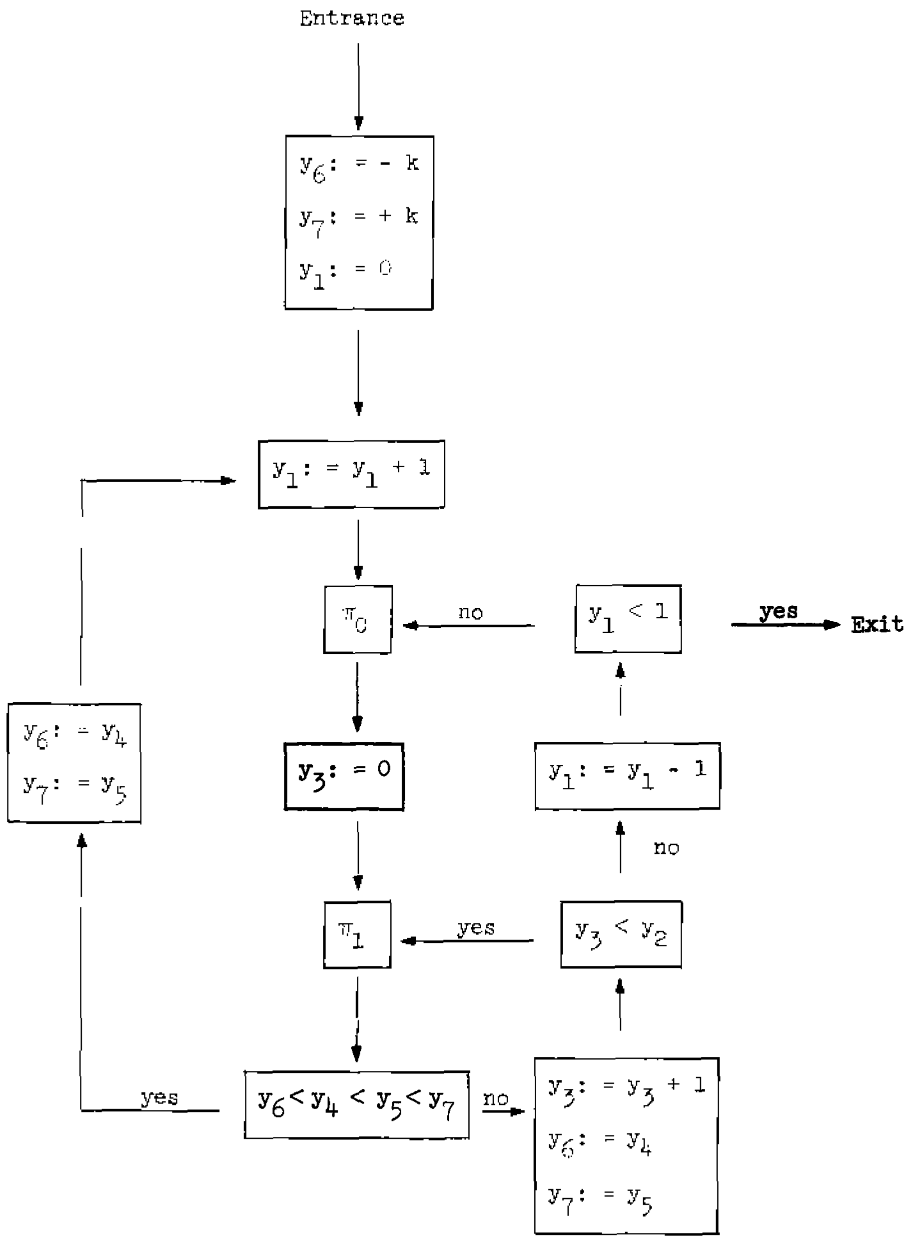


For successive values 1,2,3,... of y_1 the values of y_2, y_3 will form a converging sequence of nested intervals. Convert this program by replacing the box $y_4 < y_2 < y_3 < y_5$ by $y_4 < y_2 < z < y_3 < y_5$ where z did not occur in the original program. The converted program will not terminate if executed in $\mathfrak{R}$ when we assign the real root z as value to z, but will terminate in $\mathfrak{A}$ for all assignments for z in case the root in question does not exist in $\mathfrak{A}$. For $\mathfrak{A}$, being archimedian, is a subfield of $\mathfrak{R}$; if for some z in $\mathfrak{A}$ the program did not terminate in $\mathfrak{A}$ the limit point would equal z would be in $\mathfrak{A}$, and would be a root. Since the notion of a real closed field is formulable in first-order logic, it is, a fortiori, formulable in $L_{\omega_1,\omega}$.

LEMMA 2.3. $\mathfrak{A}$ <u>is computationally closed</u>.

By this we mean the following: Suppose we are given n elements $a_1,\ldots,a_n$ of $\mathfrak{A}$ and a program π over $\mathfrak{A}$ that computes, in terms of $a_1,\ldots,a_n$ a nested sequence of intervals. Then $\mathfrak{A}$ contains a point in the intersection. We shall need a slightly more general notion of computing a converging sequence of intervals than the one used above. Our more general notion uses the idea of "backtrack".

A program π computes a converging sequence of intervals in the following sense: Let π_0 be a one-exit program which computes y_2 in terms of $x_1,\ldots,x_n,y_1$ without changing $x_1,\ldots,x_n,y_1,y_3,y_4,y_5,y_6,y_7$. Let π_1 be a one-exit program which computes y_4 and y_5 in terms of $x_1,\ldots,x_n,y_1,y_3,y_6,y_7$ without changing $x_1,\ldots,x_n,y_1,y_2,y_3,y_6,y_7$. Then π is of the following form (in terms of flow-diagrams):



REMARK. To understand the above diagram, consider it first with the right hand side replaced by an exit. The resulting program clearly determines at each turn (i.e., for successive values of y_1) a decreasing sequence of nested intervals. The added right hand side provides for a "backtrack".

The program π computes a converging sequence of intervals if for some k the program does not terminate. If $\varphi(k,x_1,\ldots,x_n)$ expresses that π terminates for $k,x_1,\ldots,x_n$ then $\bigvee_{k=1}^{\infty} \neg \varphi(k,x_1,\ldots,x_n)$ expresses that the program computes a converging sequence of intervals if started on $x_1,\ldots,x_n$.

If in the above program the box $y_6 < y_4 < y_5 < y_7$ is replaced by $y_6 < y_4 < z < y_5 < y_7$ and $\psi(k,z,x_1,\ldots,x_n)$ expresses that the modified program terminates for $k,z,x_1,\ldots,x_n$ then $\bigvee_{k=1}^{\infty} \neg \psi(k,z,x_1,\ldots,x_n)$ expresses that z lies in the intersection of the converging sequence of intervals computed by π. Thus the formula

$$(\forall x_1) \cdots (\forall x_n)\left(\bigvee_{k=1}^{\infty} \neg \varphi(k,x_1,\ldots,x_n) \rightarrow (\exists z) \bigvee_{k=1}^{\infty} \neg \psi(k,z,x_1,\ldots,x_n)\right)$$

states, in the language $L_{\omega_1,\omega}$, that if π computes a converging sequence of intervals for some $x_1,\ldots,x_n$ then there is a point in the intersection. Now, if we associate such a sentence to each program π of the form indicated, we obtain a characterization of the notion of computational closure by a (recursive) set of formulas of $L_{\omega_1,\omega}$.

Results 2.1 and 2.3 together establish:

THEOREM 2.4. If $\mathfrak{A}$ is algorithmically equivalent to $\mathfrak{R}$ then $\mathfrak{A}$ is a computationally closed archimedian ordered field.

The remainder of this section is devoted to a proof of the converse of this theorem.

THEOREM 2.5. If $\mathfrak{A}$ is a computationally closed archimedian ordered field then $\mathfrak{A}$ is algorithmically equivalent to $\mathfrak{R}$.

Proof. Any archimedian ordered field $\mathfrak{F}$ may be considered as a subfield of $\mathfrak{R}$. By the computationally closure of $\mathfrak{F}$ in R we mean the smallest subfield of $\mathfrak{R}$ that contains $\mathfrak{F}$ and is computationally closed. Obviously, this field may be obtained by iterating the adjunction of elements of $\mathfrak{R}$ determined by computable convergent sequences of intervals. In particular, let $\mathfrak{C}$ be the computational closure of the field of rational numbers. We shall show that all computationally closed archimedian ordered fields $\mathfrak{A}$ are algorithmically equivalent to $\mathfrak{C}$.

Note that $\mathfrak{C}$ is a subfield of $\mathfrak{A}$, hence if φ is an algorithmic property of $\mathfrak{A}$ then it is one of $\mathfrak{C}$ (φ being a universal formula of $L_{\omega_1,\omega}$). It remains to show that, conversely, every algorithmic property of $\mathfrak{C}$ is one of $\mathfrak{A}$. For this it is sufficient to prove that for each algorithmic property $\varphi(x_1,\ldots,x_n)$ and for each $a_1,\ldots,a_n$ in $\mathfrak{A}$ with $\mathfrak{A} \models \neg\varphi[a_1,\ldots,a_n]$ there are $b_1,\ldots,b_n$ in $\mathfrak{C}$ such that $\mathfrak{C} \models \neg\varphi[b_1,\ldots,b_n]$.

By [1] the formula $\neg\varphi$ is obtained from atomic formulas and negated atomic formulas by the following syntactical operations: conjunction, disjunction with a negated or unnegated atomic formula, primitive substitution $\mathrm{Sub}_k(\psi)$, and conjunctions $\bigwedge_{w\in|\sigma^*|} \mathrm{Sub}_w(\psi)$. First observe that every formula $\neg\varphi$ is logically equivalent to a <u>reduced</u> formula of the form $\bigwedge_{w\in|\sigma|} \mathrm{Sub}_w(\psi)$ where σ is the signature of an appropriate program and ψ is a (finite) Boolean combination of atomic formulas. This is easily established by induction on the structure of $\neg\varphi$, (using the fact that we have infinitely many variables that we can use as dummies). The procedure that determines a point in $\mathfrak{C}^n$ which satisfies a reduced formula $\bigwedge_{w\in|\sigma|} \mathrm{Sub}_w(\psi)$ can be outlined informally as follows: To determine a computable first coordinate consider for each m the formula $(\exists x_2)\cdots(\exists x_n)\psi^{(m)}$, where $\psi^{(m)}$ is the conjunction of the first m conjuncts in $\bigwedge_{w\in|\sigma|} \mathrm{Sub}_w(\psi)$. Since $\mathfrak{A}$ is real closed, Tarski's decision method applies, and the set of all x_1 satisfying this formula is a finite collection of intervals (open, closed, half-open, or degenerated to a point) with algebraic endpoints. The formulas $(\exists x_2)\cdots(\exists x_n)\psi^{(m)}$, $m = 1,2,\ldots,$ thus determine a convergent sequence of intervals (which can be found by back-tracking of the kind described in the proof of Lemma 2.3). The points of the intersection of such a sequence satisfy $(\exists x_2)\cdots(\exists x_n) \bigwedge_{w\in|\sigma|} \mathrm{Sub}_w(\psi)$. Without going into the tedious details, it is clear that the first coordinate is then obtained by a computable convergent sequence of intervals. Using the first coordinate, we obtain similarly the second, etc.

THEOREM 2.6. <u>The algorithmic theory of</u> $\mathfrak{R}$ <u>is undecidable.</u>

<u>Proof</u>. Since the algorithmic theory of the natural numbers $\mathfrak{n} = \langle N, 0, \text{Successor}\rangle$ is undecidable, [2], it suffices to give a relative interpretation of the algorithmic theory of $\mathfrak{n}$ into that of $\mathfrak{R}$. For this, observe that the program

1: <u>do</u> $x_1 := 0$ <u>then go to</u> 2;

2: <u>do</u> $x_2 := 1$ <u>then go to</u> 3;

3: <u>if</u> $x_1 := x_0$ <u>then go to</u> 5 <u>else go to</u> 4;

4: <u>do</u> $x_1 := x_1 + x_2$ <u>then go to</u> 3, terminates exactly when started on an assignment of a natural number to x_0. Thus the algorithmic property associated to this program defines the concept of natural number.

Theorems 2.4, 2.5, and 2.6 together establish the main result of this section:

THEOREM 2.8. The algorithmic theory of computationally closed archimedian ordered fields is complete, axiomatizable, and undecidable; its models are exactly those ordered fields that are algorithmically equiavlent to the field of real numbers.

Note that, in contrast, the set of first-order consequences of the set of axioms for computationally closed archimedian ordered fields is decidable (since it is the theory of real closed fields for which we have Tarski's decision procedure). Note also that the requirement of archimedian order may be dropped since it follows easily from computational closure.

3. Miscellaneous remarks.

1. Let us call the algorithmic theory of a geometry its "construction theory." If we now define the notion of a constructively closed Euclidean geometry in the obvious manner analogous to the definition of computationally closed fields, we obtain the following consequence of Theorem 2.7 and Corollary 1.5:

THEOREM. The construction theory of constructively closed Euclidean geometries is complete, axiomatizable, and undecidable; its models are exactly those geometries which are constructively equivalent to the real Euclidean plane.

2. The definition of the notion of construction problem is left implicit in the literature. Most modern writers seem to favor a notion which we would call elementary construction problem and which could be formally defined thus: Suppose that φ is a quantifier-free formula of elementary geometry (i.e., a first-order formula):

$$\varphi\left(P_1,\ldots,P_n,\ell_1,\ldots,\ell_n,\gamma_1,\ldots,\gamma_n,P_{n+1},\ldots,P_{n+m},\ell_{n+1},\ldots,\ell_{n+m},\gamma_{n+1},\ldots,\gamma_{n+m}\right).$$

Then the elementary construction problem associated to φ consists in finding a program π over A(1) - B(8) such that π computes values for $P_{n+1},\ldots,\gamma_{n+m}$ from initial values for $P_1,\ldots,\gamma_n$ such that these values together satisfy the formula φ. Some authors even restrict themselves further by allowing only such φ that, translated into analytical terms, are equivalent to a single algebraic equation. Note that, for example, the problem of the quadrature of the circle is not an elementary construction problem. (It is, however, a construction problem in the sense of the present paper, where we may take for φ any formula of construction theory.) For the deductive theory $\mathcal{E}_2$" which is adequate for the elementary (in the above sense) theory of geometrical constructions see Tarski [5]; it is not adequate for all of traditional construction theory. For example, Kijne [4] shows that the Mohr-Mascheroni construction theorem does not hold in all models of $\mathcal{E}_2$".

3. By the well-known algebraic methods, it is possible to prove that there is a decision procedure for the problem whether a given elementary construction problem can be solved. In contrast, there is no decision procedure for the general construction problem in which φ is an arbitrary formula of construction theory.

4. In a semi-formal exposition of the theory of geometrical constructions, Kijne [4] uses two additional kinds of operations.[2] The selection operation S_1 and the adjunction operation A^g. The need for the first seems to arise there because of the lack of definiteness in the formulation of construction programs. The adjunction operation is more troublesome. Its intuitive counterpart is illustrated by: "... now, let P_i be any point interior to the given triangle, ..." With the aid of the fixed coordinate lines x, y and basis points $0, E_x$ and E_y, we are able to circumvent the use of adjunction operations in all constructions that are independent of the choice of the adjoined element (within the class of possible choices). The proof of this is well-known, [3].

REFERENCES

[1] Engeler, E., Algorithmic properties of structures, Math. Systems Theory, 1 (1967), 183-195.

[2] __________, Formal Languages: Automata and Structures, Markham Publishing Company, Chicago, (1968), viii + 81 pp.

[3] Enriques, F. (edit.), Fragen der Elementargeometrie, vol. 2, B. G. Teuber Verlag, Leipzig (1923), p. 116.

[4] Kijne, D., Plane construction field theory, Ph.D. Thesis, University of Utrecht, (1956), vi + 118 pp.

[5] Tarski, A. What is elementary geometry, The Axiomatic Method, ed. by L. Henkin, P. Suppes, and A. Tarski, North Holland Publishing Company, Amsterdam (1959), 16-29.

UNIVERSITY OF MINNESOTA

[2] We thank Professor H. Guggenheimer for drawing our attention to the work of Kijne.