Interpreting a Classical Geometric Proof with Interactive Realizability

Giovanni Birolo
Department of Mathematics
University of Turin
Italy
giovanni.birolo@gmail.com

We show how to extract a monotonic learning algorithm from a classical proof of a geometric statement by interpreting the proof by means of interactive realizability, a realizability sematics for classical logic.

The statement is about the existence of a convex angle including a finite collections of points in the real plane and it is related to the existence of a convex hull. We define real numbers as Cauchy sequences of rational numbers, therefore equality and ordering are not decidable. While the proof looks superficially constructive, it employs classical reasoning to handle undecidable comparisons between real numbers, making the underlying algorithm non-effective.

The interactive realizability interpretation transforms the non-effective linear algorithm described by the proof into an effective one that uses backtracking to learn from its mistakes. The effective algorithm exhibits a "smart" behavior, performing comparisons only up to the precision required to prove the final statement. This behavior is not explicitly planned but arises from the interactive interpretation of comparisons between Cauchy sequences.

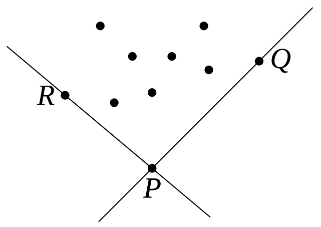
1 Introduction

Interactive realizability is a realizability semantics that extends the Brouwer-Heyting-Kolmogorov interpretation to (sub-)classical logic, more precisely to first-order intuitionistic arithmetic (Heyting Arithmetic, HA) extended by the law of the excluded middle restricted to Σ^0_1 formulas (EM_1), a system motivated by its applications in proof mining. It was introduced by Berardi and de'Liguoro in [2].

We use interactive realizability in order to study the computational content of a classical proof of the following geometric statement.

Theorem (Convex Angle)**.**
We have a finite set of at least three points in the real plane $\mathbb{R}^2$ such that no three points are on the same line. Then there exist distinct points P, Q and R such that:

- *all other points S are inside $\widehat{QPR}$,*
- *the angle $\widehat{QPR}$ is convex, that is, less than π.*



We choose this particular statement because we have a proof of it that looks algorithmic and can be easily visualized. Theorem 1 can be thought of as weakened version of the existence of the convex hull of a finite set of points.

As we said the proof we choose as example looks constructive, using only decidability of ordering over real numbers. However, it is well known that there is no effective ordering on the real numbers. In our encoding of the real numbers, totality of the ordering on the recursive reals is equivalent to EM_1.

Since the proof needs the ordering to be total, it needs EM_1. Due to the low logical complexity of excluded middle which is used, the proof may be interpreted with a simple case of interactive realizability.

We show how interactive realizability can be applied and what it can tell us about the computational content of the proof. What we get is an algorithm that, instead of comparing real numbers, makes an arbitrary guess about which one is smaller. If later it becomes apparent that the guess is wrong the algorithm retracts the choice it made since it can now make an informed decision about that particular comparison. Then the algorithm performs comparisons only when needed and only up to the required precision.

Thus we see how a simple classical proof which performs comparisons between real numbers is interpreted as a learning algorithm which uses “educated guesses” in order to avoid non effective operations. This non-trivial behavior is not explicit in the classical proof, but follows from the definition of ordering on Cauchy sequences by means of the interactive realizability interpretation.

In the present work, our main goal is to showcase interactive realizability and the backtracking algorithms it produces through a non-trivial example. For this reason, we chose to present interactive realizability as a proof interpretation technique rather than as a realizability semantics, in order to concentrate on the example and its computational interpretation without being bogged down in technical details. A more comprehensive treatment of interactive realizability can be found for instance in [1].

2 Real Numbers

In this section we present our treatment of real numbers in Heyting Arithmetic. For a more in depth treatment of real numbers from a constructive view point see [4].

There are many ways of encoding integer and rational numbers in HA and defining primitive recursive operations and predicates on them. In the following we assume that we have any such encoding and that we have decidable equality $=_{\mathbb{Q}}$ and ordering $<_{\mathbb{Q}}, \leq_{\mathbb{Q}}$ and effective operations $+_{\mathbb{Q}}, \cdot_{\mathbb{Q}}$. We use the variables q and p for rationals.

There are many equivalent ways of defining the real numbers from the rational numbers. The best known are the definition of the reals as equivalence classes of Cauchy sequences and as Dedekind cuts. We follow the first approach.

A sequence of rationals $r : \mathbb{N} \to \mathbb{Q}$ is a *Cauchy sequence* if the following holds:

$$\forall k.\ \exists k_0.\ \forall k_1, k_2.\ |r(k_0 + k_2) - r(k_0 + k_1)| < \frac{1}{2^k}. \tag{1}$$

While this sequence approximates a real number, it can do so very slowly. By means of classical reasoning, we can show that, from any Cauchy sequence, we can extract a fast-converging monotone subsequence. For this reason, instead of general Cauchy sequences, we can consider sequences of nested intervals with rational extremes whose length decreases exponentially. An interval is determined by its extremes, so we represent a sequence of intervals as a couple of sequences of rationals r^-, r^+, representing the lower and higher extremes of the intervals respectively. Then we require that r^- is increasing and r^+ is decreasing (since the intervals are nested), that $r^-(k)$ is lesser than or equal to $r^+(k)$ (since they are the lower and higher extremes of a same interval) and their difference is smaller than 2^{-k}. More precisely we say that r^- and r^+ represent a real number when they satisfy the following condition, written as a Π^0_1 formula:

$$\begin{aligned}\forall k.\ (r^-(k) \leq_{\mathbb{Q}} r^+(k)) \wedge (r^-(k) \leq_{\mathbb{Q}} r^-(k+1)) \wedge \\ \wedge (r^+(k) \geq_{\mathbb{Q}} r^+(k+1)) \wedge (r^+(k) -_{\mathbb{Q}} r^-(k) \leq_{\mathbb{Q}} 2^{(-k)}).\end{aligned} \tag{2}$$

While the choice of the specific definition of real number is somewhat arbitrary, it is significant because it affects the logical properties (in particular the degree of undecidability) of the ordering on the reals.

Now we can define an "order predicate" $\mathrm{OP}(r,s,k)$, which can be thought of as a family of strict partial orders on the real numbers indexed by natural number k. More precisely, it is a formula that determines when the sequence of nested intervals r is strictly lesser than s, at precision k. This happens when, at k, the higher extreme of an interval is strictly greater than the lower extreme of the other. Then, from that point forward, the intervals will be forever disjoint, since they are nested sequences. This allows us to write the order predicate as the formula:

$$\mathrm{OP}(r,s,k) \equiv r^+(k) <_{\mathbb{Q}} s^-(k), \tag{3}$$

which is decidable in r and s. Note that the definition of OP depends on that of real number. If we had used the classical definition of Cauchy sequence the order predicate would be the following Π^0_1 formula:

$$\mathrm{OP}'(r,s,k) \equiv \forall l.\ l \geq k \to r(l) <_{\mathbb{Q}} r(l). \tag{4}$$

This is very significant for our purposes: the order predicate in (3) is decidable in r and s (since the order on the rationals is), while in (4) it is only *negatively decidable*. This means that we have an effective method to decide (4) when it is false, but not when it is true.

We need OP to satisfy the following properties, written as rules:

$$\begin{array}{ll}
\text{OP-mon}\ \dfrac{\mathrm{OP}(r,s,k)}{\mathrm{OP}(r,s,k+1)} & \text{OP-irrefl}\ \dfrac{\mathrm{OP}(r,r,k)}{\perp} \\[2ex]
\text{OP-asym}\ \dfrac{\mathrm{OP}(r,s,k) \qquad \mathrm{OP}(s,r,l)}{\perp} & \text{OP-trans}\ \dfrac{\mathrm{OP}(r,s,k) \qquad \mathrm{OP}(s,t,l)}{\mathrm{OP}(r,t,\max(k,l))}
\end{array} \tag{5}$$

The OP-mon rule expresses a monotonicity property: when an comparison at a given precision can distinguish two approximations, then comparisons at greater precision should too. The other rules correspond to the standard axioms for a strict partial order: irreflexivity, asymmetry and transitivity.

We verify that our definition of OP satisfies these properties.

Lemma 1. *The order predicate OP defined by* (3) *satisfies the properties given in* (5).

Proof. Omitted. The properties follow directly from the definition of OP as (3) and from our representation of real number as sequences of nested intervals (2). ◻

We can now define order and equality on the reals. It is noteworthy that, while we define order and equality in terms of OP, we never use the definition of OP itself in proving their properties. We only need the properties of OP we proved in Lemma 1, thus we could proceed in the same way even if we had defined OP differently, as long as Lemma 1 holds.

They are defined as follows:

$$\begin{array}{ll}
r <_{\mathbb{R}} s \equiv \exists k.\ \mathrm{OP}(r,s,k), & r \leq_{\mathbb{R}} s \equiv \forall k.\ \neg\mathrm{OP}(s,r,k), \\
r \neq_{\mathbb{R}} s \equiv \exists k.\ \mathrm{OP}(r,s,k) \vee \mathrm{OP}(s,r,k), & r =_{\mathbb{R}} s \equiv \forall k.\ \neg\mathrm{OP}(r,s,k) \wedge \neg\mathrm{OP}(s,r,k).
\end{array}$$

Note that $<_{\mathbb{R}}$ and $\neq_{\mathbb{R}}$ are Σ^0_1 formulas and $\leq_{\mathbb{R}}$ and $=_{\mathbb{R}}$ are Π^0_1 formulas.

In order to prove Lemma 3, which is needed in the proof of Theorem 1, we need to show some of the properties of the order $\leq_{\mathbb{R}}$.

Lemma 2 (Reflexivity, Semi-Transitivity and Totality of $\leq_{\mathbb{R}}$)**.** *The following properties hold:*

$$r \leq_{\mathbb{R}} r \qquad \text{(reflexivity)}$$
$$r <_{\mathbb{R}} s \wedge s \leq_{\mathbb{R}} t \rightarrow r \leq_{\mathbb{R}} t, \qquad \text{(semi-transitivity)}$$
$$r \leq_{\mathbb{R}} s \vee s <_{\mathbb{R}} r. \qquad \text{(totality)}$$

Proof. The first two properties follows from the corresponding properties of OP. The last is a classical tautology.

- We omit the proof of reflexivity for reasons of space and irrelevance.
- In order to prove this transitive property for mixed $<_{\mathbb{R}}$ and $\leq_{\mathbb{R}}$ we have to show that $r \leq_{\mathbb{R}} t \equiv \forall k.\ \neg\mathrm{OP}(t,r,k)$, assuming $r <_{\mathbb{R}} s \equiv \exists k.\ \mathrm{OP}(r,s,k)$ and $s \leq_{\mathbb{R}} t \equiv \forall k.\ \neg\mathrm{OP}(t,s,k)$. This follows by means of the OP-trans rule:

$$\exists\mathrm{E}\,\dfrac{\exists k.\ \mathrm{OP}(r,s,k) \qquad {\to}\,\mathrm{E}\,\dfrac{\forall\mathrm{E}\,\dfrac{\forall k.\ \neg\mathrm{OP}(t,s,k)}{\neg\mathrm{OP}(t,s,\max(k,l))} \qquad \text{OP-trans}\,\dfrac{[\mathrm{OP}(t,r,k)]^1 \qquad [\mathrm{OP}(r,s,l)]^2}{\mathrm{OP}(t,s,\max(k,l))}}{\perp}}{\forall\mathrm{I}\,\dfrac{{\to}\,\mathrm{I}\,\dfrac{\perp}{\neg\mathrm{OP}(t,r,k)}\,1}{\forall k.\ \neg\mathrm{OP}(t,r,k)}}\,2$$

- When r and s denote recursive real numbers, totality is an instance of EM_1:

$$r \leq_{\mathbb{R}} s \vee s <_{\mathbb{R}} r \equiv \forall k.\ \neg\mathrm{OP}(s,r,k) \vee \exists k.\ \mathrm{OP}(r,s,k). \qquad \square$$

The proof is constructive apart from the last point, where we show that totality is actually an instance of EM_1. Note that only the reflexivity property is stated in the standard way, while transitivity and totality are written in non-standard forms. We chose these forms for two reasons: they are easier to prove and they are the exact forms we need in the proof of Lemma 3.

Until now we have used r, s and t as metavariables for real numbers in an informal way. However, since we are working in the first-order language of arithmetic, our variables range only on natural numbers and not on functions. For our example we only need to address a finite but arbitrary number of real numbers, that is, we only need a countable quantity of them. Thus we can assume that we have a countable set of pairs of function symbols indexed by the natural numbers, say $(f_n^+, f_n^-)_{n\in\mathbb{N}}$. We assume that each pair satisfies the convergence condition (2) and thus represents a real number. Then, OP can be formally defined as $f_i^+(k) <_{\mathbb{Q}} f_j^-(k)$ where i and j are arithmetic terms. Thus each real numbers is represented by a natural number, namely its index. For convenience and consistency with the standard notation for real numbers, instead of writing $i <_{\mathbb{R}} j$, we use the sugared version $r_i \leq_{\mathbb{R}} r_j$.

Now we can reason about finite sets of real numbers as sets of indexes. In the next lemma, we shall work with the sets of real numbers indexed by initial segments of the natural numbers. We show the existence of a least element in each of these sets. The least element is actually a minimum, that is, the unique least element of the set. However, in order to prove Theorem 1 we do not need to show its uniqueness, just its existence.

Lemma 3 (Least Element)**.** *For any n, the real numbers $r_0, \ldots, r_n$ have a least element with respect to $\leq_{\mathbb{R}}$. More precisely:*

$$\forall n.\ \exists i \leq n.\ \forall j \leq n.\ r_i \leq_{\mathbb{R}} r_j.$$

Proof. We proceed by induction on n.

Zero case In the base case $n = 0$ and we have to prove that $\exists i \leq 0.\ \forall j \leq 0.\ r_i \leq_{\mathbb{R}} r_j$. Both i and j can only be 0; thus we just have to check the condition $r_0 \leq_{\mathbb{R}} r_0$, which holds by reflexivity of $\leq_{\mathbb{R}}$.

Successor case In the inductive case we have to prove that $\exists i \leq n+1.\ \forall j \leq n+1.\ r_i \leq_{\mathbb{R}} r_j$. By the inductive hypothesis, let $\overline{i} \leq n$ be the index of the least element in $r_0, \ldots, r_n$. By totality of $\leq_{\mathbb{R}}$ we have two cases.

$r_{\overline{i}} \leq_{\mathbb{R}} r_{n+1}$ Then $\overline{i}$ is the index of a least element in $r_0, \ldots, r_{n+1}$, since $r_{\overline{i}} \leq_{\mathbb{R}} r_j$ when $j = n+1$ (since we are considering this case) and when $j \leq n$ by inductive hypothesis.

$r_{n+1} <_{\mathbb{R}} r_{\overline{i}}$ Then $n+1$ is the index of a least element in $r_0, \ldots, r_{n+1}$, since $r_{n+1} \leq_{\mathbb{R}} r_j$ when $j = n+1$ by reflexivity of $\leq_{\mathbb{R}}$ and when $j \leq n$ by transitivity of $<_{\mathbb{R}}$ and $\leq_{\mathbb{R}}$, since $r_{n+1} <_{\mathbb{R}} r_{\overline{i}} \leq_{\mathbb{R}} r_j$ by inductive hypothesis. □

The proof looks constructive: its computational interpretation is the usual algorithm that finds the least element in a vector, by a simple recursion or by looping on its elements. We can write it as a recursive function "`rmin`" in Haskell:

Listing 1: The Least Element Program

```
rmin 0    = 0
rmin n    = if rle (rmin (n-1)) n
  then rmin (n-1)
  else n
```

where "`rle`" is a boolean function that stands for $\leq_{\mathbb{R}}$, that is, it compares the reals indexed by its arguments. The problem is that we are unable to write "`rle`" as a terminating program. The closest approximation would be the following unfounded recursion:

Listing 2: The Lesser or Equal Program

```
rle i j        = rle_urec 0 i j
rle_urec k i j = if op j i k end
  then False
  else rle_urec (k+1) i j
```

where "`op`" is a total boolean function that stands for the order predicate OP. We can assume that "`op`" terminates for any input since OP is decidable. The problem is that $\leq_{\mathbb{R}}$ is total only classically. More precisely, totality is an instance of EM_1 because $\leq_{\mathbb{R}}$ is a Π^0_1 formula and thus negatively decidable. This can bee seen concretely in the program for "`rle`": "`rle i j`" only halts (returning "**`False`**") if "`op j i k`" is true for some k, that is, if and only if $r_i \leq r_j$ is false. On the other hand, when $r_i \leq r_j$ is true there is no such k and the evaluation of "`rle i j`" will never halt: "**`True`**" does not even appear in the program. This is the general behavior of an algorithm that computes a negatively decidable predicate: when the predicate is false it halts with the correct answer and when the predicate is true it does not halt.

For positively decidable predicates we have the dual behavior. For instance, in the case of $<_{\mathbb{R}}$ which is defined by a Σ^0_1 formula and thus positively decidable, the decision procedure can be written as:

Listing 3: The Lesser Than Program

```
rlt i j        = rlt_urec 0 i j
rlt_urec k i j = if op i j k
  then True
  else rlt_urec (k+1) i j
```

The program is very similar to the previous one, the only noteworthy changes are the order of the argument given to "`op`" and the fact that the only possible return value is "**`True`**" instead of "**`False`**". It only halts (returning "**`True`**") if "`op i j k`" is true for some k, that is, when $r_i \leq r_j$ is true.

3 The Interactive Interpretation of the Least Element Lemma

We have seen why the naive way of extracting a program from proofs fails in the case of Lemma 3. Now we give the interactive interpretation of Lemma 3. Since we are working in $\mathsf{HA}+\mathsf{EM}_1$, any proof can be thought of as a constructive proof with open assumptions, that are the instances of EM_1 that are used in the proof. The interactive realizability interpretation follows the standard BHK interpretation for the constructive parts, so we will concentrate on the interpretation of the EM_1 instances.

The only instances of EM_1 in the proof are those used to deduce the totality property:

$$r_i \leq_{\mathbb{R}} r_j \vee r_j <_{\mathbb{R}} r_i. \tag{6}$$

The left disjunct, which we call the *universal disjunct*, is Π^0_1 and negatively decidable, while the right one, the *existential disjunct* is Σ^0_1 and positively decidable. Moreover universal disjunct and negation of the existential disjunct are classically equivalent. We say that a formula is *concrete* when it is closed and atomic.

A naive attempt to give a computational content to an EM_1 instance fails, because in general EM_1 instances are undecidable. Interactive realizability proposes a way to side-step this problem. This is possible since it is not true that the computational interpretation of a proof using instances of EM_1 necessarily needs to decide these instances. Consider the case of totality of the order on the real numbers. The universal disjunct is $r_i \leq_{\mathbb{R}} r_j \equiv \forall k.\ \neg\mathrm{OP}(r_j, r_i, k)$. Being an universally quantified statement, it proves infinite instances $\neg\mathrm{OP}(r_j, r_i, k)$, one for each natural number k. A proof that uses totality may need all this infinite information or (for example, when proving a simply existential statement) may only need a finite quantity of these instances. In the second case, we can avoid the problem of effectively deciding the EM_1 instance. We only need to decide those instances that are actually used in the proof. This is possible, since each instance is decidable (being a quantifier free formula) and we assumed there is a finite quantity of them. Interactive realizability takes advantage of this fact and gives a procedure to determine which instances of the universal disjunct are needed and to iteratively decide them.

The interactive interpretation is a "relaxation" of the BHK interpretation. In the BHK interpretation the decision of a disjunction effectively selects a true disjunct, in the interactive case instead of a decision we have a sort of "educated guess". Therefore, while EM_1 cannot be realized by the BHK interpretation since there is no effective procedure to decide it, the interactive interpretation can because it yields a weaker semantics, which produces a sure result only when the goal is simply existential.

Interactive realizability revolves around the concept of *knowledge state*. A knowledge state, or simply state, is a finite object that stores information about the EM_1 instances we use in the proof. The purpose of this information is to help us decide the EM_1 instances, that is, help us in choosing which disjunct holds. Moreover, whenever the state chooses the existential disjunct, it should also produce a witness, like in the BHK interpretation.

We can represent a state as a finite partial function[1] that maps a concrete instance of EM_1 into a witness of its existential disjunct. Such a function decides or guesses a concrete instance A of EM_1: if it is undefined on A, then we choose the universal disjunct; if it is defined we chose the existential disjunct

[1]By finite partial function, we mean a partial function whose domain (the set of elements where it is defined) is finite.

with the returned witness. We are only interested in the instances appearing in the proof, namely, those of the form (6) when i, j are numerals. Thus an instance is determined by two natural numbers; since witnesses are natural numbers too, a state can be concretely defined as a finite partial function from $\mathbb{N}\times\mathbb{N}$ to $\mathbb{N}$.

For instance, consider the case of the EM_1 instances used in the proof of Lemma 3. When we have to decide (6), we check the state on the pair (i, j). At first, let us assume that the state is undefined on (i, j). This means we have no knowledge about the universal disjunct $r_i \leq_{\mathbb{R}} r_j$. Since we cannot effectively check that the universal disjunct holds, we make an educated guess and assume that $r_i \leq_{\mathbb{R}} r_j$ is true. Clearly this assumption could very well be wrong, which may or may not become apparent later in the proof. Keeping track of this assumption, we carry on with the proof. Every time we use this assumption to prove a decidable instance of its we check if the instance holds. More concretely, if later in the proof we use the assumption $r_i \leq_{\mathbb{R}} r_j$ to deduce that $\neg\mathrm{OP}(j,i,k)$ for some k, we check that $\neg\mathrm{OP}(j,i,k)$ holds. If this is the case, we carry on with the proof: $r_i \leq_{\mathbb{R}} r_j$ could still be false, but at least the particular instance we are using is true. If this is not the case, we have found a counterexample to the assumption $r_i \leq_{\mathbb{R}} r_j$: being negatively decidable, the counterexample is enough to effectively decide that it is false. Therefore we stop following the proof because we have chosen the wrong disjunct in the EM_1 instance (6).

Moreover, a counterexample to $r_i \leq_{\mathbb{R}} r_j$ is a natural number k such that $\mathrm{OP}(j,i,k)$. Therefore k is a witness for the existential disjunct $r_j <_{\mathbb{R}} r_i$. We can use this new knowledge to add (i, j) to the domain of the state with value k. Remember that we assumed the state to be undefined on (i, j), which is why we assumed the universal disjunct to be true in the first place.

At this point, we forget what we did after guessing (wrongly) that the universal disjunct was true and start again. More precisely, we need to backtrack to a computation state *before* we decided the EM_1 instance in question and repeat our decision with the extended state. Since the extended state is defined on (i, j) and yields k, this time we decide the EM_1 instance differently: we choose the existential disjunct $r_j <_{\mathbb{R}} r_i$ with k as witness. Now we are sure that our choice is the correct one and not a guess, since we have effectively decided that the existential disjunct holds (we can since it is positively decidable).

In order for the interactive interpretation to produce correct results, we need to assume that the state is sound, that is, when it is defined, the witness it yields is actually a witness. More formally, a state s is sound if, for any pair (i, j), we have that $\mathrm{OP}(j,i,s(i, j))$ holds. This assumption is not problematic: the empty state, namely the state that is always undefined, satisfies it vacuously. Moreover, in the interactive interpretation we outlined above, we only extend a state with an actual witness. In other words, the extension preserves the soundness property.

To summarize, the general procedure is the following:

1. we start from any sound state (usually the empty state),
2. we follow the proof choosing any EM_1 instance according to the state,
3. if we discover that we wrongly assumed the universal disjunct of an EM_1 instance:
    (a) we extend the state with the counterexample we found,
    (b) we backtrack to a point before the EM_1 instance we guessed wrong,
    (c) we proceed as in step 2,
4. if we never discover that we wrongly assumed an universal disjunct we carry on until the end of the proof and we are done.

The exact point we need to backtrack to is not relevant, as long as it is before the decision of the EM_1 instance. A simple choice would be the very beginning, in which case we do not need to keep track of where we decided the EM_1 instance. In this case we only need a simple abort operator in order to formally write interactive realizers. A monadic version of this approach is given in [3]. A more efficient choice is to backtrack right before the decision point, so that we do not need to repeat the computations that took place before it, since they are not affected by the extension of the state. However this approach would require more sophisticated control operators.

Interactive realizability can be thought as a “smart”, albeit “partial”, decision algorithm for negatively decidable statements. This can be seen comparing it with the naive algorithm given in Listing 2. It is partial because a real decision is impossible, so it only considers a finite number of instances, unlike the unbounded recursion employed by the program in Listing 2. It is smart because it does not perform a blind search, trying in order all the natural numbers. Instead it uses the proof itself to find the counterexamples. There is a reasonable expectation that the ideas underlying the proof provide a more focused way of selecting counterexamples than a blind search (this of course depends on the proof itself).

Until now we considered a single instance of the EM_1 axiom, but little changes if there is more than one. We will return to this point later. In the proof of Lemma 3, one instance of EM_1 is used for each inductive step in the proof. When we interpret the proof with the empty state, for each of these instances we assume that the universal disjunct holds. Therefore the proof is interpreted as follows. In the base step we choose r_0. In the first inductive step, we have to decide the EM_1 instance $r_0 \leq_{\mathbb{R}} r_1 \vee r_1 <_{\mathbb{R}} r_0$. Since the state is empty, we assume that $r_0 \leq_{\mathbb{R}} r_1$. Thus we keep r_0 as the least element of r_0, r_1. In the second inductive step, we have to decide the EM_1 instance $r_0 \leq_{\mathbb{R}} r_2 \vee r_2 <_{\mathbb{R}} r_0$. Since the state is empty, we again assume that $r_0 \leq_{\mathbb{R}} r_2$. Thus we keep r_0 as the least element of r_0, r_1, r_2. At the end of the proof, we have assumed the following universal disjuncts:

$$r_0 \leq_{\mathbb{R}} r_1, r_0 \leq_{\mathbb{R}} r_2, \ldots, r_0 \leq_{\mathbb{R}} r_n. \tag{7}$$

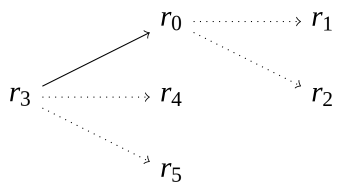
Under these assumptions, we have found that the least element is r_0. Rather disappointing, isn't it?

The reason for this is that the universal disjuncts $r_i \leq_{\mathbb{R}} r_j$ are never instanced, so we have neither opportunity nor reason to falsify one of them. However this may change if Lemma 3 is used inside a bigger proof. This will happen later in the proof of Theorem 1. In this case the outer proof might instance these assumptions and discover them wrong, in which case we have to backtrack to the proof of Lemma 3.

Let us see how Lemma 3 behaves when its conclusion is used to deduce decidable instances. Assume that $n = 5$. If the state is empty, then Lemma 3 tells us that r_0 is a least element. This means that $r_0 \leq_{\mathbb{R}} r_i$ for any i. Imagine that we use Lemma 3 in a bigger proof to prove that $r_0 \leq_{\mathbb{R}} r_3$. This is one of the EM_1 instances we assumed in (7). Moreover, imagine that, after instantiating this assumption, we discover that $r_0 \leq_{\mathbb{R}} r_3$ does not hold at precision 33, that is, $\mathrm{OP}(r_3, r_0, 33)$ holds. Then we have to extend the domain of the state to $(0,3)$ with value 33. At this point we backtrack, say at the beginning of the proof of Lemma 3.

We again start from r_0 and proceed like before. The first and second inductive steps again select r_0 as the least element, assuming that $r_0 \leq_{\mathbb{R}} r_1$ and $r_0 \leq_{\mathbb{R}} r_2$. Things change at the third inductive step when we have to decide $r_0 \leq_{\mathbb{R}} r_3 \vee r_3 <_{\mathbb{R}} r_0$. Since now the state has a relevant witness, this time we choose the existential disjunct with witness 33, thus selecting r_3 as the new least element. In the next inductive steps we again assume the universal disjuncts $r_3 \leq_{\mathbb{R}} r_4$ and $r_3 \leq_{\mathbb{R}} r_5$, since the state has no information on them. Thus the least element is r_3. A summary of our decisions is represented in Figure 1. Now imagine that we were to discover

Figure 1: A graph showing the result of the least element computation


$$\begin{tikzcd}
 & r_0 \arrow[r, dotted] \arrow[dr, dotted] & r_1 \\
r_3 \arrow[ur] \arrow[r, dotted] \arrow[dr, dotted] & r_4 & r_2 \\
 & r_5 &
\end{tikzcd}$$

Full arrows represent information provided by the state, dotted arrows “guessed” information the state knows nothing about.

a counterexample to $r_3 \leq_{\mathbb{R}} r_2$, say at precision 25. This statement is not one of the universal disjuncts that we assumed. By looking at the proof or at Figure 1, we can see that it has been deduced by the semi-transitivity property from $r_3 <_{\mathbb{R}} r_0$ and $r_0 \leq_{\mathbb{R}} r_2$. The first is the existential disjunct for which we found a witness, so we are sure that it holds. Thus the wrong assumption is $r_0 \leq_{\mathbb{R}} r_2$. By checking the proof of semi-transitivity we can see that the counterexample for $r_0 \leq_{\mathbb{R}} r_2$ is $\max(25,33)$, thus 33 again. We extend the state accordingly and repeat the least element computation, which results in new least element r_2. In Figure 2 we summarize the two iterations we saw until now and add some more, as an example.

Figure 2: An example of evaluations of the interactive interpretation of Lemma 3.

Iter	**State**	**Least element**	**Used**	**Deduced from**	**Discovered**
1*st*		$$\begin{tikzcd} r_0 \arrow[r, dash, dotted] & r_{1,\ldots,5} \end{tikzcd}$$	$r_0 \leq_{\mathbb{R}} r_3$	$r_0 \leq_{\mathbb{R}} r_3$	$r_3 <_{\mathbb{R}} r_0$
2*nd*	$r_3 <_{\mathbb{R}} r_0$	$$\begin{tikzcd} & r_0 \arrow[r, dash, dotted] & r_{1,2} \\ r_3 \arrow[ur, dash] \arrow[dr, dash, dotted] & & \\ & r_{4,5} & \end{tikzcd}$$	$r_3 \leq_{\mathbb{R}} r_2$	$r_3 <_{\mathbb{R}} r_0$ $r_0 \leq_{\mathbb{R}} r_2$	$r_2 <_{\mathbb{R}} r_0$
3*rd*	$r_2 <_{\mathbb{R}} r_0$ $r_3 <_{\mathbb{R}} r_0$	$$\begin{tikzcd} & r_0 \arrow[r, dash, dotted] & r_1 \\ r_2 \arrow[ur, dash] \arrow[dr, dash, dotted] & & \\ & r_{3,4,5} & \end{tikzcd}$$	$r_2 \leq_{\mathbb{R}} r_3$	$r_2 \leq_{\mathbb{R}} r_3$	$r_3 <_{\mathbb{R}} r_2$
4*th*	$r_2 <_{\mathbb{R}} r_0$ $r_3 <_{\mathbb{R}} r_0$ $r_3 <_{\mathbb{R}} r_2$	$$\begin{tikzcd} & r_2 \arrow[r, dash] & r_0 \arrow[r, dash, dotted] & r_1 \\ r_3 \arrow[ur, dash] \arrow[dr, dash, dotted] & & & \\ & r_{4,5} & & \end{tikzcd}$$	$r_3 \leq_{\mathbb{R}} r_1$	$r_3 <_{\mathbb{R}} r_2$ $r_2 <_{\mathbb{R}} r_0$ $r_0 \leq_{\mathbb{R}} r_1$	$r_1 <_{\mathbb{R}} r_0$
5*th*	$r_1 <_{\mathbb{R}} r_0$ $r_2 <_{\mathbb{R}} r_0$ $r_3 <_{\mathbb{R}} r_0$ $r_3 <_{\mathbb{R}} r_2$	$$\begin{tikzcd} & r_0 \\ r_1 \arrow[ur, dash] \arrow[dr, dash, dotted] & \\ & r_{2,3,4,5} \end{tikzcd}$$	$r_1 \leq_{\mathbb{R}} r_4$	$r_1 \leq_{\mathbb{R}} r_4$	$r_4 <_{\mathbb{R}} r_1$
6*th*	$r_1 <_{\mathbb{R}} r_0$ $r_2 <_{\mathbb{R}} r_0$ $r_3 <_{\mathbb{R}} r_0$ $r_3 <_{\mathbb{R}} r_2$ $r_4 <_{\mathbb{R}} r_1$	$$\begin{tikzcd} & & r_0 \\ & r_1 \arrow[ur, dash] \arrow[dr, dash, dotted] & \\ r_4 \arrow[ur, dash] \arrow[dr, dash, dotted] & & r_{2,3} \\ & r_5 & \end{tikzcd}$$	...	...	...

Iter: *the iteration represented by the current row,* **State:** *the existential disjuncts witnessed by the state,* **Least element:** *the least element yielded by Lemma 3,* **Used:** *a concrete consequence of Lemma 3 that is falsified in the proof,* **Deduced from:** *the premisses we deduced the falsified consequence from,* **Discovered:** *the existential assumption we found a witness of.*

In general we do not use all the information in the state in each iteration: for example, in the third iteration we do not use $r_3 <_{\mathbb{R}} r_0$, which we discovered in the first iteration and used in the second iteration. This happens because the state affects which instances of EM_1 are used in the proof, which should be apparent from the given iterations.

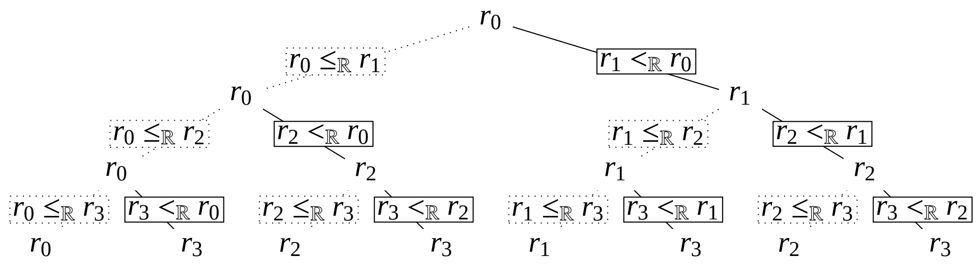
In the iterations listed in Figure 2, we compute the following sequence of least element candidates: $r_0, r_3, r_2, r_3, r_1, r_4$. The fact that r_3 appears two times may cause doubts regarding the termination of the backtracking algorithm. The termination of the backtracking algorithms in interactive realizability has been proven in general, see Theorem 2.15 in [1].

In this particular case we can understand why r_3 is computed two times by taking a closer look at the tree of the possible computations of the least element, which is shown in Figure 3. For reasons of space, we only show the tree for $n = 3$, which is enough to see what happens up to the fifth iteration in Figure 2. We can see that the first five iterations in Figure 2 correspond to the computation paths ending with the first five leaves from the left in Figure 3, in order.

Moreover, from the computation tree we can see that we never perform the same computation more

Figure 3: The computation tree of the least element for $n = 3$

.



Each path represents a possible computation, proceeding from root to leaf, where non-leaf nodes are the current least element candidates and the leaf is the final result. Each branching corresponds to an EM_1 *instance, where the left branch is taken when we guess that the universal disjunct holds for lack of information and the right branch is taken when the state contains the relevant witness.*

than once. Indeed, assume we have just followed a particular computation path. When we backtrack we increment the state adding a witness of one of the EM_1 instances we encountered along the path, an instance we did not have a witness for. This means that in the next computation, when we arrive at the node corresponding to that EM_1 instance, instead of taking the left path as we did previously (since the state did not have a witness for that instance), we take the right path, because this time we do have a witness (since we just extended the state with it). Therefore, each time we backtrack, the computation path ends with a leaf that is more to the right in Figure 3. This gives a bound to the number of backtrackings, namely $2^n - 1$.

This is very different from what one could expect by a superficial look at the proof of Lemma 3. Indeed, if the order on the reals were decidable, then this simple and natural proof would be quite efficient, since its complexity would be linear in n. However, its interactive interpretation has exponential complexity. This can be seen in the computation tree too: a single computation corresponds to a path and paths have length n. On the other hand, since we have backtracking, in the worst case we may have to perform every possible computation. Naturally, since the order on the reals *is* undecidable, an actual comparison is impossible.

Moreover, while in the worst case the interactive interpretation needs a time that is exponential in n, in general it is hard to estimate the amount of backtracking that will be actually performed, for two different reasons.

The first one is that the actual order of $r_0, \dots, r_n$ affects heavily the operation of the algorithm. Indeed, assume that r_0 is the least element: the interactive interpretation only performs n dummy comparisons and immediately returns a least element candidate that, in this case, is the actual least element, so no backtracking can ensue later.

The second reason is that the backtracking is controlled by how the least element candidate returned by the interactive interpretation is used. It is possible for the interactive interpretation to return a candidate that is not a least element, but such that its use in an outer proof is does not cause backtracking. In other words, we only need to compute a least element candidate that is good enough instead of the correct one and this can translate to a faster computation, again depending on the situation. This also explains why the interactive interpretation is effective even if a certainly correct least element cannot be found effectively.

In the BHK interpretation, the computational content of a proof is usually much longer than the algorithm it describes. This can be easily seen by comparing the program in Listing 1 with the proof Lemma 3[2]. The reason for this discrepancy is that the proof contains both the algorithm described in Listing 1 and the evidence for its correctness. In general, in the computational content of a proof in the BHK interpretation we can separate the part that computes values and such (the informative computation) from the part that computes the evidence showing that the values are correct (the correctness computation). The correctness computation does not affect the result of the informative computation and can be safely discarded when we are only interested in algorithm extraction.

This is not the case for the computational content in the interactive interpretation. Here the correctness part of the computation affects the backtracking, which affects the state, which in turn affects the informative part of computation and thus the computed values. Therefore, in interactive realizability both parts of the proof interact to produce the final result. This is apparent in the second iteration, when we falsify $r_3 \leq_{\mathbb{R}} r_2$ and we have to retrace the proof of semi-transitivity, which is a non informative proof, in order to find which EM_1 instance we guessed wrongly and to compute the witness that we need to extend the state. This shows that in the interactive interpretation we cannot forget how we proved the correctness of our computations.

4 The Real Plane

In this section we introduce the real plane, points, lines and some relations between them. We use elementary analytic geometry: points are represented by coordinates, lines by equations and proofs are mostly computations with real numbers.

We represent a point as a pair of real numbers, its coordinates. Formally we can say that a point is just a natural number i and that there is a primitive recursive function mapping indexes into pairs of real numbers. As we did for real numbers, in order to improve readability we add some sugar to the notation and use the metavariables P, Q, R, S for arithmetic terms used as indexes of points. When we use the index of a point both as a number and as a point, we write it as i in the first case and as P_i in the second. We write the coordinates of a point P as (x_P, y_P) and of a point P_i as (x_i, y_i). A line passing through two points PQ is written as PQ.

Before proceeding we need to introduce further infrastructure for the real numbers. Any rational number q can be represented as a real number by taking the constant sequence of the degenerate interval $[q,q]$. Let $0_{\mathbb{R}}$ be the representation of the rational zero. We can define addition, subtraction and multiplication on the nested interval sequences by using the corresponding rational operation point-wise on the extremes. It is possible to retain the exponential convergence by taking a suitable sub-sequence. This can be done effectively and follows from the continuity of the operations on the rationals. Thus we can safely assume that we have addition, subtraction and multiplication on the reals.

In order to write the formal statement of Theorem 1, we need a way to determine the position of a point with respect to a line.

First of all consider two points P and Q. We can write the equation that a point R has to satisfy to be on the line going through them:

$$(x_Q - x_P)(y_R - y_P) - (x_R - x_P)(y_Q - y_P) =_{\mathbb{R}} 0_{\mathbb{R}}. \tag{8}$$

If the left-hand side is zero then R is on the same line with P and Q. When left-hand side is not zero, we can use its sign to distinguish which side of PQ R is on. We call these sides left and right. We write

[2]A straightforward formalization of the proof in the Coq proof assistant is ten times longer than Listing 1.

$\mathsf{left}(P,Q,R)$ (resp. $\mathsf{right}(P,Q,R)$) and we say that R is to the *left* (resp. *right*) of the line passing through the points P and Q when

$$\mathsf{left}(P,Q,R) \equiv (x_Q - x_P)(y_R - y_P) - (x_R - x_P)(y_Q - y_P) >_{\mathbb{R}} 0_{\mathbb{R}},$$
$$\mathsf{right}(P,Q,R) \equiv (x_Q - x_P)(y_R - y_P) - (x_R - x_P)(y_Q - y_P) <_{\mathbb{R}} 0_{\mathbb{R}}.$$

Both left and right are positively decidable, since they are defined by means of $<_{\mathbb{R}}$. Moreover, note that R is to the left of PQ if and only if Q is to the right of PR. We say that P is *above* Q if $y_P \geq_{\mathbb{R}} y_Q$ and that R is *below* Q when $y_R \leq_{\mathbb{R}} y_Q$.

5 The Geometric Part of the Proof

Now we are ready to present the rest of the proof of the main statement. We divide the proof in two parts, the first given as a lemma. Since these proofs are more complex, for reason of readability and space we will not be as formal as we have been until now.

From this point onward we assume that no three points are on the same line, formally:

$$\forall P,Q,R.\ \mathsf{left}(P,Q,R) \vee \mathsf{right}(P,Q,R). \tag{9}$$

This is a strong assumption, even more so because we require this to hold constructively: since left and right are Σ^0_1 formulas defined with $\leq_{\mathbb{R}}$, we assume that we have an effective map that given three points yields the precision we need to reach in order to check that R is not on the line PQ. In other words, we are assuming that we have a procedure that effectively decides instances of the left and right predicates. The effective computation we extract uses this procedure as a parameter.

A further consequence is that all points must be distinct: when $x_P =_{\mathbb{R}} x_Q$ and $y_P =_{\mathbb{R}} y_Q$, the left-hand side in (8) is always zero for any R.

In the next lemma the points Q_0, Q_1, Q_2 are three generic points, that is, Q_i is not necessarily the point indexed by the natural number i. Moreover we assume that the index i in Q_i is interpreted up to congruence modulo 3 and thus always falls in $\{0,1,2\}$. For instance, when we write Q_4, we actually mean Q_1. We write the coordinates of Q_i as (x_i, y_i), with the same conventions for the index. We prove that when three points are one to the left of the other with respect to a central one, one of them is necessarily lower than the central point.

Lemma 4 (Three points)**.** *Assume* (9) *and let P, Q_0, Q_1 and Q_2 be four points in the real plane such that Q_{i+1} is to the left (resp. right) of PQ_i for any $i < 3$. Then at least one of Q_0, Q_1, Q_2 is strictly below P. Formally:*

$$\forall P, Q_0, Q_1, Q_2.\ (\forall i < 3.\ \mathsf{left}(P, Q_i, Q_{i+1})) \rightarrow \exists i < 3.\ y_i <_{\mathbb{R}} y_P.$$

We omit the proof for reasons of space. Since it is a constructive proof, its interactive interpretation coincides with its BHK interpretation and thus is not particularly relevant for our analysis.

We can now prove the main statement.

Theorem 1 (Convex Angle)**.** *Assume* (9)*. For any $n \geq 2$, we can select three points P, Q and R from $\{P_0, \ldots, P_n\}$ such that all the remaining points fall in the angle $\widehat{QPR}$, that is, all points are to the left of PQ and to the right of PR.*

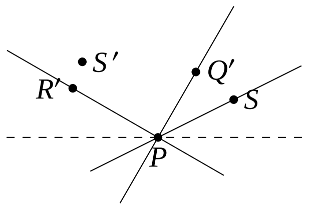
$$\forall n \geq 2.\ \exists i,j,k \leq n.\ \forall l \leq n.\ l \neq i \rightarrow (l \neq j \rightarrow \mathsf{left}(P_i, P_j, P_l)) \wedge (l \neq k \rightarrow \mathsf{right}(P_i, P_k, P_l)).$$

Note that the convexity of the angle $\widehat{QPR}$ is assured, because we require that R is to the left of PQ and Q to the right of PR.

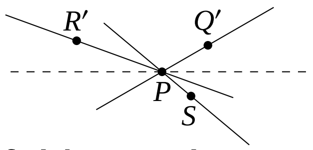
Classical proof. Let P be the point with the least vertical coordinate and choose other two points Q' and R', which are our candidates for Q and R respectively. We want all points except P to be to the left of PQ and to the right of PR. If Q' is to the left of PR', we swap Q' and R'. Thus we know that Q' is to the right of PR' and R' is to the left of PQ'.

Now consider any point S except P, Q' and R'. We have four cases:

- If S is to the left of PQ' and if it is to the right of PR', then we keep Q' and R' as candidates for Q and R.
- If S is to the right of PQ', then we choose S as the new candidate for Q.
  Clearly Q' is to the left of PS. Moreover, any other point S', which we already checked to be to the left of PQ', is to the left of PS too. This is a consequence of (9) and Lemma 4. Indeed, from (9), we know that S' is either to the left or to the right of PS. We already know that S is to the right of PQ' and Q' is to the right of PS'.

  If S' were to the right of PS, then by Lemma 4, one of Q', S or S' would have been strictly lower than P which would be a contradiction, since P is the lowest point. Thus S' is to the left of PS.
- Symmetrically, if S is to the left of PR', then we choose S as the new candidate for R.
- We show that S cannot be to the right of PQ' and to the left of PR'.
  If this were the case, Q' would be to the left of PS and S would be to the left of PR'. Since we know that R' is to the left of PQ', by Lemma 4, one of S, Q' or R' would be strictly lower than P. This is a contradiction, since P is the lowest point by Lemma 3.

We repeat this procedure for all the points except P, Q' and R' and we find the required points Q and R. □

For convenience we have written the proof as an iterative algorithm. The proof is actually by induction on a slightly stronger version of the final statement, that adds the requirement for P to be lower than all the other points.

6 The Interactive Interpretation

Before studying the interactive interpretation of the whole proof of Theorem 1 along with its lemmas, we need to understand their computational significance. Thus we stop for a moment and recall some general considerations on the computational meaning of formulas in the BHK interpretation and, more specifically, in the Curry-Howard correspondence.

As a consequence of the proofs-as-programs and formulas-as-types interpretation, the conclusion of a proof (that is, the statement it proves) can be thought of as the specification of the program representing the proof.

In order to understand how the interactive interpretation works, it is important to distinguish computations that can be carried out effectively from those that cannot. Consider a proof of a statement of the form $\forall x.\ \exists y.\ A$. If we read it as a specification, it calls for a program that describes a function, a subroutine. It takes a natural number as an argument named x and returns a pair containing a natural number y and a program/proof that y satisfies A. All of our theorems begin with universal quantifications and implications, that is, they are specification for programs describing functions with arguments. Thus, in order to have an actual computation we have to provide the program with the required arguments.

We can now explain the interactive interpretation of the whole proof, composed of the two lemmas and the final algorithmic proof. We focus on the interaction between these parts without analyzing each part in detail as we have done for Lemma 3.

We start by considering the statement of Theorem 1. Assume that we are given a natural number n. In the proof we work with the first $n+1$ points of the enumeration.

The proof is an iterative procedure to select P, Q and R satisfying the following *bounding condition*:

$$\forall l \leq n.\ l \neq i \rightarrow (l \neq j \rightarrow \mathsf{left}(P_i, P_j, P_l)) \wedge (l \neq k \rightarrow \mathsf{right}(P_i, P_k, P_l)). \qquad (10)$$

The bounding condition specifies an informative computation, since left and right are defined by means of $<_{\mathbb{R}}$, which is an existential quantification. Thus its proof computes some witnesses, namely the precision of the comparisons we need to check that the bounding condition holds. While we are mainly interested in the choice of the points P, Q and R and not in the information needed to prove the bounding condition itself, the precision of the computation provided by (10) is actually used in interactive interpretation since it can cause backtracking.

We claim that this bounding condition specifies an effective computation. First of all, the outer universal quantification is bounded, thus, in order to compute the condition, we have to compute the body of the quantification $n+1$ times. The same holds for the conjunctions. Thus the effectiveness of the whole condition follows from the effectiveness of the conjuncts. The implications are effective: their only argument, the proof of the antecedent, is arithmetical atomic, hence irrelevant, thus the computations they specify must be constant functions. Therefore, we can effectively compute them by applying them to any single argument. Finally their consequents specify effective computations, thanks to (9), the assumption that no three points are on the same line. Thus, proofs of the bounding condition describe effective computations.

Now we can start following the proof. In the beginning, the lowest point P is selected using Lemma 3 on the vertical coordinate. Consider the statement of Lemma 3:

$$\forall n.\ \exists i \leq n.\ \forall j \leq n.\ r_i \leq_{\mathbb{R}} r_j.$$

As a specification, it calls for a program that, given n, yields the value i and the correctness computation that checks that i is the least element. Since the correctness computation cannot be carried out effectively (it is negatively decidable), the interactive interpretation computes a trivial least element the first time. If later in the proof we happen to partially compute the correctness computation, then we may discover new information and backtrack again to the least element computation. Since Lemma 3 does not necessarily return a least element, but only a least element candidate, P is not the lowest point either, but just a lowest point candidate.

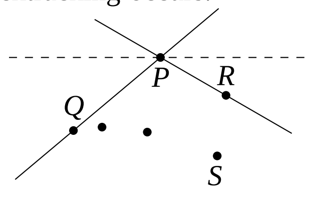
The role of Lemma 4 is to prove that some point is strictly lower than P, thus producing a contradiction. In the classical proof this ensures that undesirable situations never happen. In the interactive interpretation however, since P is not necessarily the lowest point, no contradiction occurs. Instead, what happens is that we actually are in one of the cases we had excluded in the classical proof. At this point, in order to deduce the contradictory statement, we have partially computed the correctness computation returned by Lemma 3 and thus discovered which assumption was incorrectly guessed. We compute the relevant witness and extend the state accordingly. Then we compute a new lowest point candidate and continue again following the proof of Theorem 1 until either we can satisfy its conclusion or we backtrack again.

We use Lemma 4 in two places in the proof of Theorem 1. The first use takes place when, while iterating on the points, we discover that the bounding condition fails for some S and we choose S as the new candidate for Q or R. We use Lemma 4 to show that this choice satisfies the bounding condition for all the previous points we iterated over until now. More precisely we use Lemma 4 to prove that, if the bounding conditions fails for S, then one of Q, R or S is strictly lower than P. As we described previously, this in turn starts the backtracking.

We also use Lemma 4 to claim that the bounding condition cannot fail because S is both to the right of PQ and to the left of PR. This case was excluded completely in the classical proof, since it always leads to contradiction. When it occurs in the interactive interpretation, we backtrack for sure since the bounding condition cannot be satisfied. More precisely, in this case Lemma 4 proves that one of Q, R or S is strictly lower than P. Therefore, in order to get the contradiction, we instantiate the assumptions $y_P \leq_{\mathbb{R}} y_Q$, $y_P \leq_{\mathbb{R}} y_R$ and $y_P \leq_{\mathbb{R}} y_S$ with enough precision to falsify at least one of them.

As a last example, consider a situation where the state is empty and thus P is simply the first point in the enumeration. Assume that the points are arranged as shown in Figure 4. Since the bounding condition is satisfied immediately, we never need to use Lemma 4. Thus backtracking never ensues. This mean that P, while certainly not the lowest point, is a good enough candidate and we do not need another one. This is one of the cases we mentioned where the interactive interpretation produces a fast computation, since the lowest point is only computed once and the proof ends with no backtracking. This shows how the behavior of interactive interpretation of Lemma 3 depends heavily on the final statement of the proof.

Figure 4: A situation where no backtracking occurs.



References

[1] Federico Aschieri & Stefano Berardi (2010): *Interactive Learning-Based Realizability for Heyting Arithmetic with* EM$_1$. *Logical Methods in Computer Science* 6(3), doi:`10.2168/LMCS-6(3:19)2010`.

[2] Stefano Berardi & Ugo de'Liguoro (2008): *A Calculus of Realizers for* EM$_1$ *Arithmetic (Extended Abstract)*. In: *CSL*, pp. 215–229, doi:`10.1007/978-3-540-87531-4_17`.

[3] Giovanni Birolo (2013): *Interactive Realizability, Monads and Witness Extraction*. Ph.D. thesis, University of Turin. Available at `http://arxiv.org/abs/1304.4091`.

[4] A.S. Troelstra & D. van Dalen (1988): *Constructivism in Mathematics*. Studies in Logic and the Foundations of Mathematics, Elsevier Science. Available at `http://books.google.it/books?id=-tc2qp0-2bsC`.