CS6180: Lecture 13

Robert Constable

October 2, 2017

1 Lecture Summary

In this lecture we will focus on the structure of refinement proofs. This is the proof style implemented in the Nuprl proof assistant. It is basically a top down Gentzen style sequent calculus [4]. This style is closely related to the tableaux style proofs as presented by Smullyan in his classic book *First-Order Logic* [10].

We will focus first on expressing the computational content of the constructive proofs, then we will show that using a *virtual evidence* semantics, we can also express *classical first-order logic.* For example, we can express a constructive interpretation of $P \vee \sim P$ which we write as $\{P \vee \sim P\}$.

The supplemental material for the lecture includes several proofs from the Nuprl web page, in particular from this section of the web page: *Logical Investigations, with the Nuprl Proof Assistant* with this url:

http://www.nuprl.org/MathLibrary/LogicalInvestigations/index.html

2 Virtual evidence

The new idea proposed here is to define *virtual constructive evidence* for *classical propositions* using the *refinement type* of computational type theory to specify the classical computational content. The following refinement type, $\{Unit|P\}$, is critical. Its only element is the unique element $\star$ of $Unit$, provided P is known, otherwise it is the empty type. If P is known by constructive evidence p, then the refinement type has $\star$ as its only *explicit* element, and p is hidden *implicit evidence.* We imagine that p has been "squashed" to $\star$, and we call $\{Unit|P\}$

"squashed P."

The new axiom, $\sim\sim P \Rightarrow \{Unit|P\}$, creates virtual evidence for P. That evidence can be used to construct virtual evidence in other refinement types by following the standard rules for them. There is constructive evidence for $\sim\sim P$ but only virtual evidence for P. We abbreviate $\{Unit|P\}$ to $\{P\}$, so the new axiom is simply $\sim\sim P \Rightarrow \{P\}$. This axiom, called here *Classical Introduction*, is similar to the law of *Double Negation Elimination* (DNE) favored by Kolmogorov to axiomatize classical logic as an extension of constructive logic. If there is only constructive evidence for $\sim\sim P$, and none known for P, then there is only virtual evidence for $\{P\}$ that is also hidden and can be unhidden only when proving classical propositions. *Virtual constructive evidence carries more information than is provided by standard classical semantics.* This new semantics integrates classical and constructive logic and is possible because constructive type theory expresses more kinds of evidence than classical set theory.

In a sense, mathematics has been constructive since Euclid. What we call "classical mathematics" is a 19th century trend, say from 1880 onwards. There are topics in mathematics that have always been constructive. Given a choice, mathematicians generally favor constructive proofs. Edwards [3] takes a modern look at this constructive thread as does Henrici [6]. Intuitionistic mathematics is an approach to constructive mathematics developed by L.E.J. Brouwer starting with his doctoral thesis in 1907 and developed over his lifetime in a series of deeply novel and fundamental mathematical results [7, 12, 11].[1] Kolmogorov appears to have been the first logician to provide an axiomatic account of Brouwer's ideas for intuitionistic mathematics. Moreover, Kolmogorov suggested a unification of intuitionistic and classical logics based on his *double negation embedding* of classical logics into intuitionistic ones. Glivenko [5] proved the two versions are equally expressive for propositional logic. Kuroda [8] extended the result to first-order logic. They used proof theoretic techniques. Murthy [9] implemented this embedding in Nuprl and used it to produce an automated formal transformation of a classical proof of Higman's Lemma into a constructive proof. Kolmogorov's approach would also allow mixing of classical and constructive logics in which the classical theorems used double negations to translate the logical operators into constructive versions. We show in this lecture how to accomplish such a translation by purely semantics means.

The new idea presented here involves two steps. The first is to hide the evidence for a proposition P using the *refinement type*. The classical meaning of P is based on the constructive refinement type $\{Unit|P\}$ where $Unit$ is the type with precisely one element, $\star$. To know $\{Unit|P\}$ constructively, we need evidence p for P, but it is not included with

[1] Brouwer called his work *modern mathematics*, see [11].

the evidence type. We can say that the evidence is forgotten or "squashed down to $\star$". It becomes *virtual evidence* or *imaginary evidence* when we talk about $\{Unit|P\}$. Only $\star$ belongs to this type when it is non empty. The evidence p is *virtual*.

We also refer to this refinement type as *squashed* P. It is clear that there is evidence for $P \Rightarrow \{Unit|P\}$, so we know this implication constructively. However, we do not know $\{Unit|P\} \Rightarrow P$ in general because the implication only has access to $\star$ and not the evidence for P. Even if we had known evidence for P and hidden it, to prove this proposition, we would need to reconstruct that evidence. Clearly we do not have explicit evidence for $\{Unit|P \vee \sim P\} \Rightarrow (P \vee \sim P)$ for all propositions P.

We will see below that the computational evidence for $P \Rightarrow \{Unit|P\}$ is a function that takes evidence for P and produces $\star$ in $\{Unit|P\}$. The function discards the evidence for P when it produces $\star$. To know that this function has this type requires that we have evidence for P *even though we don't save it with the type*. We don't carry evidence for P with the type as a matter of efficiency since it might require a substantial amount of space that is not needed in computations. But this reason opens the possibility to define virtual evidence as we do next.

Another way to explain the meaning of the refinement type $\{Unit|P\}$ is in terms of a type of the form $\{x : A|B(x)\}$, which consists of those elements a of type A such that evidence b for $B(a)$ is known. Thus an element belongs to $\{Unit|P\}$ if and only if it belongs to $Unit$ and evidence p for P is known. If constructive evidence p for P is known, then $\star$ belongs to $\{Unit|P\}$. We interpret this type as "squashing" the evidence p which might be complex, down to a single point, $\star$. The constructive details are hidden, but this can only be done once evidence is known in the first place or built up from assumptions that have this classical character. This "hiding operation" was introduced in "Constructive Mathematics as a Programming Logic I: Some Principles of Theory" [2] and used extensively ever since. This step does not take us beyond constructive logic, but it introduces the idea that there is a type that is designed to *hide evidence*. We look next at properties of this type.

If we know $\sim P$ constructively, then we know that $\{Unit|P\}$ is definitely empty – "there is nothing to hide". When we know constructively that there is no evidence for $\sim P$, then we know that $\{Unit| \sim\sim P\}$ *cannot be empty*. We also know that it can have only one possible unhidden element, namely $\star$. Inspired by Kolmogorov's observation that the classical axiom for Double Negation Elimination, $\sim\sim P \Rightarrow P$, gives us full classical logic, we consider whether we can make constructive sense of the idea that $\sim\sim P \Rightarrow \{Unit|P\}$. *If we don't actually have evidence for P, we can't constructively inhabit the type* $\{Unit|P\}$. To do so requires evidence for P, even though once we have that evidence, we hide it.[2] We call this hidden evidence *virtual* because we can use $\{Unit|P\}$ *as if we had evidence for* P when

[2]In proof assistants such as Nuprl, the evidence can be found in the proof object, but it is not part of the refinement type.

we are trying to prove other refinement types. That is how the refinement rules work in constructive type theory.

For classical reasoning, constructive evidence for P is not required. Classical reasoners consider the possibility that there might be evidence available in ways yet unknown. Perhaps an *Omniscient Assistant* (OA) knows convincing evidence. Once a deeper understanding is achieved using virtual evidence, it might be possible to find constructive evidence. This has often happened in the past.

In light of the above observations, the **second step** is adding the following new axiom of **Classical Introduction (CI)**:

$$\sim\sim P \Rightarrow \{Unit|P\}.$$

The required constructive evidence to know that $\star$ belongs to this type is knowing specific evidence p for P.[3] We might not have such evidence since we only assume we know $\sim\sim P$. In particular, we do not have evidence when we consider $\sim\sim (P \vee \sim P) \Rightarrow \{Unit|(P \vee \sim P)\}$ even though we have constructive evidence for $\sim\sim (P \vee \sim P)$.

Our proposal is to forego including evidence for P when realizing the new axiom $\sim\sim P \Rightarrow \{Unit|P\}$. In doing this, we give up evidence semantics, but not entirely. We can justify this rule under the interpretation that *propositions justified by virtual evidence cannot be disproved*. We can prove in a constructive metatheory such as the one formalized by Anand and Rahli [1] that the augmented theory cannot prove $False$. For every rule, we know that if the hypotheses are not false, then the conclusion is not false. *This approach agrees with the classical notion that evidence is not required for axioms.* It agrees with the constructive notion that axioms should be realizable in that the constant function whose value is $\star$ *is an adequate realizer for the only evidence that must be visible.* The missing evidence is not carried along with the type, and when the type occurs in assumptions, only the virtual evidence is available. That virtual constructive evidence is all that is necessary for classical reasoning. The only explicit evidence needed for computation is $\star$. Since constructively $\{Unit|P\}$ can not be empty, and since it can only have $\star$ as a member, this must be the member. *So this axiom is constructive in the sense that we have not lost the required computational meaning of the type.*

This simple rule narrows the difference between constructive and classical reasoning to a very small point. Since our explanation is given in a constructive meta-theory, we preserve a constructive meaning for the rules. However, for some rules we use virtual evidence knowing that it *cannot lead to constructively false conclusions.*

To summarize, if there is constructive evidence for $\sim\sim P$, say nnp, then we can find a

[3]We might also want to name this Constructive DNE.

function to give us the *accessible evidence* for $\{Unit|P\}$, namely $\star$. That is all we are allowed to infer constructively from the type. The constant function whose value is $\star$ is the realizer for this axiom. There is no loss of classical content when we make this positive assertion that we have trivial evidence $\star$ for P. Thus we can provide *computational evidence* for the implication $\sim\sim P \Rightarrow \{Unit|P\}$, that is, evidence for $\sim\sim P \Rightarrow \{P\}$. With our new axiom we can prove:

$$\{Unit|P\} \Leftrightarrow \sim\sim P.$$

The Classical Introduction axiom reveals the essential difference in evidence requirements between classical and constructive logic, and it allows us to conduct classical reasoning with refinement types in a way that can be explained constructively without assuming that we have an oracle that can provide enough evidence to justify the proposition constructively, yet *can provide computational evidence that is sufficient for classical reasoning*. By mixing the classical and constructive modes, we allow a natural style of classical reasoning that supports extracting programs from proofs that allow some classical steps.

In this context, one way of understanding $\{P\}$ is that by proving this squashed proposition, one knows that it is *classically consistent*. In some cases there is constructive evidence in hand for P. In other cases the constructive status of P might be entirely unknown, yet to be discovered or perhaps forgotten or only a partly scribbled note in the margin or perhaps evidence known only to an Omniscient Assistant that might be revealed one day or even discovered by a proof assistant that knows $\{P\}$ and is set the goal to keep looking for a proof of P. In any case, establishing $\{P\}$, raises the possibility of the stronger result P. Many constructive results have been discovered by exploiting this possibility.

What we also know is that the classical reasoning mode is semantically consistent with constructive reasoning. This will not be proved here. It can be established in the semantic model of CTT and could thus be formally proved in Coq.

3 Web resources

We provide on the Nuprl web page a number of examples of proofs using virtual evidence under the heading *Logical Investigations with the Nuprl Proof Assistant* written by Anne Trostle and me, see http://www.nuprl.org/MathLibrary/LogicalInvestigations/index.html.

References

[1] Abhishek Anand and Vincent Rahli. Towards a formally verified proof assistant. In *ITP*, pages 27–44, 2014.

[2] Robert L. Constable. Constructive mathematics as a programming logic I: Some principles of theory. In *Annals of Mathematics*, volume 24, pages 21–37. Elsevier Science Publishers, B.V. (North-Holland), 1985. Reprinted from *Topics in the Theory of Computation*, Selected Papers of the International Conference on Foundations of Computation Theory, FCT '83.

[3] Harold M. Edwards. Introduction to my book "essays in constructive mathematics". In Thierry Coquand, Henri Lombardi, and Marie-Françoise Roy, editors, *Mathematics, Algorithms, Proofs*, number 05021 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2006. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.

[4] Gerhard Gentzen. Investigations into logical deduction (1934). In M. Szalo, editor, *The Collected Paers of Gerhard Gentzen*. North-Holland, Amsterdam, 1969.

[5] V. Glivenko. Sur quelques points de la logique de m. brouwer. *Bulletins de la classe des sciences*, 15(2):183–188, 1929.

[6] Peter Henrici. *Applied and Computational Complex Analysis*, volume 1–3. John Wiley and Sons, New York, 1988.

[7] A. Heyting, editor. *L. E. J. Brouwer Collected Works*, volume 1. North-Holland, Amsterdam, 1975.

[8] S. Kuroda. Intuitionistische Untersuchungen der formalistchen Logik. *Nagoya Mathematical Journal*, 2:35–47, 1951.

[9] Chetan Murthy. An evaluation semantics for classical proofs. In *Proceedings of the 6^{th} Symposium on Logic in Computer Science*, pages 96–109, Vrije University, Amsterdam, The Netherlands, July 1991. IEEE Computer Society Press.

[10] Raymond M. Smullyan. *First-Order Logic*. Dover Publications, New York, 1995.

[11] Mark van Atten. *On Brouwer*. Wadsworth Philosophers Series. Thompson/Wadsworth, Toronto, Canada, 2004.

[12] Walter P. van Stigt. *Brouwer's Intuitionism*. North-Holland, Amsterdam, 1990.