Chapter 8
Proof Theory

Jeremy Avigad

Abstract Proof theory began in the 1920s as a part of Hilbert's program, which aimed to secure the foundations of mathematics by modeling infinitary mathematics with formal axiomatic systems and proving those systems consistent using restricted, finitary means. The program thus viewed mathematics as a system of reasoning with precise linguistic norms, governed by rules that can be described and studied in concrete terms. Today such a viewpoint has applications in mathematics, computer science, and the philosophy of mathematics.

8.1 Introduction

At the turn of the nineteenth century, mathematics exhibited a style of argumentation that was more explicitly computational than is common today. Over the course of the century, the introduction of abstract algebraic methods helped unify developments in analysis, number theory, geometry, and the theory of equations, and work by mathematicians like Richard Dedekind, Georg Cantor, and David Hilbert towards the end of the century introduced set-theoretic language and infinitary methods that served to downplay or suppress computational content. This shift in emphasis away from calculation gave rise to concerns as to whether such methods were meaningful and appropriate in mathematics. The discovery of paradoxes stemming from overly naive use of set-theoretic language and methods led to even more pressing concerns as to whether the modern methods were even consistent. This led to heated debates in the early twentieth century and what is sometimes called the "crisis of foundations."

In lectures presented in 1922, Hilbert launched his *Beweistheorie*, or Proof Theory, which aimed to justify the use of modern methods and settle the problem of foundations once and for all. This, Hilbert argued, could be achieved as follows:

J. Avigad (✉)
Department of Philosophy and Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, PA 15213, USA
e-mail: avigad@cmu.edu

S. O. Hansson, V. F. Hendricks (eds.), *Introduction to Formal Philosophy*, Springer Undergraduate Texts in Philosophy, https://doi.org/10.1007/978-3-319-77434-3_8

- First, represent portions of the abstract, infinitary mathematical reasoning in question using formal axiomatic systems, which prescribe a fixed formal language and precise rules of inference.
- Then view proofs in these systems as finite, combinatorial objects, and prove the consistency of such systems—i.e. the fact that there is no way to derive a contradiction—using unobjectionable, concrete arguments.

In doing so, said Hilbert,

> ...we move to a higher level of contemplation, from which the axioms, formulae, and proofs of the mathematical theory are themselves the objects of a contentional investigation. But for this purpose the usual contentual ideas of the mathematical theory must be replaced by formulae and rules, and imitated by formalisms. In other words, we need to have a strict formalization of the entire mathematical theory.... In this way the contentual thoughts (which of course we can never wholly do without or eliminate) are removed elsewhere—to a higher plane, as it were; and at the same time it becomes possible to draw a sharp and systematic distinction in mathematics between the formulae and formal proofs on the one hand, and the contentual ideas on the other. [17]

Gödel's second incompleteness theorem shows that any "unobjectionable" portion of mathematics is insufficient to establish its own consistency, let alone the consistency of any theory properly extending it. Although this dealt a blow to Hilbert's program as it was originally formulated, the more general project of studying mathematical reasoning in syntactic terms, especially with respect to questions of algorithmic or otherwise concrete content, has been fruitful. Moreover, the general strategy of separating syntactic and semantic concerns and of maintaining a syntactic viewpoint where possible has become a powerful tool in formal epistemology. (See [34, 42] for more on Hilbert's program.)

Today, Proof Theory can be viewed as the general study of formal deductive systems. Given that formal systems can be used to model a wide range of types of inference—modal, temporal, probabilistic, inductive, defeasible, deontic, and so on—work in the field is varied and diverse. Here I will focus specifically on the proof theory of *mathematical* reasoning, but even with this restriction, the field is dauntingly broad: the 1998 *Handbook of Proof Theory* [9] runs more than 800 pages, with a name index that is almost as long as this article. As a result, I can only attempt to convey a feel for the subject's goals and methods of analysis, and help ease the reader into the broader literature. References are generally to surveys and textbooks, and results are given without attribution.

In the next section, I describe natural deduction and a sequent calculus for first-order logic, and state the cut-elimination theorem and some of its consequences. This is one of the field's most fundamental results, and provides a concrete example of proof-theoretic method. After that, I survey various aspects of proof-theoretic analysis, and, finally, in the last section, I discuss some applications.

8.2 Natural Deduction and Sequent Calculi

I will assume the reader is familiar with the language of first-order logic. Contemporary logic textbooks often present formal calculi for first-order logic with a long list of axioms and a few simple rules, but these are generally not very convenient for modeling deductive arguments or studying their properties. A system which fares better on both counts in given by Gerhard Gentzen's system of *natural deduction*, a variant of which we will now consider.

Natural deduction is based on two fundamental observations. The first is that it is natural to describe the meaning, or appropriate use, of a logical connective by giving the conditions under which one can *introduce* it, that is, derive a statement in which that connective occurs, and the methods by which one can *eliminate* it, that is, draw conclusions from statements in which it occurs. For example, one can establish a conjunction $\varphi \wedge \psi$ by establishing both φ and ψ, and, conversely, if one assumes or has previously established $\varphi \wedge \psi$, one can conclude either φ or ψ, at will.

The second observation is that it is natural to model logical arguments as taking place under the context of a list of hypotheses, either implicit or explicitly stated. If Γ is a finite set of hypotheses and φ is a first-order formula, the *sequent* $\Gamma \Rightarrow \varphi$ is intended to denote that φ follows from Γ. For the most part, these hypotheses stay fixed over the course of an argument, but under certain circumstances they can be removed, or *canceled*. For example, one typically proves an implication $\varphi \to \psi$ by temporarily assuming that φ holds and arguing that ψ follows. The introduction rule for implication thus reflects the fact that deriving ψ from a set of hypotheses Γ together with φ is the same as deriving $\varphi \to \psi$ from Γ.

Writing Γ, φ as an abbreviation for $\Gamma \cup \{\varphi\}$, the rules for natural deduction are shown in Fig. 8.1. The quantifier rules are subject to the usual restrictions. For example, in the introduction rule for the universal quantifier, the variable x cannot be free in any hypothesis. For intuitionistic logic, one also needs the rule *ex falso sequitur quodlibet*, which allows one to conclude $\Gamma \Rightarrow \varphi$ from $\Gamma \Rightarrow \bot$, where $\bot$ represents falsity. One can then define negation, $\neg\varphi$, as $\varphi \to \bot$. For classical logic, one adds *reductio ad absurdum*, or proof by contradiction, which allows one to conclude $\Gamma \Rightarrow \varphi$ from $\Gamma, \neg\varphi \Rightarrow \bot$.

For many purposes, however, *sequent calculi* provide a more convenient representation of logical derivations. Here, sequents are of the form $\Gamma \Rightarrow \Delta$, where Γ and Δ are finite sets of formulas, with the intended meaning that the conjunction of the hypotheses in Γ implies the *disjunction* of the assertions in Δ. The rules are as shown in Fig. 8.2. The last rule is called the *cut rule*: it is the only rule containing a formula in the hypothesis that may be entirely unrelated to the formulas in the conclusion. Proofs that do not use the cut rule are said to be *cut free*. One obtains a proof system for intuitionistic logic by restricting Δ to contain at most one formula, and adding an axiomatic version of *ex falso sequitur quodlibet*: $\Gamma, \bot \Rightarrow \varphi$. The cut-elimination theorem is as follows:

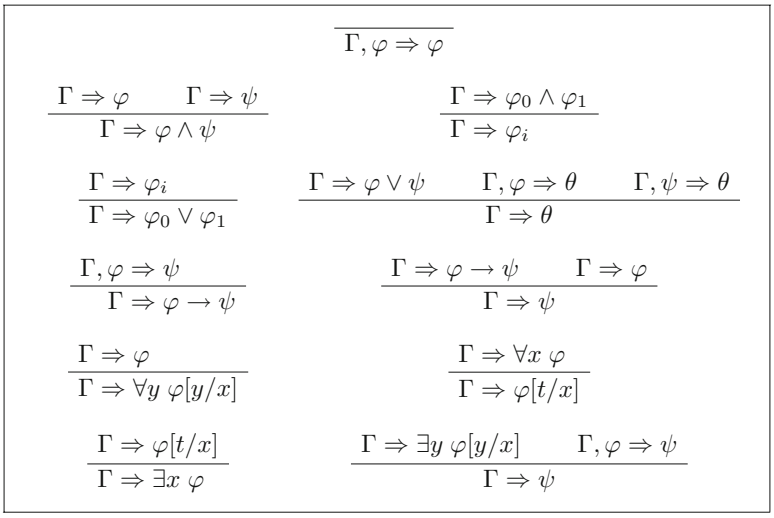

Fig. 8.1 Natural deduction. Derivability of a sequent $\Gamma \Rightarrow \varphi$ means that φ is a consequence of the set of hypotheses Γ, and Γ, φ denotes $\Gamma \cup \{\varphi\}$

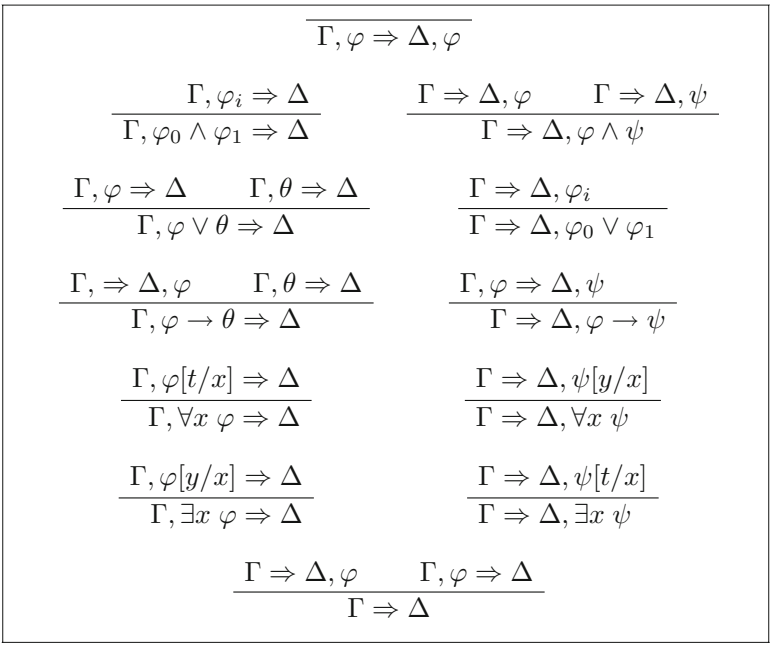

Fig. 8.2 The sequent calculus

Theorem 2.1 *If* $\Gamma \Rightarrow \Delta$ *is derivable in the sequent calculus with cut, then it is derivable without cut.*

Gentzen's proof gives an explicit algorithm for removing cuts from a proof. The algorithm, unfortunately, can yield an iterated exponential increase in the size of proofs, and one can show that there are cases in which such an increase cannot be avoided. The advantage of having a cut-free proof is that the formulas in each sequent are built up directly from the formulas in the sequents above it, making it easy to extract useful information. For example, the following are two consequences of the cut-elimination theorem, easily proved by induction on cut-free proofs.

The first is known as *Herbrand's theorem*. Recall that a formula of first-order logic is said to be *existential* if it consists of a block of existential quantifiers followed by a quantifier-free formula. Similarly, a formula is said to be *universal* if it consists of a block of universal quantifiers followed by a quantifier-free formula. Herbrand's theorem says that if it is possible to prove an existential statement from some universal hypotheses, then in fact there is an explicit sequence of terms in the language that witness the truth of the conclusion.

Theorem 2.2 *Suppose* $\exists \vec{x}\ \varphi(\vec{x})$ *is derivable in classical first-order logic from a set of hypotheses* Γ*, where* φ *is quantifier-free and the sentences in* Γ *are universal sentences. Then there are sequences of terms* $\vec{t}_1, \vec{t}_2, \ldots, \vec{t}_k$ *such that the disjunction* $\varphi(\vec{t}_1) \vee \varphi(\vec{t}_2) \vee \ldots \vee \varphi(\vec{t}_k)$ *has a quantifier-free proof from instances of the sentences in* Γ.

For intuitionistic logic, one has a stronger property, known as the *explicit definability property*.

Theorem 2.3 *Suppose* $\exists \vec{x}\ \varphi(\vec{x})$ *is derivable in intuitionistic first-order logic from a set of hypotheses* Γ *in which neither* $\vee$ *nor* $\exists$ *occurs in a strictly positive part. Then there are terms* $\vec{t}$ *such that* $\varphi(\vec{t})$ *is also derivable from* Γ.

Theorem 2.2 provides a sense in which explicit information can be extracted from certain classical proofs, and Theorem 2.3 provides a sense in which intuitionistic logic is constructive. We have thus already encountered some of the central themes of proof-theoretic analysis:

- Important fragments of mathematical reasoning can be captured by formal systems.
- One can study the properties of these formal systems, for example, describing transformations of formulas and proofs, translations between formulas and proofs in different systems, and canonical normal forms for formulas and proofs.
- The methods provide information about the logic that is independent of the choice of formal system that is used to represent it.

For more on the cut-elimination theorems, see [11, 23, 31, 36, 38].

8.3 Methods and Goals

8.3.1 Classical Foundations

Recall that Hilbert's program, broadly construed, involves representing mathematical reasoning in formal systems and then studying those formal systems as mathematical objects themselves. The first step, then, requires finding the right formal systems. It is common today to view mathematical reasoning as consisting of a properly mathematical part that is used in conjunction with more general forms of logical reasoning, though there are still debates as to where to draw the line between the two. In any case, the following list portrays some natural systems of reasoning in increasing logical/mathematical strength:

1. pure first-order logic
2. primitive recursive arithmetic (denoted *PRA*)
3. first-order arithmetic (*PA*)
4. second-order arithmetic (PA^2)
5. higher-order arithmetic (PA^ω)
6. Zermelo-Fraenkel set theory (*ZF*)

Primitive recursive arithmetic was designed by Hilbert and Bernays to be a patently finitary system of reasoning. The system allows one to define functions on the natural numbers using a simple schema of primitive recursion, and prove facts about them using a principle of induction:

$$\varphi(0) \wedge \forall x\ (\varphi(x) \rightarrow \varphi(x+1)) \rightarrow \forall x\ \varphi(x).$$

In words, if φ holds of 0 and, whenever it holds of some number, x, it holds of $x+1$, then φ holds of every number. Here φ is assumed to be a quantifier-free formula. In fact, one can replace this axiom with a suitable induction *rule*, whereby primitive recursive arithmetic can be formulated without quantifiers at all. Surprisingly, via coding of finitary objects as natural numbers, this system is expressive and strong enough to develop most portions of mathematics that involve only finite objects and structures [3]. Peano arithmetic can be viewed as the extension of *PRA* with induction for all first-order formulas.

There is no effective axiomatization of second- or higher-order logic that is complete for the standard semantics, where second-order quantifiers are assumed to range over all subsets of the universe of individuals. As a result, one has to distinguish axiomatic second- and higher-order logic from the corresponding semantic characterization. Axiomatically, one typically augments first-order logic with comprehension rules that assert that every formula defines a set (or predicate):

$$\exists X\ \forall y\ (X(y) \leftrightarrow \varphi)$$

Here φ is a formula in which X does not occur, although φ is allowed to have other free variables in addition to y. One can augment these with suitable choice principles as well. Second-order arithmetic can be viewed as the extension of Peano arithmetic with second-order logic and second-order principles of induction, but one can, alternatively, interpret second-order arithmetic in second-order logic together with an axiom asserting the existence of an infinite domain. Similar considerations hold for higher-order logic as well.

Axioms for set theory can be found in any introductory set theory textbook, such as [22]. Of course, these axioms can be extended with stronger hypotheses, such as large cardinal axioms. For information on primitive recursive arithmetic, see [14, 38]; for first-order arithmetic, see [11, 15, 18]; for second-order arithmetic, see [35]; for higher-order arithmetic, see [36].

8.3.2 Constructive Foundations

Given the history of Hilbert's program, it should not be surprising that proof theorists have also had a strong interest in formal representations of constructive and intuitionistic reasoning. From an intuitionistic standpoint, the use of the excluded middle, $\varphi \vee \neg\varphi$, is not acceptable, since, generally speaking, one may not know (or have an algorithm to determine) which disjunct holds. For example, in classical first-order arithmetic, one is allowed to assert $\varphi \vee \neg\varphi$ for a formula φ that expresses the twin primes conjecture, even though we do not know which is the case. If one restricts the underlying logic to intuitionistic logic, however, one obtains *Heyting arithmetic*, which is constructively valid.

Stronger systems tend to be based on what has come to be known as the Curry-Howard-Tait *propositions as types* correspondence. The idea is that, from a constructive perspective, any proposition can be viewed as specifying a type of data, namely, the type of construction that warrants the claim that the proposition is true. A proof of the proposition is thus a construction of the corresponding type. For example, a proof of $\varphi \wedge \psi$ is a proof of φ paired with a proof of ψ, and so $\varphi \wedge \psi$ corresponds to the type of data consisting of pairs of type φ and ψ. Similarly, a proof of $\varphi \rightarrow \psi$ should be a procedure transforming a proof of φ into a proof of ψ, so $\varphi \rightarrow \psi$ corresponds to a type of functions. This gives rises to systems of *constructive type theory,* of which the most important examples are *Martin-Löf type theory* and an impredicative variant designed by Coquand and Huet, the *calculus of constructions*. Thus, our representative sample of constructive proof systems, in increasing strength, runs as follows:

1. intuitionistic first-order logic
2. primitive recursive arithmetic (*PRA*)
3. Heyting arithmetic (*HA*)
4. Martin-Löf type theory (*ML*)
5. the calculus of inductive constructions (*CIC*)

Good references for intuitionistic systems in general are [7, 39]. For more information on type theory, see [30]; for the calculus of inductive constructions in particular, see [8].

8.3.3 Reverse Mathematics

In the 1970s, Harvey Friedman observed that by restricting the induction and comprehension principles in full axiomatic second-order arithmetic, one obtains theories that are strong enough, on the one hand, to represent significant parts of ordinary mathematics, but weak enough, on the other hand, to be amenable to proof-theoretic analysis. He then suggested calibrating various mathematical theorems in terms of their axiomatic strength. Whereas in ordinary (meta)mathematics, one proves theorems from axioms, Friedman noticed that it is often the case that a mathematical theorem can be used in the other direction, namely, to prove an underlying set-existence principle, over a weak base theory. That is, it is often the case that a theorem of mathematics is formally *equivalent* to a set comprehension principle that is used to prove it.

In that years that followed, Friedman, Stephen Simpson, and many others worked to calibrate the axiomatic assumptions used in a wide range of subjects. They isolated five key theories along the way:

1. RCA_0: a weak base theory, conservative over primitive recursive arithmetic, with a *recursive comprehension axiom*, that is, a principle of comprehension for recursive (computable) sets.
2. WKL_0: adds *weak König's lemma,* a compactness principle, to RCA_0.
3. ACA_0: adds the *arithmetic comprehension axiom*, that is, comprehension for arithmetically definable sets.
4. ATR_0: adds a principle of *arithmetical transfinite recursion,* which allows one to iterate arithmetic comprehension along countable well-orderings.
5. Π^1_1–CA_0: adds the Π^1_1 *comprehension axiom*, that is, comprehension for Π^1_1 sets.

Simpson [35] provides the best introduction to these theories and the reverse mathematics program.

8.3.4 Comparative Analysis and Reduction

We have now seen a sampling of the many formal systems that have been designed to formalize various aspects of mathematics. Proof theorists have also invested a good deal of energy in understanding the relationships between the systems. Often, results take the form of *conservation theorems* which fit the following pattern, where T_1 and T_2 are theories and Γ is a class of sentences:

> Suppose T_1 proves a sentence φ, where φ is in Γ. Then T_2 proves it as well (or perhaps a certain translation, φ').

Such a result, when proved in a suitably restricted base theory, provides a foundational reduction of the theory T_1 to T_2, justifying the principles of T_1 relative to T_2. For example, such theorems can be used to reduce:

- an infinitary theory to a finitary one
- a nonconstructive theory to a constructive one
- an impredicative theory to a predicative one
- a nonstandard theory (in the sense of nonstandard analysis) to a standard one

For example:

1. Versions of primitive recursive arithmetic based on classical, intuitionistic, or quantifier-free logic all prove the same Π_2 theorems (in an appropriate sense) [38].
2. The Gödel-Gentzen double-negation interpretation and variations, like the Friedman-Dragalin A-translation, interpret a number of classical systems in intuitionistic ones, such as *PA* in *HA* [1, 10, 12, 38, 39].
3. There are various translations between theories in the language of (first-, second-, or higher-order) arithmetic and subsystems of set theory [27, 35].
4. Both $I\Sigma_1$, the subsystem of Peano arithmetic in which induction is restricted to Σ_1 formulas, and WKL_0, the subsystem of second-order arithmetic based on Weak König's Lemma, are conservative over primitve recursive arithmetic for the class of Π_2 sentences [2, 11, 15, 20, 33, 35, 38].
5. Cut elimination or an easy model-theoretic argument shows that a restricted second-order version, ACA_0, of Peano arithmetic is a conservative extension of Peano arithmetic itself. Similarly, Gödel-Bernays-von Neumann set theory *GBN*, which has both sets and classes, is a conservative extension of Zermelo-Fraenkel set theory. See, for example, [28, 35]. In general, proofs in ACA_0 may suffer an iterated exponential increase in length when translated to *PA*, and similarly for *GBN* and *ZF*, or $I\Sigma_1$ and *PRA*.
6. Theories of nonstandard arithmetic and analysis can be calibrated in terms of the strength of standard theories [19].
7. The axiom of choice and the continuum hypothesis are conservative extensions of set theory for Σ^2_1 sentences in the analytic hierarchy [22].

Such results draw on a variety of methods. Some can be obtained by direct translation of one theory into another. Many are proved using cut-elimination or normalization [11, 33]. The double-negation translation is a remarkably effective tool when it comes to reducing classical theories to constructive ones, and can often be supplemented by realizability, functional interpretation, or other arguments [1, 20, 37]. Model-theoretic methods can often be used, though they do not provide specific algorithms to carry out the translation [15, 18]. Even forcing methods, originally developed as a set-theoretic technique, can be fruitfully be applied in proof-theoretic settings [4, 22].

8.3.5 Characterizing Logical Strength

The results described in the previous section serve to characterize the strength of one axiomatic theory in terms of another. Showing that a theory T_2 is conservative over T_1 shows that, in particular, T_2 is consistent, if T_1 is. This provides a comparison of the *consistency strength* of the two theories.

But there are other ways of characterizing the strength of a theory. For example, the notion of an *ordinal* generalizes the notion of a counting number. Starting with the natural numbers, we can add an infinite "number," ω, and keep going:

$$0, 1, 2, 3, \ldots, \omega, \omega + 1, \omega + 2, \omega + 3, \ldots$$

We can then proceed to add even more exotic numbers, like $\omega \cdot 2$, ω^2, and ω^ω. The ordering on these particular expressions is computable, in the sense that one can write a computer program to compare any two them. What makes them ordinals is that they satisfy a principle of *transfinite induction,* which generalizes the principle of induction on the natural numbers. Ordinal analysis gauges the strength of a theory in terms of such computable ordinals: the stronger a theory is, the more powerful the principles of transfinite induction it can prove. See, for example, [26, 27, 36].

Alternatively, one can focus on a theory's *computational strength*. Suppose a theory T proves a statement of the form $\forall x\ \exists y\ R(x, y)$, where x and y range over the natural numbers, and R is a computationally decidable predicate. This tells us that a computer program that, on input x, systematically searches for a y satisfying $R(x, y)$ always succeeds in finding one. Now suppose f is a function that, on input x, returns a value that is easily computed from the least y satisfying $R(x, y)$. For example, $R(x, y)$ may assert that y codes a halting computation of a particular Turing machine on input x, and f may return the result of such a computation. Then f is a computable function, and we can say that the theory, T, proves that f is totally defined on the natural numbers. A simple diagonalization shows that no effectively axiomatized theory can prove the totality of every computable function in this way, so this suggests using the set of computable functions that the theory can prove to be total as a measure of its strength.

A number of theories have been analyzed in these terms. For example, by the results in the last section, the provably total computable functions of PRA, $I\Sigma_1$, RCA_0, and WKL_0 are all the primitive recursive functions. In contrast, one can characterize the provably total computable functions of PA and HA in terms of higher-type primitive recursion [5, 37], or using principles of primitive recursion along an ordinal known as ε_0 [27, 36]. Weaker theories of arithmetic can be used to characterize complexity classes like the polynomial time computable functions [11].

8.4 Applications

In this final section, I will describe some of the ways that proof theory interacts with other disciplines. As emphasized in the introduction, I am only considering applications of the traditional, metamathematical branch of proof theory. Formal deductive methods, more broadly, have applications across philosophy and the sciences, and the use of proof-theoretic methods in the study of these formal deductive systems is far too diverse to survey here.

8.4.1 Proof Mining

One way in which traditional proof-theoretic methods have been applied is in the process of extracting useful information from ordinary mathematical proofs. The reductive results of the twentieth century showed, in principle, that many classical proofs can be interpreted in constructive terms. In practice, these ideas have been adapted and extended to the analysis of ordinary mathematical proofs. Georg Kreisel described the process of extracting such information as "unwinding proofs," and Ulrich Kohlenbach has more recently adopted the name "proof mining" [20].

Substantial work is needed to turn this vague idea into something practicable. Ordinary mathematical proofs are not presented in formal systems, so there are choices to be made in the formal modeling. In addition, the general metamathematical tools have to be tailored and adjusted to yield the information that is sought in particular domains. Thus the work requires a deep understanding of both the proof-theoretic methods and the domain of mathematics in question. The field has already had a number of successes in fields like functional analysis and ergodic theory; see, for example, [20].

8.4.2 Combinatorial Independences

Yet another domain where a syntactic, foundational perspective is important is in the search for natural combinatorial independences, that is, natural finitary combinatorial principles that are independent of conventional mathematical methods. The Paris-Harrington statement [24] is an early example of such a principle. Since then, Harvey Friedman, in particular, has long sought to find exotic combinatorial behavior in familiar mathematical settings. Such work gives us glimpes into what goes on just beyond ordinary patterns of mathematical reasoning, and yields interesting mathematics as well. See the extensive introduction to [13] for an overview of results in this area.

8.4.3 Constructive Mathematics and Type Theory

As noted above, proof theory is often linked with constructive mathematics, for historical reasons. After all, Hilbert's program was initially an attempt to justify mathematics with respect to methods that are finitary, which is to say, syntactic, algorithmic, and impeccably constructive. Contemporary work in constructive mathematics and type theory draws on the following facts:

- Logical constructions can often be interpreted as programming principles.
- Conversely, programming principles can be interpreted as logical constructions.
- One can thereby design (constructive) proof systems that combine aspects of both programming and proving.

The references under Sect. 8.3.2 above provide logical perspectives on constructive type theory. For a computational perspective, see [25].

8.4.4 Automated Reasoning and Formal Verification

Another domain where proof-theoretic methods are of central importance is in the field of automated reasoning and formal verification. In computer science, researchers use formal methods to help verify that hardware and software are bug-free and conform to their specifications. Moreover, recent developments have shown that computational formal methods can be used to help verify the correctness of complex mathematical proofs as well. Both efforts have led to interactive approaches, whereby a user works with a computational proof assistant to construct a formal proof of the relevant claims. The have also led to more automated approaches, where software is supposed to carry out the task with little user input. In both cases, proof-theoretic methods are invaluable, for designing the relevant logical calculi, for isolating features of proofs that enable one to cut down the search space and traverse it effectively, and for replacing proof search with calculation wherever possible.

For more information on automated reasoning, see [16, 29]. For more information on formally verified mathematics, see [41], or the December 2008 issue of the *Notices of the American Mathematical Society*, which was devoted to formal proof.

8.4.5 Proof Complexity

Finally, the field of proof complexity combines methods and insights from proof theory and computational complexity. For example, the complexity class NP can be viewed as the class of problems for which an affirmative answer has a short (polynomial-size) proof in a suitable calculus. Thus the conjecture that NP is not

equal to co-NP (which is weaker than saying P is not equal to NP) is equivalent to saying that in general there is no propositional calculus that has efficient proofs of every tautology. Stephen Cook has suggested that one way of building up to the problem is to show that *particular* proof systems are not efficient, by establishing explicit lower bounds. Such information is also of interest in automated reasoning, where one wishes to have a detailed understanding of the types of problems that can be expected to have short proofs in various calculi. The works [21, 28, 32, 40] provide excellent introductory overviews.

References

1. Avigad, J. (2000). Interpreting classical theories in constructive ones. *Journal of Symbolic Logic, 65*, 1785–1812.
2. Avigad, J. (2002). Saturated models of universal theories. *Annals of Pure and Applied Logic, 118*, 219–234.
3. Avigad, J. (2003). Number theory and elementary arithmetic. *Philosophia Mathematica, 11*, 257–284.
4. Avigad, J. (2004). Forcing in proof theory. *Bulletin of Symbolic Logic, 10*, 305–333.
5. Avigad, J., & Feferman, S. Gödel's functional ("Dialectica") interpretation. In [9] (pp. 337–405).
6. Barwise, J. (Ed.), (1977). *The handbook of mathematical logic*. Amsterdam: North-Holland. [Contains a number of introductory articles on proof theory and related topics.]
7. Beeson, M. J. (1985). *Foundations of constructive mathematics*. Berlin: Springer.
8. Bertot, Y., & Castéran, P. (2004). *Interactive theorem proving and program development: Coq'Art: The calculus of inductive constructions*. Berlin: Springer.
9. Buss, S. R. (Ed.). (1998). *The handbook of proof theory*. Amsterdam: North-Holland. [Provides a definitive overview of the subject.]
10. Buss, S. R. An introduction to proof theory. In Buss [9] (pp. 1–78).
11. Buss, S. R. First-order proof theory of arithmetic. In Buss [9] (pp. 79–147)
12. Feferman, S. Theories of finite type related to mathematical practice. In Barwise [6] (pp. 913–971).
13. Friedman, H. (to appear). *Boolean relation theory and incompleteness*. Cambridge University Press.
14. Goodstein, R. L. (1957). *Recursive number theory: A development of recursive arithmetic in a logic-free equation calculus*. Amsterdam: North-Holland.
15. Hájek, P., & Pudlák, P. (1993). *Metamathematics of first-order arithmetic*. Berlin: Springer.
16. Harrison, J. (2009). *Handbook of practical logic and automated reasoning*. Cambridge: Cambridge University Press.
17. Hilbert, D. (1922). Neubegründung der Mathematik. Erste Mitteilung. *Abhandlungen aus dem mathematischen Seminar der Hamburgischen Universität, 1*, 157–177. Translated by Ewald, W. (1996). As the new grounding of mathematics. First report. In Ewald, W. (Ed.), *From Kant to Hilbert: A source book in the foundations of mathematics* (Vol. 2, pp. 1115–1134) Oxford: Clarendon.
18. Kaye, R. (1991). *Models of Peano arithmetic*. Oxford: Clarendon.
19. Keisler, H. J. (2006). Nonstandard arithmetic and reverse mathematics. *Bulletin of Symbolic Logic, 12*, 100–125.
20. Kohlenbach, U. (2008). *Applied proof theory: Proof interpretations and their use in mathematics*. Berlin: Springer. [An introduction to proof mining.]

21. Krajíček, J. (1995). *Bounded arithmetic, propositional logic, and complexity theory*. Cambridge: Cambridge University Press.
22. Kunen, K. (1980). *Set theory: An introduction to independence proofs*. Amsterdam: North-Holland.
23. Negri, S., & von Plato, J. (2008). *Structural proof theory.* Cambridge: Cambridge University Press.
24. Paris, J., & Harrington, L. A mathematical incompleteness in Peano arithmetic. In [6] (pp. 1133–1142)
25. Pierce, B. (2004). *Advanced topics in types and programming languages*. Cambridge, MA: MIT Press.
26. Pohlers, W. Subsystems of set theory and second order number theory. In Buss [9] (pp. 209–335).
27. Pohlers, W. (2009). *Proof theory: The first step into impredicativity*. Berlin: Springer. [An introduction to ordinal analysis.]
28. Pudlák, P. The lengths of proofs. In [9] (pp. 547–637).
29. Robinson, J. A., & Voronkov, A. (Eds.). (2001). *Handbook of automated reasoning* (Vols. 1 and 2). Amsterdam/New York: Elsevier; Cambridge: MIT Press.
30. Sambin, G. (Ed.). (1998). *Twenty-five years of constructive type theory*. Oxford: Clarenden.
31. Schwichtenberg, H. Proof theory: Some aspects of cut-elimination. In Barwise [6] (pp. 867–895).
32. Segerlind, N. (2007). The complexity of propositional proofs. *Bulletin of Symbolic Logic, 13*, 417–481.
33. Sieg, W. (1985). Fragments of arithmetic. *Annals of Pure and Applied Logic, 28*, 33–72.
34. Sieg, W. (1999). Hilbert's programs: 1917–1922. *Bulletin of Symbolic Logic, 5*, 1–44.
35. Simpson, S. G. (1999). *Subsystems of second-order arithmetic*. Berlin: Springer
36. Takeuti, G. (1987). *Proof theory* (2nd ed.). Amsterdam: North-Holland.
37. Troelstra, A. S. Realizability. In [9] (pp. 407–473).
38. Troelstra, A. S., & Schwichtenberg, H. (2000). *Basic proof theory* (2nd ed.). Cambridge: Cambridge University Press. [An introductory text.]
39. Troelstra, A. S., & van Dalen, D. (1988). *Constructivism in mathematics: An introduction* (vols. 1 and 2). Amsterdam: North-Holland. [An overview of constructive mathematics.]
40. Urquhart, A. (1995). The complexity of propositional proofs. *Bulletin of Symbolic Logic, 1*, 425–467.
41. Wiedijk, F. (2006). *The seventeen provers of the world*. Berlin: Springer.
42. Zach, R. (2006). *Hilbert's program then and now*. In D. Jacquette (Ed.), *Philosophy of logic* (pp. 411–447). Amsterdam: Elsevier. [A nice historical overview.]