Post's Problem and the Priority Method in CIC

Haoyi Zeng[1], Yannick Forster[2], and Dominik Kirst[3]

[1] Saarland University, Germany
[2] Inria Paris, France
[3] Ben-Gurion University, Israel

Abstract

We describe our formalisation of a solution to Post's Problem using the priority method in synthetic computability theory. Compared to a usual, analytic approach employing explicit models of computation, a synthetic approach axiomatically considers all functions $\mathbb{N} \to \mathbb{N}$ to be computable. We work in the Calculus of Inductive Constructions and mechanise all proofs in the Coq proof assistant.

Background Posed by Emil Post in 1944 [16], Post's problem asks whether there are semi-decidable, yet undecidable predicates that are not Turing-reducible from the halting problem. Post's problem has been a crucial open question driving research in computability theory until a breakthrough came with the positive solution by Friedberg and Muchnik [9, 14] in 1956/57. They introduced independently what is now known as the priority method, in order to show that there exist two semi-decidable, Turing-reduction incomparable degrees. The priority method has since become a cornerstone in the field of computability theory, essential for exploring and understanding the structure of computability degrees [12, 13, 20].

Today, virtually every textbook on computability theory (e.g. [23, 18, 22, 15]) discusses Post's problem and the use of the priority method. From the perspective of machine-checked proofs, the interactive theorem proving community has successfully formalised cutting-edge mathematics in several proof assistants, however, formalising computability theory remains a challenge. A main intricacy is the use of models of computation for formal proofs [8], due to the level of uninteresting details involved that stay invisible on paper.

A solution is proposed by *synthetic* computability [17, 1], which exploits constructive mathematics as its foundation. In a synthetic approach to computability theory, *every* function is considered computable. For instance, the decidability of a predicate $P : X \to \mathbb{P}$ is now defined as $\exists f : X \to \mathbb{B}.\ P\ x \leftrightarrow f\ x = \mathsf{true}$, which eliminates the need to show f computable in a model.

Since the Calculus of Inductive Constructions (CIC) is a constructive system where the law of excluded middle stays consistent even when assuming axioms for synthetic computability, it is natural to ask questions of constructive reverse analysis. The formalisation and constructive analysis of synthetic computability have received attention in recent years, encompassing the study of many-one reduction, Post's theorem, the arithmetic hierarchy, and Turing computability [3, 4, 7, 5, 11]. In 2021, Andrej Bauer has posed the challenge to "give a synthetic proof of Friedberg-Mucnik theorem" [2].

Due to the historic importance of Post's problem, we consider the successful completion of this challenge a milestone in synthetic computability and the formalisation of computability theory. Notably, the necessity for the priority method will play a crucial role in the further development of machine-checked synthetic computability.

Low Simple Predicate Soare's solution to Post's problem [21] constructs a so-called low simple predicate directly, rather than proving the full Friedberg-Muchnik theorem constructing two incomparable predicates. Since a synthetic notion of simple predicates has been defined in previous work [4], we here focus on the aspect of lowness. A predicate P is *low* if its Turing jump P' is reducible to the halting problem H, ie. $P' \preceq_T H$.

This leaves us with three key questions: 1) What is the simplest technique to establish that a predicate is reducible to the halting problem? 2) How can we formalise the priority method synthetically? 3) How can we instantiate the method to obtain a low simple predicate? We address these questions within the framework of synthetic computability, following Soare [23].

1) Limit computablility We conjecture that the simplest way to give a reduction to the halting problem in our case is by using the notion of limit computability [19, 10]. It is equivalent to being reducible to the halting problem, but easier to establish. We use the notion of the characteristic relation $\hat{P} : X \to \mathbb{B} \to \mathbb{P}$ of a predicate P, where $\hat{P}\ x\ \mathsf{true} \leftrightarrow P\ x$ and $\hat{P}\ x\ \mathsf{false} \leftrightarrow \neg P\ x$. We call $P : X \to \mathbb{P}$ *limit-computable* if there is a function $f : X \to \mathbb{N} \to \mathbb{B}$ such that

$$\forall x\ b.\ \hat{P}(x, b) \leftrightarrow \exists n.\ \forall m > n.\ f(x, m) = b.$$

Based on this definition and the previously developed Turing computability in synthetic computability [6], we have already verified the limit lemma, which states that a predicate P is limit-computable if and only if P is reducible to H.

This brings us to the position that we just need to prove limit computability of P' to prove lowness of the to-be-constructed simple predicate P.

2) The finite injury priority method The priority method can be used to construct semi-decidable predicates P satisfying an infinite set of requirements R_e. In order to construct P, the following inductive predicate computably binds a list L to each stage n, where then $P\ x\ :=\ \exists n\ L.\ n \rightsquigarrow L \land x \in L$. The predicate is parameterized by an extension $\gamma : \mathbb{N}^* \to \mathbb{N} \to \mathbb{N} \to \mathbb{P}$, which is used to determine whether an element can enter the predicate at step $n+1$ and can recursively depend on L that fulfils $n \rightsquigarrow L$.

$$\frac{}{0 \rightsquigarrow [\,]} \qquad \frac{n \rightsquigarrow L \quad \gamma^L_n\ x}{n+1 \rightsquigarrow x :: L} \qquad \frac{n \rightsquigarrow L \quad \forall x.\ \neg\ \gamma^L_n\ x}{n+1 \rightsquigarrow L}$$

In this work, we consider the simplest form of the priority method, the finite injury priority method, as originally developed by Friedberg and Muchnik, which is sufficient for constructing low simple predicates. The term "finite injury" refers to the possibility that some of the requirements in R_e on P might be broken during the construction of P due to the satisfaction of others, also known as "injury". However, because the injury is finite, P will ultimately satisfy all R_e as required.

3) Instantiation to a low simple predicate To construct a low simple predicate, γ is instantiated to an appropriate extension.

We took advantage of formalised proof to achieve this goal by constructing the proof in a modular way. First, we considered an extension γ, ensuring that some requirements R_e are met, which implies P is a simple predicate.

In the second step, we observed that certain conditions in γ can be abstracted to any convergent function ω. According to Soare's construction, we refined the results in oracle computability, providing a suitable definition to concretize this function ω, so that some requirements N_e will be finitely injured at some stages, but are eventually satisfied, which entail that this predicate is low.

Future work Currently, on top of a typical formulation of Church's thesis for synthetic computability [1], we assume the law of excluded middle for all proofs, i.e. full classical logic. As a next step, we plan to weaken this assumptions as much as possible, e.g. by restricting the logical complexity of propositions in axioms, and then perform a constructive reverse analysis.

Acknowledgements We would like to thank Gert Smolka for the discussions, and the anonymous reviewers for their helpful feedback.

References

[1] Andrej Bauer. First steps in synthetic computability theory. *Electronic Notes in Theoretical Computer Science*, 155:5–31, 2006. doi:10.1016/j.entcs.2005.11.049.

[2] Andrej Bauer. Synthetic mathematics with an excursion into computability theory (slide set). *University of Wisconsin Logic seminar*, 2020. URL: http://math.andrej.com/asset/data/madison-synthetic-computability-talk.pdf.

[3] Yannick Forster. Computability in constructive type theory. 2021. doi:10.22028/D291-35758.

[4] Yannick Forster and Felix Jahn. Constructive and synthetic reducibility degrees: Post's problem for many-one and truth-table reducibility in Coq. In *CSL 2023-31st EACSL Annual Conference on Computer Science Logic*, 2023. doi:10.4230/LIPIcs.CSL.2023.21.

[5] Yannick Forster and Dominik Kirst. Synthetic Turing reducibility in constructive type theory. 28th International Conference on Types for Proofs and Programs (TYPES 2022), 2022. URL: https://types22.inria.fr/files/2022/06/TYPES_2022_paper_64.pdf.

[6] Yannick Forster, Dominik Kirst, and Niklas Mück. Oracle computability and Turing reducibility in the calculus of inductive constructions. In Chung-Kil Hur, editor, *Programming Languages and Systems - 21st Asian Symposium, APLAS 2023, Taipei, Taiwan, November 26-29, 2023, Proceedings*, volume 14405 of *Lecture Notes in Computer Science*, pages 155–181. Springer, 2023. doi:10.1007/978-981-99-8311-7_8.

[7] Yannick Forster, Dominik Kirst, and Niklas Mück. The Kleene-Post and Post's theorem in the calculus of inductive constructions. 2024. doi:10.4230/LIPIcs.CSL.2024.29.

[8] Yannick Forster, Fabian Kunze, and Maximilian Wuttke. Verified programming of Turing machines in Coq. In *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*, pages 114–128, 2020. doi:10.1145/3372885.3373816.

[9] Richard M Friedberg. Two recursively enumerable sets of incomparable degrees of unsolvability (solution of Post's problem, 1944). *Proceedings of the National Academy of Sciences*, 43(2):236–238, 1957. doi:10.1073/pnas.43.2.236.

[10] E Mark Gold. Limiting recursion. *The Journal of Symbolic Logic*, 30(1):28–48, 1965. doi:10.2307/2270580.

[11] Dominik Kirst, Yannick Forster, and Niklas Mück. Synthetic Versions of the Kleene-Post and Post's Theorem. 28th International Conference on Types for Proofs and Programs (TYPES 2022), 2022. URL: https://types22.inria.fr/files/2022/06/TYPES_2022_paper_65.pdf.

[12] Alistair H Lachlan. The priority method for the construction of recursively enumerable sets. In *Cambridge Summer School in Mathematical Logic: Held in Cambridge/England, August 1–21, 1971*, pages 299–310. Springer, 2006. doi:10.1007/BFb0066779.

[13] Manuel Lerman and Robert Soare. *d*-simple sets, small sets, and degree classes. *Pacific Journal of Mathematics*, 87(1):135–155, 1980. doi:10.2140/pjm.1980.87.135.

[14] Albert A Muchnik. On the unsolvability of the problem of reducibility in the theory of algorithms. In *Dokl. Akad. Nauk SSSR*, volume 108, pages 194–197, 1956.

[15] Piergiorgio Odifreddi. *Classical recursion theory: The theory of functions and sets of natural numbers*. Elsevier, 1992. doi:10.2307/2274492.

[16] Emil L Post. Recursively enumerable sets of positive integers and their decision problems. 1944. doi:10.1090/s0002-9904-1944-08111-1.

[17] Fred Richman. Church's thesis without tears. *The Journal of symbolic logic*, 48(3):797–803, 1983. doi:10.2307/2273473.

[18] Hartley Rogers Jr. *Theory of recursive functions and effective computability.* MIT press, 1987. doi:10.2307/2271523.

[19] Joseph R Shoenfield. On degrees of unsolvability. *Annals of mathematics*, 69(3):644–653, 1959. doi:10.2307/1970028.

[20] Stephen G Simpson. Degrees of unsolvability: a survey of results. In *Studies in Logic and the Foundations of Mathematics*, volume 90, pages 631–652. Elsevier, 1977. doi:10.1016/S0049-237X(08)71117-0.

[21] Robert I Soare. The infinite injury priority method1. *The Journal of Symbolic Logic*, 41(2):513–530, 1976. doi:10.1017/S0022481200051598.

[22] Robert I Soare. Recursively enumerable sets and degrees. *Bulletin of the American Mathematical Society*, 84(6):1149–1181, 1978.

[23] Robert I Soare. Turing computability. *Theory and Applications of Computability. Springer*, 2016. doi:10.1007/978-3-642-31933-4.