Final Bachelor's Project Talk

Post's Problem and The Priority Method in Synthetic Computability

Haoyi Zeng

Advisors: Yannick Forster and Dominik Kirst
Supervisor: Prof. Gert Smolka
Programming Systems Lab

Date:04.09.24

Bachelor Thesis

Corollary 6.55 *Assuming Σ_1-LEM, there exists a low simple predicate.*

Oracle Computability and Turing Reducibil

[2] Ben-Gurion University of the Negev, Beer-Sheva, Israel
kirst@cs.bgu.ac.il
[3] Saarland University and MPI-SWS, Saarland Informatics Campus, Saarbr
Germany
s8nimuec@stud.uni-saarland.de

Abstract. We develop synthetic notions of oracle computability an Turing reducibility in the Calculus of Inductive Constructions (CIC the constructive type theory underlying the Coq proof assistant. As usua in synthetic approaches, we employ a definition of oracle computation based on meta-level functions rather than object-level models of compu tation, relying on the fact that in constructive systems such as CIC al definable functions are computable by construction. Such an approac lends itself well to machine-checked proofs, which we carry out in Coq.

There is a tension in finding a good synthetic rendering of the higher order notion of oracle computability. On the one hand, it has to be in formative enough to prove central results, ensuring that all notions ar faithfully captured. On the other hand, it has to be restricted enoug to benefit from axioms for synthetic computability, which usually con cern first-order objects. Drawing inspiration from a definition by Andre Bauer based on continuous functions in the effective topos, we use a no tion of sequential continuity to characterise valid oracle computations.

As main technical results, we show that Turing reducibility forms a upper semilattice, transports decidability, and is strictly more expressiv than truth-table reducibility, and prove that whenever both a predicat p and its complement are semi-decidable relative to an oracle q, then Turing-reduces to q.

Keywords: Type theory · Logical foundations · Synthetic computabilit theory · Coq proof assistant

* Yannick Forster received funding from the European Union's Horizon 2020 and innovation programme under the Marie Skłodowska-Curie grant agreen 101024493. Dominik Kirst is supported by a Minerva Fellowship of the Stiftung Gesellschaft fuer die Forschung mbH

Constructive and Synthetic Reducibility Degrees

Post's Problem for Many-one and Truth-table Reducibility in Coq

Yannick Forster
Saarland University, Saarland Informatics Campus, Saarbrücken, Germany
Inria, Gallinette Project-Team, Nantes, France

Felix Jahn
Saarland University, Saarland Informatics Campus, Saarbrücken, Germany

Abstract

We present a constructive analysis and machine-checked theory of one-one, many-one, and trut reductions based on synthetic computability theory in the Calculus of Inductive Construction

assume classical axioms, not even Markov's principle, still yielding the expected strong resu

2012 ACM Subject Classification Theory of computation → Constructive mathematics; Type

Keywords and phrases type theory, computability theory, constructive mathematics, Coq

Digital Object Identifier 10.4230/LIPIcs.CSL.2023.16

Supplementary Material github.com/uds-psl/coq-synthetic-computability/tree/redde

Funding *Yannick Forster*: received funding from the European Union's Horizon 2020 researc innovation programme under the Marie Skłodowska-Curie grant agreement No. 101024493.

Acknowledgements We thank Dominik Kirst, Gert Smolka, Lennard Gäher, and Andrej Duden for discussions and feedback on the drafts of this paper, as well as the reviewers for their com

1 Introduction

The founding moment of "computability theory" deserving of the suffix "theory" was r Emil L. Post's 1944 paper [25]. Post introduced the concepts of one-one, many-one truth-table reducibility and identified and answered important questions on the stru of the reducibility degrees induced by these relations. Centrally, Post was interested i question whether there are enumerable but undecidable degrees such that an undecida proof cannot be done by reduction from the halting problem. For many-one and truth reducibility, Post was able to construct such degrees by introducing simple and hypers sets, which are still taught in modern textbook presentations of the field. The que whether an enumerable, undecidable problem which is not Turing-reducible from the h problem exists became known as *Post's problem*, and we reuse the terminology for *problem for many-one reducibility* ($\preceq_m$) and *Post's problem for truth-table reducibility*

Early in his paper, Post remarks *'That mathematicians generally are oblivious importance of this work of Gödel, Church, Turing, Kleene, Rosser and others as it affe subject of their own interest is in part due to the forbidding, diverse and alien formalis*

31st EACSL Annual Conference on Computer Science Logic (CSL 2023).
Editors: Bartek Klin and Elaine Pimentel; Article No. 16; pp. 16:1–16:21
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

The Kleene-Post and Post's Theorem in the

Abstract

The Kleene-Post theorem and Post's theorem are two central and historically important results in the development of oracle computability theory, clarifying the structure of Turing reducibility degrees. They state, respectively, that there are incomparable Turing degrees and that the arithmetical hierarchy is connected to the relativised form of the halting problem defined via Turing jumps.

We study these two results in the calculus of inductive constructions (CIC), the constructive type theory underlying the Coq proof assistant. CIC constitutes an ideal foundation for the formalisation of computability theory for two reasons: First, like in other constructive foundations, computable functions can be treated via axioms as a purely synthetic notion rather than being defined in terms of a concrete analytic model of computation such as Turing machines. Furthermore and uniquely, CIC allows consistently assuming classical logic via the law of excluded middle or weaker variants on top of axioms for synthetic computability, enabling both fully classical developments and taking the perspective of constructive reverse mathematics on computability theory.

In the present paper, we give a fully constructive construction of two Turing-incomparable degrees à la Kleene-Post and observe that the classical content of Post's theorem seems to be related to the arithmetical hierarchy of the law of excluded middle due to Akama et. al. Technically, we base our investigation on a previously studied notion of synthetic oracle computability and contribute the first consistency proof of a suitable enumeration axiom. All results discussed in the paper are mechanised and contributed to the Coq library of synthetic computability.

2012 ACM Subject Classification Theory of computation → Constructive mathematics; Type theory

Keywords and phrases Constructive mathematics, Computability theory, Logical foundations, Constructive type theory, Interactive theorem proving, Coq proof assistant

Supplementary Material github.com/uds-psl/coq-synthetic-computability/tree/code-paper-kleene-post-post.

Funding *Yannick Forster*: received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 101024493.
Dominik Kirst: is supported by a Minerva Fellowship of the Minerva Stiftung Gesellschaft für die Forschung mbH.

1 Introduction

We study two well-known results in computability theory from the perspective of synthetic mathematics: the Kleene-Post theorem [30], stating that there are incomparable Turing degrees[1] and Post's theorem [39], establishing a close link between Turing jumps and the

[1] The seminal 1954 paper by Kleene and Post establishes various other results besides this one. In

Main Result [Muchnik 1956] [Friedberg 1957] [Lerman and Soare 1980] [Nemoto 2024]

Corollary 6.55 *Assuming* Σ_1-LEM, *there exists a low simple predicate.*

We mechanise a solution to **Post's problem** in **synthetic computability**

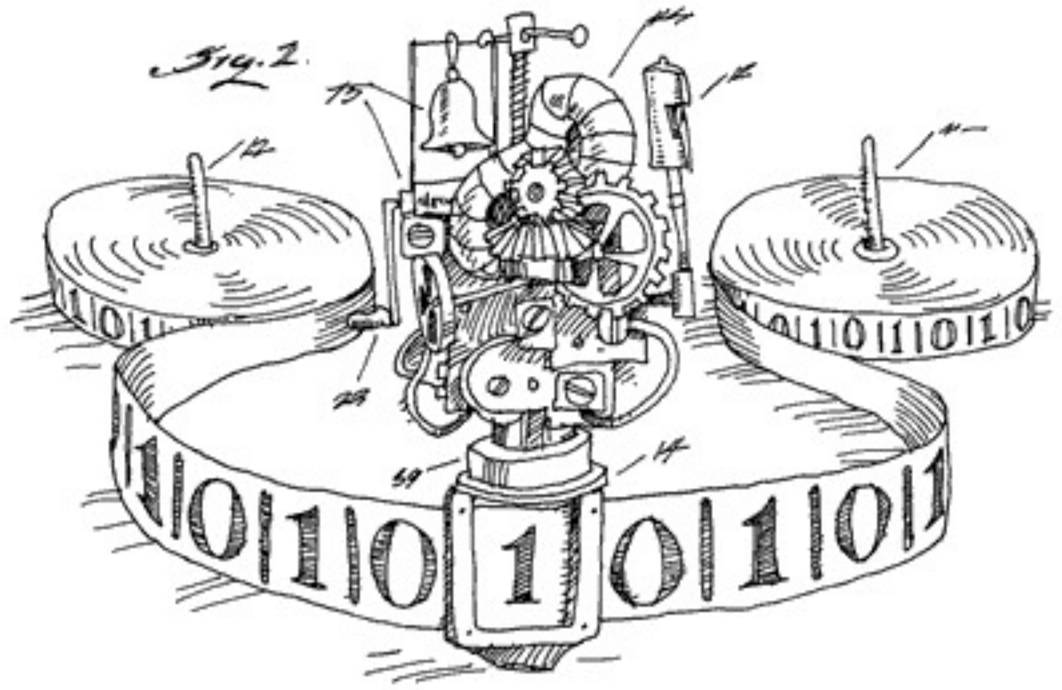
Background

Synthetic Computability [Richman 1983] [Bauer 2006]

P is **Decidable**: $\exists f : X \to \mathbb{B} \,.\, P\ x \leftrightarrow f\ x = \text{true}$ ~~$\land\ f$ is computable~~

What is computable ?

Turing Machine



λ-Calculus

$$\begin{aligned} \mathsf{fix}\ F &:= \lambda x.\, \mathsf{fix}'\ \mathsf{fix}'\ F\ x \\ \mathsf{fix}' &:= \lambda f, F.\, F\ (\lambda x.\, f\ f\ F\ x) \end{aligned}$$

Synthetic Computability

Synthetic Computability

A predicate $P : X \to \mathbb{P}\mathrm{rop}$ is

Decidable $\exists\, f : X \to \mathbb{B}\,.\, P\ x \leftrightarrow\ f\ x = \mathsf{true}$

Semi-decidable $\exists\, f : X \to \mathbb{N} \to \mathbb{B}\,.\, P\ x \leftrightarrow \exists n\,.\, f\ x\ n = \mathsf{true}$

For any $P : X \to \mathbb{P}\mathrm{rop}$ and $Q : Y \to \mathbb{P}\mathrm{rop}$

Many-one Reduction

$$P \preceq_m Q \ := \quad \exists f : X \to Y\,.\, \forall x\,.\, P\ x \leftrightarrow Q\ (f\ x)$$

Church's Thesis

There is an enumerator $\theta : \mathbb{N} \to (\mathbb{N} \rightharpoonup \mathbb{N})$ for all partial functions:

$$\forall f : \mathbb{N} \rightharpoonup \mathbb{N}.\ \exists e.\ \forall x\ v.\ f\ x \downarrow v \leftrightarrow \theta_e\ x \downarrow v$$

"Does a Turing machine halt on a given input?"

Halting Problem $H\, x \;:=\; \theta_x\ x \downarrow$

Why CIC ? CIC + CT + LEM is (believed to be) consistent

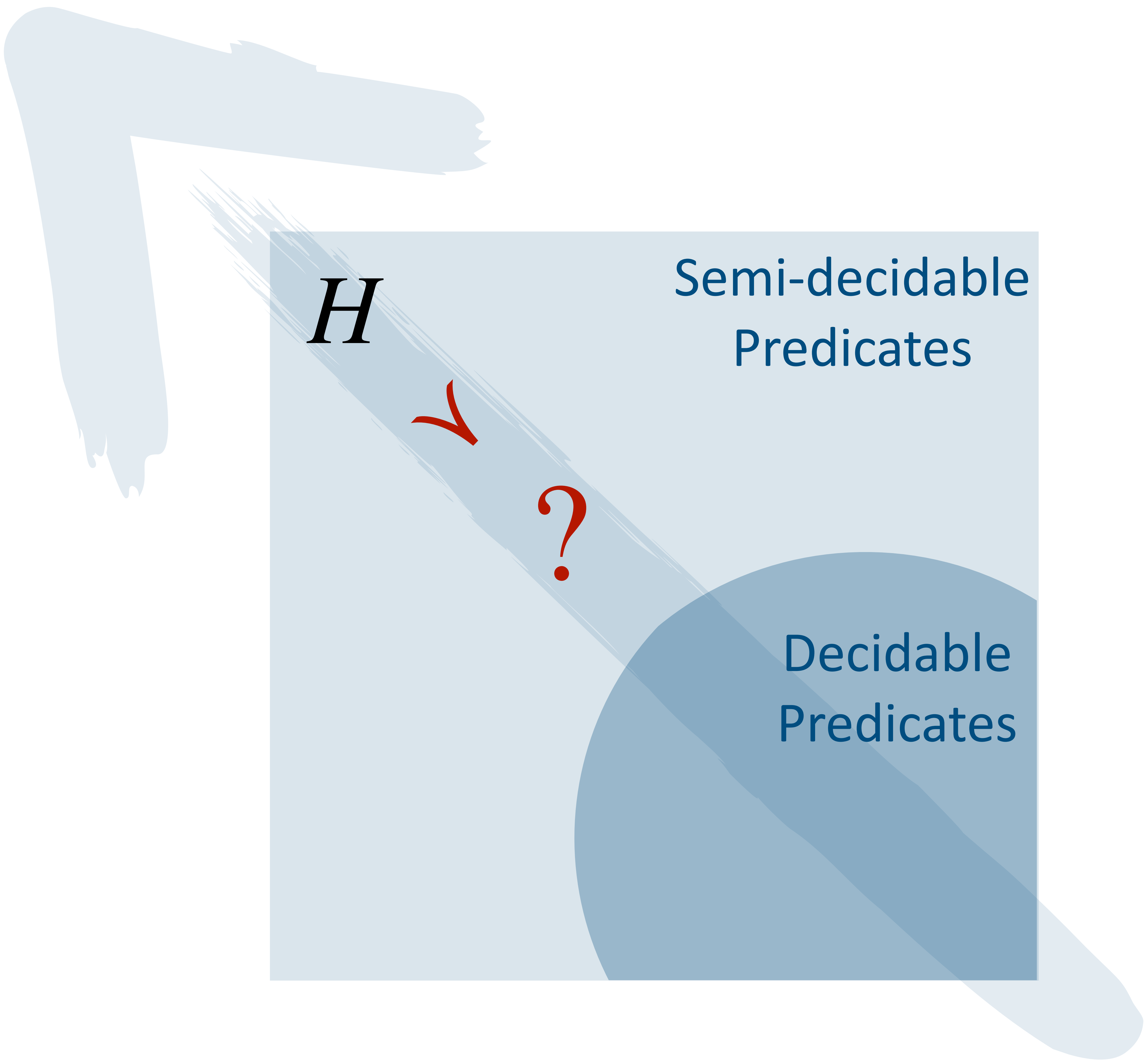
Post’s Problem

“Is there an undecidable,

semi-decidable predicate

that is strictly easier than the Halting problem?”

- Post, 1944

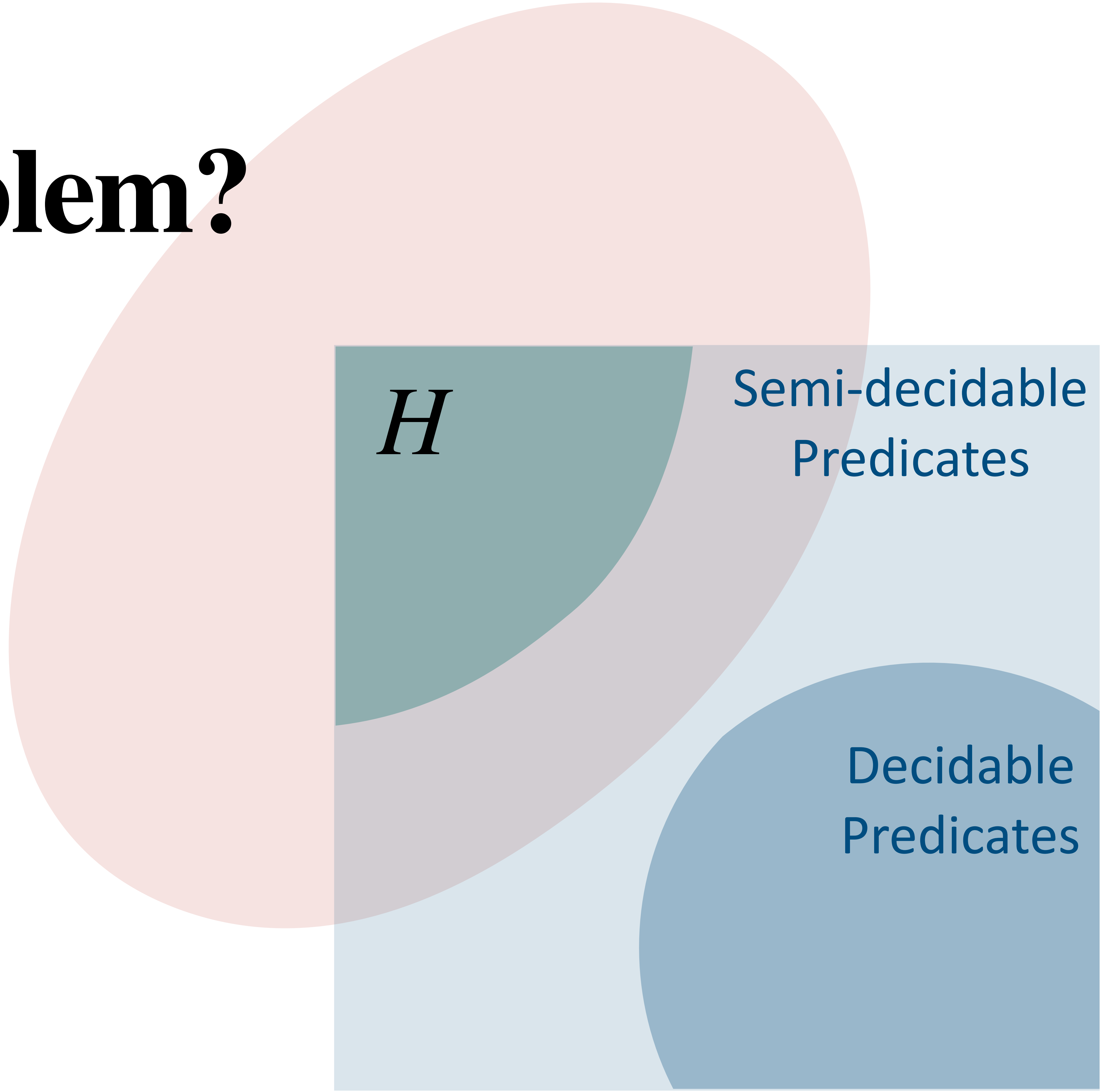


Easier than Halting Problem?

P is reducible to Q

Many-one Reduction: $H \preceq_m P$

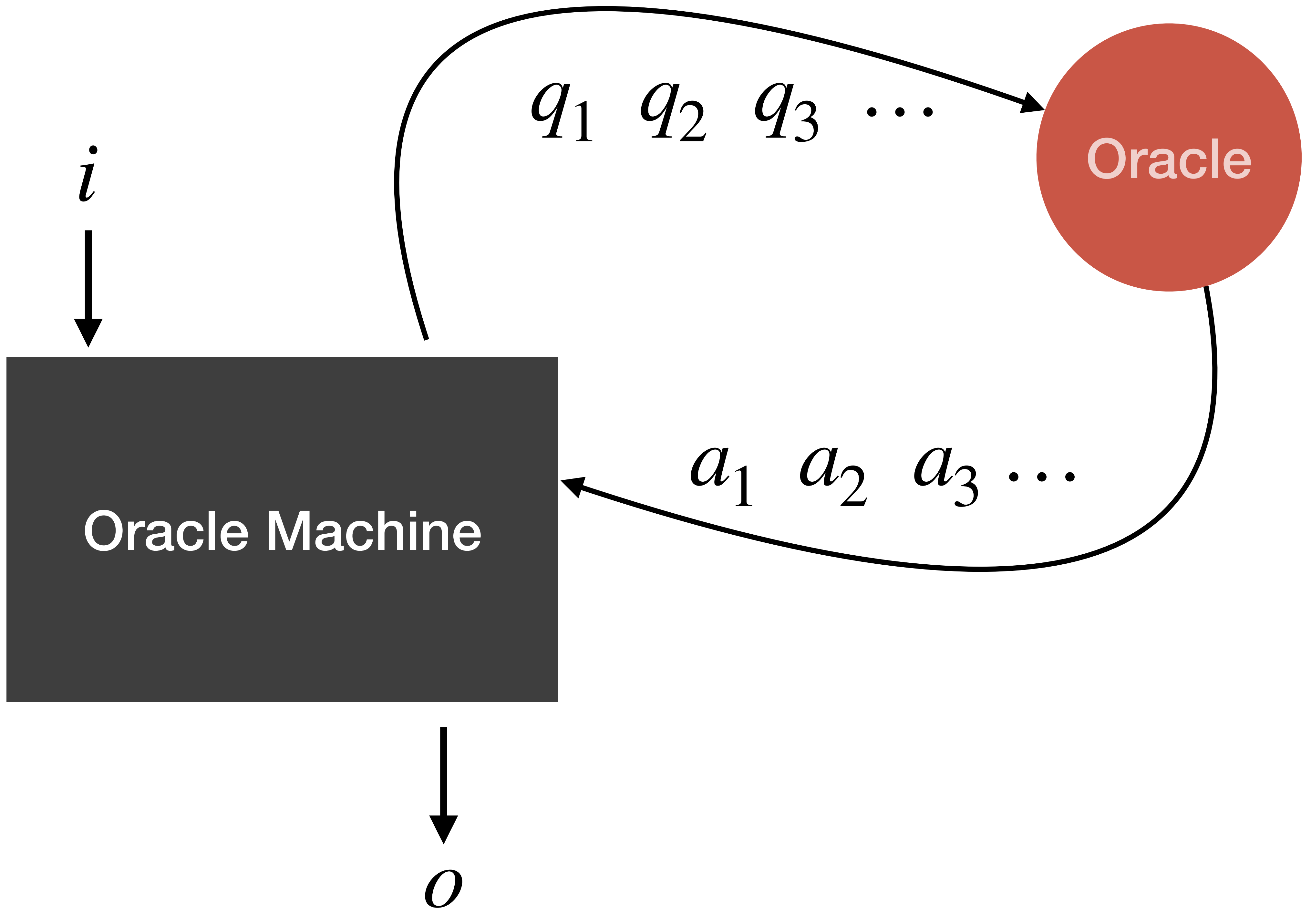
Turing Reduction: $H \preceq_T P$



Consider reductions in a more general sense, i.e., Turing reduction, which is also the problem Post left open in his paper [Post 1944].

Oracle Computability in synthetic computability

Synthetic notation of Oracle Computable (O. C.) :



O.C. is capturing by some underlying computable object:

- The first definition of oracle computability by modulus continuity [Bauer 2021]
- A series definitions of oracle computability in CIC [Forster 2021][Kirst, Mück & Forster 2022] [Forster, Kirst & Mück 2023]
- Oracle modalities [Swan 2024]

The First Challenge

Oracle computability is non-trivial in synthetic computability

[Forster, Kirst & Mück 2023]

[redacted]

But,

We need to consider oracle machines that can be "executed"

(Other frameworks are similar)

Definition 3.22 (Step-Indexed Oracle Machines) *For any given datatype* T, *a function* $\phi : (\mathbb{N} \to \mathbb{N}^* \rightharpoonup \mathbb{N} + \mathsf{T}) \to (\mathbb{N} \to \mathbb{P}\mathrm{rop}) \to \mathbb{N} \to \mathbb{N} \to \mathsf{T}^?$ *is called a step-indexed oracle machine, if it meets the following requirement for any oracle machine* e *with a semi-decidable oracle* $p : \mathbb{N} \to \mathbb{P}\mathrm{rop}$:

$$\forall x\, b.\, \Xi_e\, \hat{p}\, x\, b \to \lim_{n \to \alpha}$$

where we abbreviate $\phi_e^{p_n}\ x\ n$ *as* $\phi_e^p(x)[n]$.

Definition 3.30 (Use Functions) *Given a step-indexed oracle machine* ϕ, *a use function* $u : \mathbb{N} \to (\mathbb{N} \to \mathbb{P}\mathrm{rop}) \to \mathbb{N} \to \mathbb{N} \to \mathbb{N}$ *is a function that satisfies the following properties: Let* $w = u_e^q(x)[n]$,

$$\phi_e^q(x)[n]\downarrow \to \forall p.\, p \equiv_w \hat{q}_n \to \Xi_e\, \hat{p}\, e\, \star, \tag{3.1}$$

$$\phi_e^q(x)[n]\downarrow \to q_{n+1} \equiv_w q_n \to \phi_e^q(x)[n+1]\downarrow, \tag{3.2}$$

$$\phi_e^q(x)[n]\downarrow \to q_{n+1} \equiv_w q_n \to u_e^q(x)[n+1] = w, \tag{3.3}$$

Solutions to Post’s Problem

Finite extension method [Post 1944]

Simple Predicate

... **After 12 years**

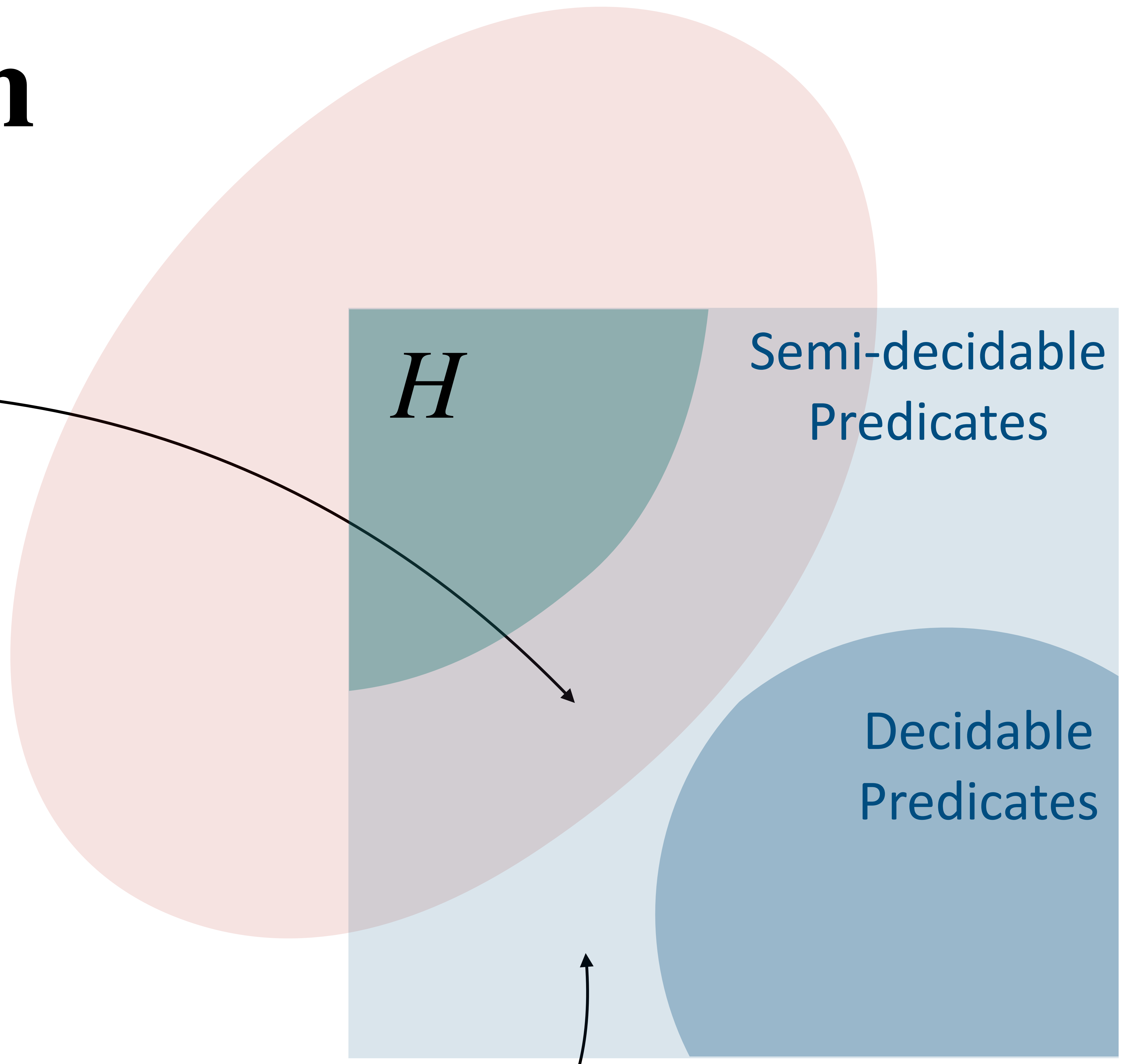
The Priority Method

Friedberg–Muchnik Theorem

[Mučnik 1956] [Friedberg 1957]

Low Simple Predicate

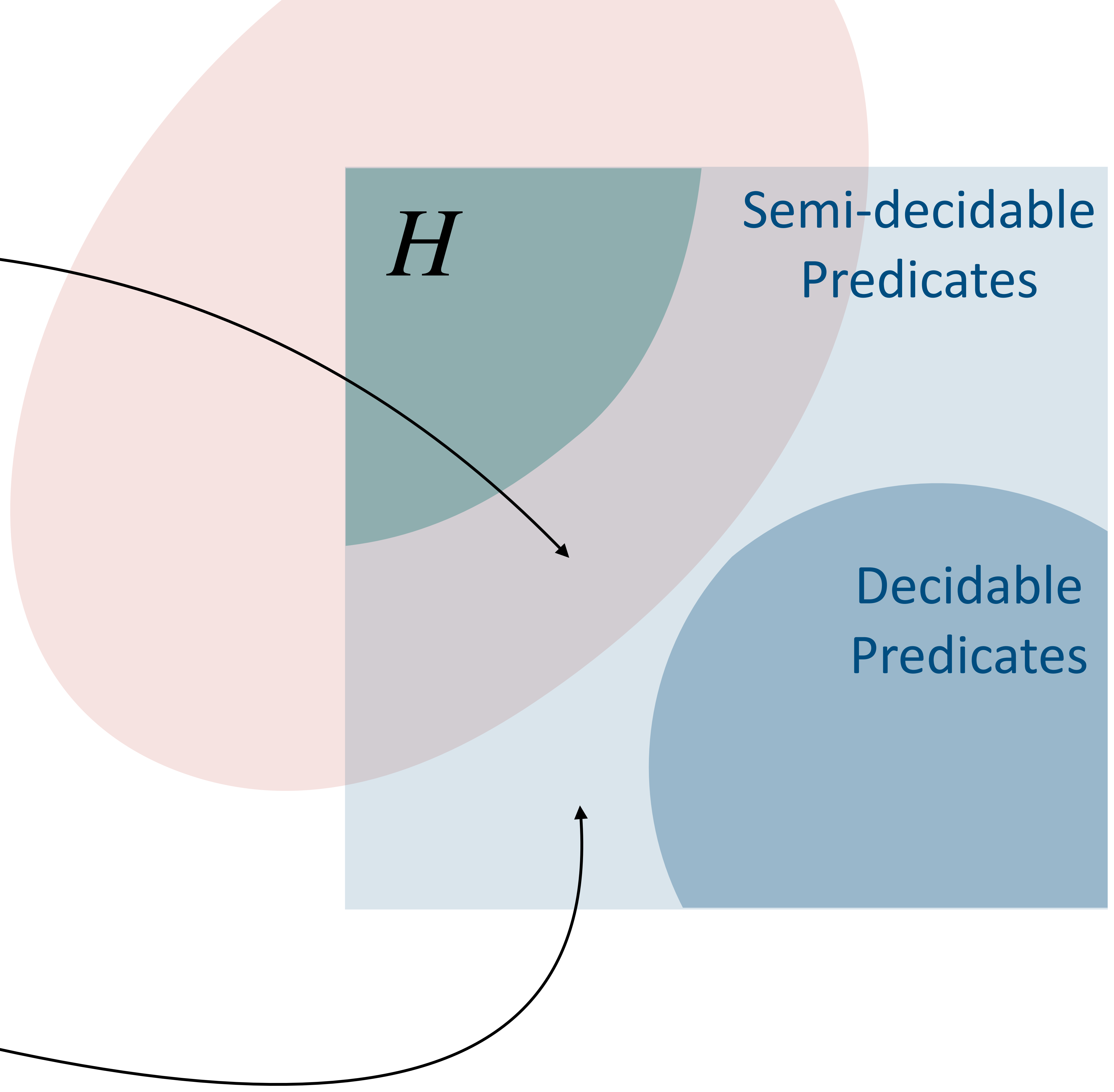
[Lerman & Soare 1980] [Soare 1999]



Solutions to Post's Problem in synthetic computability

in synthetic computability [Forster & Jahn 2023] **Simple Predicate**

in synthetic computability now **Low Simple Predicate**



The Priority Method

Low simple predicates

Not Turing-reducible from the halting problem

semi-decidable, yet undecidable

Low Simple Predicate

$S_{\gamma(\omega)}$ is a semi-decidable low simple predicate

The Priority Method

$$\gamma : \mathbb{N} \to \mathbb{N}^* \to \mathbb{N} \to \mathbb{P}\mathrm{rop}$$

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

$$S_\gamma\, x \coloneqq \exists n\, L.\, (n \rightsquigarrow L) \land (x \in L)$$

γ is computable and functional (Extension) $\Longrightarrow$ S_γ is **semi-decidable**

Simple Extension

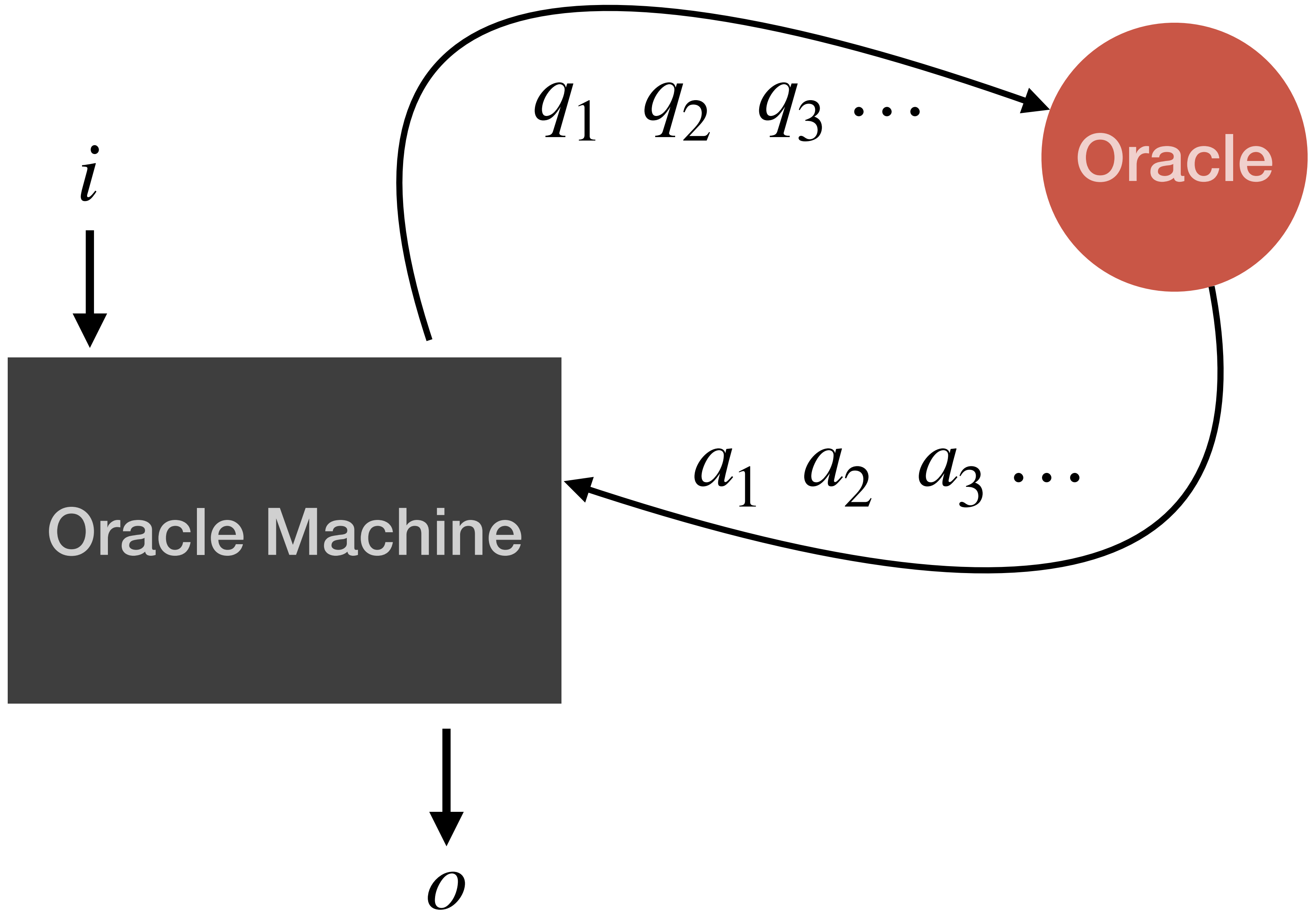
$$\alpha(\omega)^{L}_{n}\ e\ x \ \coloneqq\ x \in \mathcal{W}_e[n] \land \omega^{L}_{n}(e) < x$$
$$\beta(\omega)^{L}_{n}\ e \ \coloneqq\ L \# \mathcal{W}_e[n] \land \exists x.\ \alpha(\omega)^{L}_{n}\ e\ x$$
$$\gamma(\omega)^{L}_{n}\ x \ \coloneqq\ \exists e.\ e < n \land e \text{ is } \mu\, e.\ \beta(\omega)^{L}_{n}\ e \land x \text{ is } \mu x.\ \alpha(\omega)^{L}_{n}\ e\ x$$

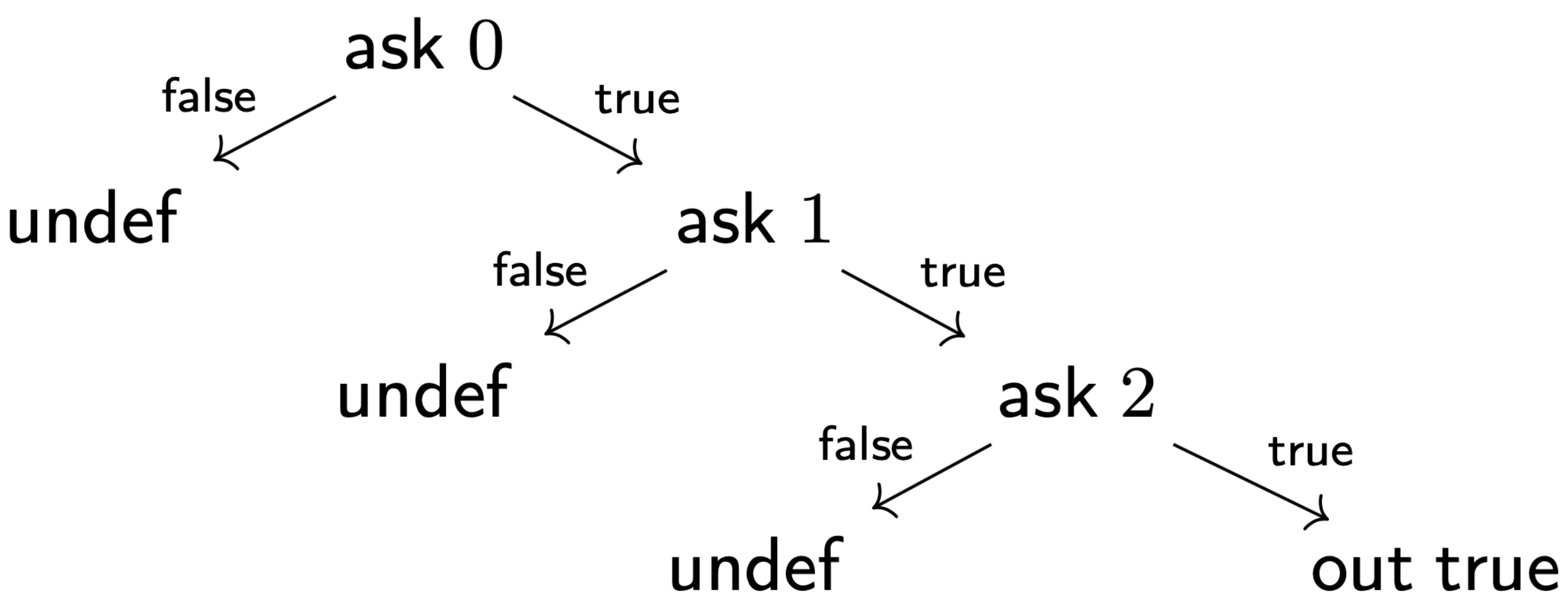
$$P_e(S) \ \coloneqq\ \neg\mathcal{L}(\mathcal{W}_e) \to \mathcal{W}_e \cap S \neq \emptyset$$

$$P_1 \prec P_2 \prec P_3 \prec P_4 \prec P_5 \prec P_6 \bullet \bullet \bullet$$

$\dot{\omega}$ greater than $2e$ and convergent (wall functions) $\Rightarrow$ $S_{\gamma(\omega)}$ is **simple**

Use Function



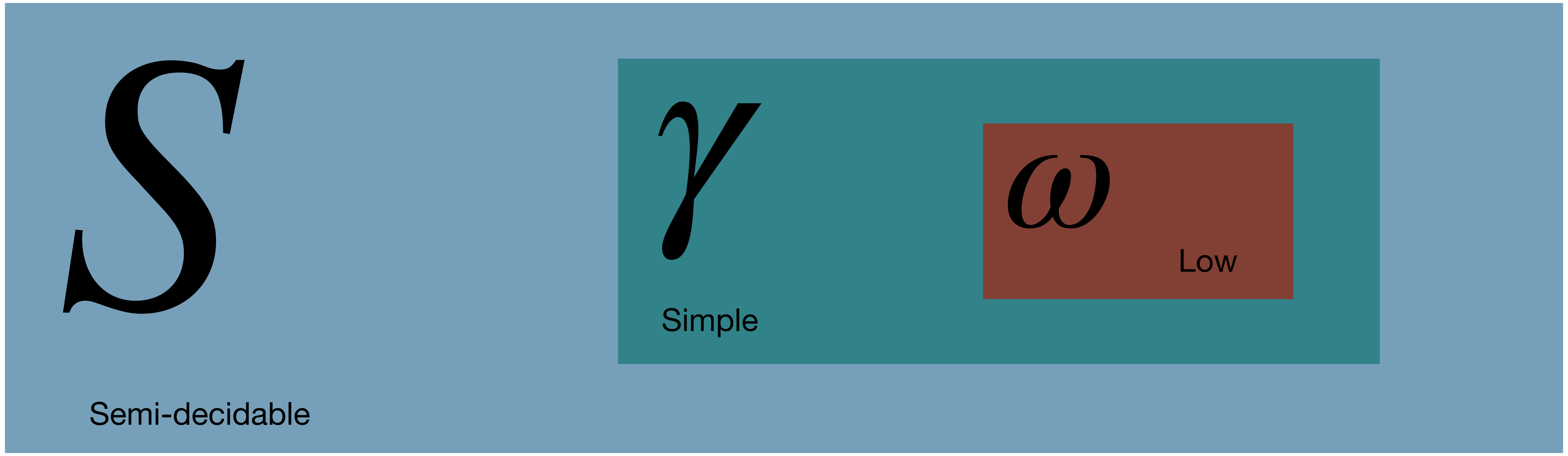


$$u_e^q(x)[n] := \max(\ \textbf{List of questions}\ ) + 1$$

The modulus of continuity

Low Wall

$$
\begin{aligned}
u^L_e(e)[n] &\coloneqq \max_{e' \leqslant e}(u^L_{e'}(e')[n]) \\
\omega^L_n(e) &\coloneqq \max(2 \cdot e, u^L_e(e)[n])
\end{aligned}
$$



The Limit Lemma $\Rightarrow$ $S_{\gamma(\omega)}$ is **Low**

Classical Axioms [Akama 2004]

$$\mathsf{LEM} \coloneqq \forall p : \mathbb{P}\mathrm{rop}.\, p \vee \neg p$$

$$\Sigma_n - \mathsf{LEM} \coloneqq \forall k.\, \forall p : \mathbb{N}^k \to \mathbb{P}\mathrm{rop}.\, \Sigma_n\, p \to \forall v.\, p\, v \vee \neg p\, v$$

$$\Sigma_n \coloneqq \{p \mid \forall x.\, p\, x \leftrightarrow \exists y_1\, \forall y_2\, \exists y_3\, \forall y_4\, \ldots\, R(x, y_1, \ldots, y_n) \wedge \mathcal{D}(R)\}$$

$$\Sigma_n - \mathsf{LEM} \Rightarrow \Sigma_{n-1} - \mathsf{LEM} \Rightarrow \cdots \Rightarrow \Sigma_2 - \mathsf{LEM} \Rightarrow \Sigma_1 - \mathsf{LEM} \Rightarrow \Sigma_0 - \mathsf{LEM}$$

$$\Updownarrow$$

$$\mathsf{LPO} \coloneqq \forall f : \mathbb{N} \to \mathbb{B}.\, (\exists n.\, f\, n = \mathsf{true}) \vee (\forall n.\, f\, n = \mathsf{false})$$

Classical Axioms

TYPES 2024

[Zeng, Forster, and Kirst 2024]

$\Sigma_2 - \text{LEM} \implies \exists S \, . \, S$ is a low simple predicate

Logic Colloquium 2024 [Nemoto 2024] A paper proof in analytic computability

$\Sigma_1 - \text{LEM} \implies \exists S \, . \, S$ is a low simple predicate

We mechanise this result in synthetic computability

CCC 2024

[Zeng, Forster, Kirst, and Nemoto 2024]

Post's Problem in Constructive Mathematics

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

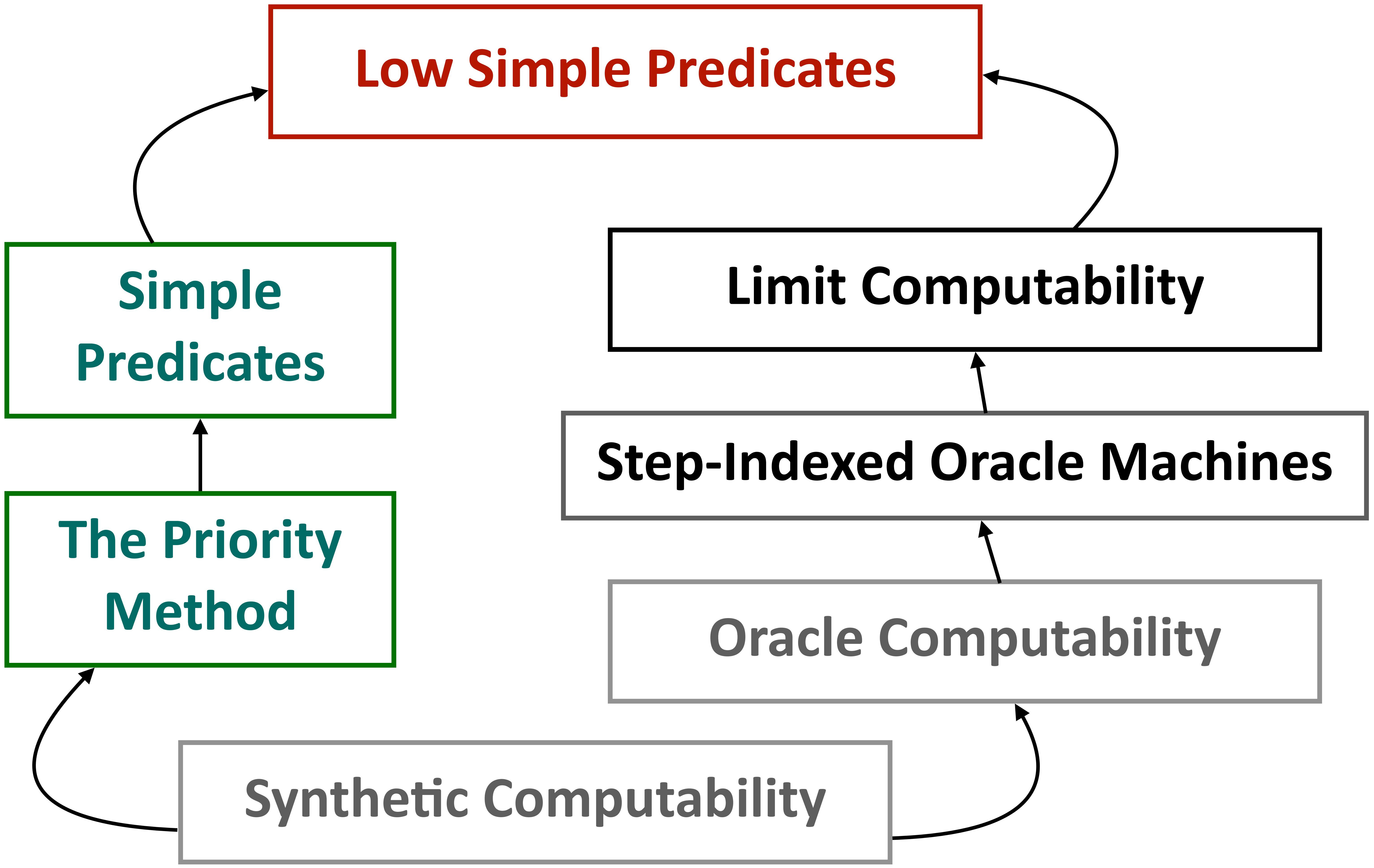
[1] Saarland University, Germany
[2] Inria Paris, France
[3] Tohoku University, Japan

Formalisation in Coq

Lines of code: ~ 1200

Lines of code: ~ 1000

Lines of code: ~ 400



Contribution

This thesis makes the following contributions:

- A definition of synthetic step-indexed oracle machines and use functions
- A synthetic notion of limit computability and the limit lemma
- The first synthetic and mechanised solution to Post's problem

Future Work

- Friedberg-Muchnik Theorem
- Low Basis Theorem
- Use of $\Sigma_1 - \mathrm{LEM}$
- ...

References

[Forster, Kirst & Mück 2023] Yannick Forster, Dominik Kirst and Niklas Mück. Oracle computability and Turing reducibility in the calculus of inductive constructions. *APLAS 2023*

[Post 1944] Post, E. L. Recursively enumerable sets of positive integers and their decision problems.

[Forster & Jahn 2023] Yannick Forster and Felix Jahn. Constructive and Synthetic Reducibility Degrees: Post's Problem for Many-one and Truth-table Reducibility in Coq. CSL 2023

References

[Friedberg 1957] 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 of the United States of America. Vol. 43, no. 2. pp. 236–238.

[Lerman & Soare 1980] LERMAN, M., AND SOARE, R. d-simple sets, small sets, and degree classes. Pacific Journal of Mathematics 87, 1 (1980), 135–155.

[Soare 1999] SOARE, R. Recursively enumerable sets and degrees: A study of computable functions and computably generated sets. Springer Science & Business Media, 1999.

References

[Mučnik 1956] Mučnik Albert Abramovich On the unsolvability of the problem of reducibility in the theory of algorithms. Doklady Akademii Nauk SSSR. 108: 194–197.

[Forster, Kirst & Mück 2024] Yannick Forster, Dominik Kirst and Niklas Mück. The kleene-post and post’s theorem in the calculus of inductive constructions CSL

[Shoenfield 1959] Shoenfield, J. R. On degrees of unsolvability. Annals of mathematics 69, 3 (1959), 644–653.

[Gold 1965] Gold, E. M. Limiting recursion. The Journal of Symbolic Logic 30, 1 (1965),28–48.

Appendix A

Oracle Computable

Based on a notion of computability of functionals $F : (Q \to A \to \mathbb{P}) \to (I \to Q \to \mathbb{P})$, The argument $R : Q \to A \to \mathbb{P}$ is to be read as the oracle relating questions $q : Q$ to answers $a : A$, $i : I$ is the input to the computation, and $o : O$ is the output, such an F is considered (oracle)-computable if there is an underlying computation tree $\tau : I \to A^* \rightharpoonup (Q + O)$:

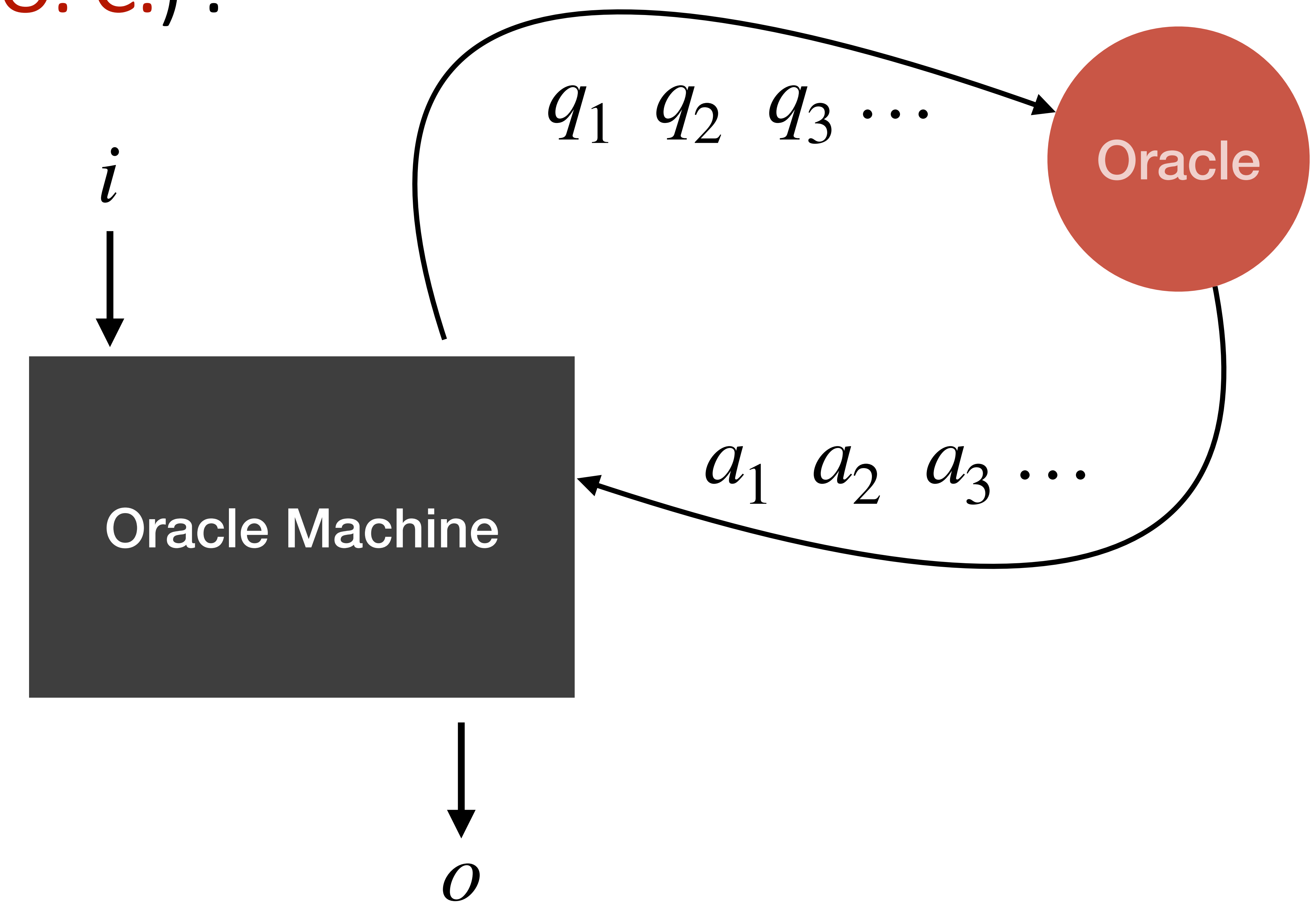
$$\forall R\ x\ b\,.\,F\ R\ x\ b \iff \exists qs\ as\,.\ \tau\ x; R \vdash qs; as \wedge \tau\ x\ as \triangleright\ \mathrm{out}\ b$$

where the interrogation relation $\sigma; R \vdash qs; as$ is inductively defined for $\sigma : A^* \rightharpoonup Q + O$ as:

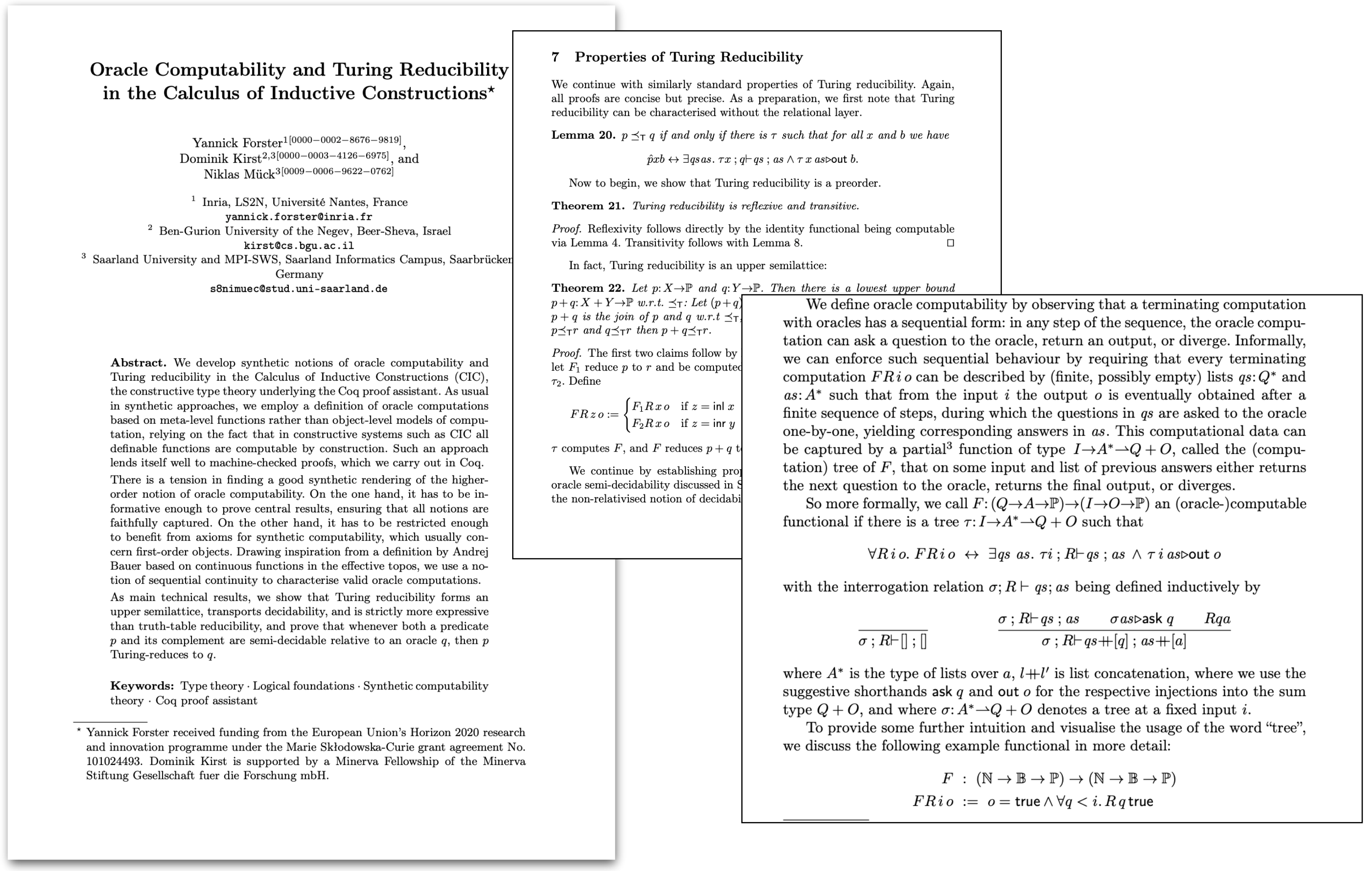
$$\frac{}{\sigma\ ; R \vdash [\,]; [\,]} \qquad \frac{\sigma\ ; R \vdash qs; as \quad \sigma\ as \triangleright \mathrm{ask}\ q \quad R(q, a)}{\sigma\ ; R \vdash qs @ [q]; as @ [a]}$$

Turing reducible in synthetic computability

Synthetic notation of Oracle Computable (O. C.) :



F is O.C. is capturing by some underlying computable object [Forster, Kirst & Mück 2023]



Oracle Computability and Turing Reducibility in the Calculus of Inductive Constructions*

Yannick Forster[1][0000−0002−8676−9819], Dominik Kirst[2,3][0000−0003−4126−6975], and Niklas Mück[3][0009−0006−9622−0762]

[1] Inria, LS2N, Université Nantes, France
yannick.forster@inria.fr
[2] Ben-Gurion University of the Negev, Beer-Sheva, Israel
kirst@cs.bgu.ac.il
[3] Saarland University and MPI-SWS, Saarland Informatics Campus, Saarbrücken Germany
s8nimuec@stud.uni-saarland.de

Abstract. We develop synthetic notions of oracle computability and Turing reducibility in the Calculus of Inductive Constructions (CIC), the constructive type theory underlying the Coq proof assistant. As usual in synthetic approaches, we employ a definition of oracle computations based on meta-level functions rather than object-level models of computation, relying on the fact that in constructive systems such as CIC all definable functions are computable by construction. Such an approach lends itself well to machine-checked proofs, which we carry out in Coq.
There is a tension in finding a good synthetic rendering of the higher-order notion of oracle computability. On the one hand, it has to be informative enough to prove central results, ensuring that all notions are faithfully captured. On the other hand, it has to be restricted enough to benefit from axioms for synthetic computability, which usually concern first-order objects. Drawing inspiration from a definition by Andrej Bauer based on continuous functions in the effective topos, we use a notion of sequential continuity to characterise valid oracle computations.
As main technical results, we show that Turing reducibility forms an upper semilattice, transports decidability, and is strictly more expressive than truth-table reducibility, and prove that whenever both a predicate p and its complement are semi-decidable relative to an oracle q, then p Turing-reduces to q.

Keywords: Type theory · Logical foundations · Synthetic computability theory · Coq proof assistant

* Yannick Forster received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 101024493. Dominik Kirst is supported by a Minerva Fellowship of the Minerva Stiftung Gesellschaft fuer die Forschung mbH.

7 Properties of Turing Reducibility

We continue with similarly standard properties of Turing reducibility. Again, all proofs are concise but precise. As a preparation, we first note that Turing reducibility can be characterised without the relational layer.

Lemma 20. *$p \preceq_T q$ if and only if there is τ such that for all x and b we have*

$$\hat{p}xb \leftrightarrow \exists qs\, as.\ \tau x\,;\, q \vdash qs\,;\, as \land \tau\, x\, as \rhd \mathsf{out}\ b.$$

Now to begin, we show that Turing reducibility is a preorder.

Theorem 21. *Turing reducibility is reflexive and transitive.*

Proof. Reflexivity follows directly by the identity functional being computable via Lemma 4. Transitivity follows with Lemma 8. □

In fact, Turing reducibility is an upper semilattice:

Theorem 22. *Let $p: X \to \mathbb{P}$ and $q: Y \to \mathbb{P}$. Then there is a lowest upper bound $p+q: X+Y \to \mathbb{P}$ w.r.t. $\preceq_T$: Let $(p+q)$ … $p+q$ is the join of p and q w.r.t $\preceq_T$ … $p \preceq_T r$ and $q \preceq_T r$ then $p+q \preceq_T r$.*

Proof. The first two claims follow by … let F_1 reduce p to r and be computed … τ_2. Define

$$F R z o := \begin{cases} F_1 R x o & \text{if } z = \mathsf{inl}\ x \\ F_2 R x o & \text{if } z = \mathsf{inr}\ y \end{cases}$$

τ computes F, and F reduces $p+q$ t…

We continue by establishing pro… oracle semi-decidability discussed in S… the non-relativised notion of decidabi…

We define oracle computability by observing that a terminating computation with oracles has a sequential form: in any step of the sequence, the oracle computation can ask a question to the oracle, return an output, or diverge. Informally, we can enforce such sequential behaviour by requiring that every terminating computation $F R i o$ can be described by (finite, possibly empty) lists $qs: Q^*$ and $as: A^*$ such that from the input i the output o is eventually obtained after a finite sequence of steps, during which the questions in qs are asked to the oracle one-by-one, yielding corresponding answers in as. This computational data can be captured by a partial[3] function of type $I \to A^* \rightharpoonup Q + O$, called the (computation) tree of F, that on some input and list of previous answers either returns the next question to the oracle, returns the final output, or diverges.

So more formally, we call $F: (Q \to A \to \mathbb{P}) \to (I \to O \to \mathbb{P})$ an (oracle-)computable functional if there is a tree $\tau: I \to A^* \rightharpoonup Q + O$ such that

$$\forall R\, i\, o.\ F R i o \leftrightarrow \exists qs\ as.\ \tau i\,;\, R \vdash qs\,;\, as \land \tau\, i\, as \rhd \mathsf{out}\ o$$

with the interrogation relation $\sigma; R \vdash qs; as$ being defined inductively by

$$\frac{}{\sigma\,;\, R \vdash [\,]\,;\,[\,]} \qquad \frac{\sigma\,;\, R \vdash qs\,;\, as \quad \sigma\, as \rhd \mathsf{ask}\ q \quad R q a}{\sigma\,;\, R \vdash qs +\!\!+ [q]\,;\, as +\!\!+ [a]}$$

where A^* is the type of lists over a, $l +\!\!+ l'$ is list concatenation, where we use the suggestive shorthands $\mathsf{ask}\ q$ and $\mathsf{out}\ o$ for the respective injections into the sum type $Q + O$, and where $\sigma: A^* \rightharpoonup Q + O$ denotes a tree at a fixed input i.

To provide some further intuition and visualise the usage of the word "tree", we discuss the following example functional in more detail:

$$F : (\mathbb{N} \to \mathbb{B} \to \mathbb{P}) \to (\mathbb{N} \to \mathbb{B} \to \mathbb{P})$$
$$F R i o := o = \mathsf{true} \land \forall q < i.\ R\, q\, \mathsf{true}$$

Turing reduction $P \leq Q := \exists F.\ F$ is O.C. $\land\ \forall x.\ \begin{array}{l} P\,x \leftrightarrow F\,\hat{Q}\,x\ \mathsf{tt} \\ \neg P\,x \leftrightarrow F\,\hat{Q}\,x\ \mathsf{ff} \end{array}$

Appendix B

Step index function

We insert this oracle O into our Turing machine by fixing a n, and subsequently run τ. Based on this effectively computable oracle, we can define a total function Φ as follows:

$$\Phi_{\tau}^{O(n)}\ x\ i\ j \ := \begin{cases} \ulcorner \text{out } o \urcorner & \text{if } (\tau\ x\ []) \rightsquigarrow_j \text{out } o \\ \ulcorner \text{ask } q \urcorner & \text{if } (\tau\ x\ []) \rightsquigarrow_j \text{ask } q \text{ and } \ i = 0 \\ \Phi_{\tau @ [\chi_O\ n\ q]}^{O(n)}\ x\ i'\ j & \text{if } (\tau\ x\ []) \rightsquigarrow_j \text{ask } q \text{ and } \ i = S\ i' \\ \text{none} & \text{otherwise} \end{cases}$$

Given that P is Turing reducible to $\varnothing$, we obtain the computable tree τ. Building upon the step-index function described above, we define the following function:

$$\chi_P(s, x) \ := \begin{cases} b & \text{if } \Phi_{\tau}^{K}(x)[s] = \ulcorner b \urcorner \\ \text{tt} & \text{otherwise} \end{cases}$$

Low Simple Predicate 1

undecidable predicate

Simple predicate:

If a predicate P is simple, then P is semi-decidable and $\neg(P \preceq \emptyset)$

Turing jump of P:

$P' \ x \leftrightarrow x$-th oracle machine with oracle P halts on x

Low Simple Predicate 2

Low predicate: A predicate P is low, if the Turing jump of P is reducible to K

$$P' \preceq K \Rightarrow \neg(K \preceq P)$$

Low Simple predicate: $\varnothing \prec P \prec K$, where $P \prec Q := P \preceq Q \wedge \neg(P \preceq Q)$

A positive solution to Post's Problem

Showing a predicate is reducible to K is difficult!

Use Function

Let $k = \varphi_e^A(e)[n]$

$$\Phi_e^p(e)[n] = \ulcorner\star\urcorner \to \forall q.\ q \equiv_k p[n] \to \Xi_e\ \hat{q}\ e\ \star$$

$$\varphi_e^p(x)[n] = k \to p[n] \equiv_k p[n+1] \to \varphi_e^p(x)[n+1] = k$$