The Journal of Symbolic Logic
Volume 89, Number 1, March 2024

THE FIRST-ORDER LOGIC OF CZF IS INTUITIONISTIC FIRST-ORDER LOGIC

ROBERT PASSMANN

Abstract. We prove that the first-order logic of CZF is intuitionistic first-order logic. To do so, we introduce a new model of transfinite computation (Set Register Machines) and combine the resulting notion of realisability with Beth semantics. On the way, we also show that the propositional admissible rules of CZF are exactly those of intuitionistic propositional logic.

§1. Introduction. The *first-order logic of a theory* T consists of those first-order formulas for which all substitution instances are provable in T. A classical result of Friedman and Ščedrov [7] is that very few axioms suffice for a set theory to exceed the logical strength of intuitionistic first-order logic:

Theorem (Friedman and Ščedrov (1986)). *Let T be a set theory based on intuitionistic first-order logic that contains the axioms of extensionality, pairing and (finite) union, as well as the separation schema. Then the first-order logic of T exceeds the strength of intuitionistic first-order logic.*

This result applies to intuitionistic Zermelo–Fraenkel Set Theory (IZF) but not to constructive Zermelo–Fraenkel set theory (CZF) because the separation schema of CZF is restricted to Δ_0-formulas. It has, thus, been a long-standing open question whether the first-order logic of CZF exceeds the strength of intuitionistic logic as well. We give an answer to this question:

Theorem (see Corollary 5.15). *The first-order logic of* CZF *is intuitionistic first-order logic.*

We prove this result by developing a realisability semantics for CZF based on a new model of transfinite computation, the so-called *Set Register Machines* (SRMs). Related notions of realisability had earlier been studied by Rathjen [22] and Tharp [23]. Our main result is obtained by adapting a technique that van Oosten [19] had developed for Heyting arithmetic: we combine the resulting notion of SRM-realisability with Beth semantics to obtain a model of CZF that matches logical truth in a universal Beth model.

Carl, Galeotti, and Passmann [4] gave a first proof-theoretic application of transfinite computability and provided a realisability interpretation for (infinitary) IKP set theory using OTMs. In particular, they proved that the propositional

Received December 2, 2021.
2020 *Mathematics Subject Classification.* Primary 03F50, Secondary 03B20, 03B55, 03E70, 03F55.
Key words and phrases. intuitionistic set theory, constructive set theory, first-order logic, tautologies of a theory, admissible rules.

0022-4812/24/8901-0015
DOI:10.1017/jsl.2022.51

admissible rules of IKP are exactly the admissible rules of intuitionistic propositional logic. On the way to proving our main result, we will prove the same result for CZF. Our motivation for introducing SRMs instead of working with OTMs is that the former are easier adapted for realising stronger set theories than IKP. This work is thus another fruitful application of techniques of transfinite computability to proof-theoretic questions.

1.1. Overview. After recalling some preliminaries in Section 2, we will begin, in Section 3, with introducing our new notion of transfinite machines, the so-called *set register machines* (SRMs). The main result of this section will be a generalisation of a classical result by Kleene and Post about the existence of mutually irreducible degrees of computability. In Section 4, we introduce a realisability semantics based on SRMs and show that (a certain extension of) these machines allows to realise CZF set theory. It also serves as a preparation for Section 5, in which we will combine our realisability semantics with Beth models to prove our main result.

§2. Preliminaries.

2.1. Constructive set theory. We will be concerned with constructive Zermelo–Fraenkel set theory, CZF, and now recall its definition and some basic facts. First, recall the axiom schemes of *strong collection*,

$$\forall a[\forall x \in a \exists y \varphi(x,y) \to \exists b(\forall x \in a \exists y \in b \varphi(x,y) \wedge \forall y \in b \exists x \in a \varphi(x,y))],$$

for all formulas φ, in which b is not free, and *subset collection*,

$$\forall a \forall b \exists c \forall u[\forall x \in a \exists y \in b \varphi(x,y,u) \to \exists d \in c(\forall x \in a \exists y \in d \varphi(x,y,u) \wedge \forall y \in d \exists x \in a \varphi(x,y,u))],$$

for all formulas φ, in which c is not free. By Δ_0-separation we denote the restriction of the separation schema to Δ_0-formulas.

Definition 2.1. Constructive Zermelo–Fraenkel Set Theory, CZF, is based on intuitionistic first-order logic in the language of set theory and consists of the following axioms and axiom schemes: extensionality, pairing, union, empty set, infinity, Δ_0-separation, strong collection, subset collection, and $\in$-induction.

We denote CZF without the subset collection schema by CZF^-. The exponentiation axiom states that function sets exists:

$$\forall a \forall b \exists c \forall f (f \in c \leftrightarrow \text{“}f \text{ is a function from } a \text{ to } b\text{”}).$$

The following is well known (consult, e.g., Aczel and Rathjen [1]).

Fact 2.2. *In* CZF^-, *the power set axiom implies the subset collection axiom. Moreover, in* CZF^-, *the subset collection scheme implies the exponentiation axiom.*

2.2. Logics and De Jongh's theorem. Given a theory T, based on intuitionistic logic, the logically valid principles of T may exceed those valid in intuitionistic logic. The most well-known example of this phenomenon is probably the following consequence of what is known as *Diaconescu's theorem* (see Diaconescu [5] and Goodman and Myhill [8]): IZF extended with the axiom of choice implies the law of excluded middle, i.e., $\mathrm{IZF} + \mathrm{AC} \vdash \varphi \vee \neg\varphi$ for all set-theoretic formulas φ. This

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

suggests that it is incorrect to say that the logic of IZF + AC is intuitionistic: after all, the law of excluded middle is valid! For this reason, we define the propositional and first-order logics of a theory T as follows, in terms of translations.

DEFINITION 2.3. Let T be a theory in a language $\mathcal{L}_T$. A *propositional translation* is a function τ assigning $\mathcal{L}_T$-sentences to propositional formulas such that:

(i) $\tau(p)$ is an $\mathcal{L}_T$-sentence for every propositional letter p,
(ii) $\tau(\bot) = \bot$, and
(iii) $\tau(A \circ B) = \tau(A) \circ \tau(B)$ for $\circ \in \{\wedge, \vee, \to\}$.

As customary with translations, we will often write A^τ instead of $\tau(A)$.

DEFINITION 2.4. The *propositional logic of* T, $\mathbf{PL}(T)$, consists of all propositional formulas A such that $T \vdash A^\tau$ for all propositional translations τ.

A result concerning the first-order logic of Heyting arithmetic was proved by de Jongh in his doctoral dissertation [14, 15]. We denote intuitionistic propositional logic by **IPC** and intuitionistic first-order logic by **IQC**.

THEOREM 2.5 (de Jongh (1970)). *The propositional logic of Heyting arithmetic is intuitionistic propositional logic*, $\mathbf{PL}(\mathrm{HA}) = \mathbf{IPC}$.

This result is now known as *de Jongh's theorem*, and, in general, we say that a theory T *satisfies de Jongh's theorem* whenever $\mathbf{PL}(T) = \mathbf{IPC}$.

DEFINITION 2.6. Let T be a theory in a language $\mathcal{L}_T$. A *first-order translation* is a function τ assigning $\mathcal{L}_T$-formulas to propositional formulas such that:

(i) $\tau(R(x_1, \dots, x_n))$ is an $\mathcal{L}_T$-formula φ with free variables among $x_1, \dots, x_n$,
(ii) $\tau(\bot) = \bot$,
(iii) $\tau(A \circ B) = \tau(A) \circ \tau(B)$ for $\circ \in \{\wedge, \vee, \to\}$, and
(iv) $\tau(\mathcal{Q}xA(x)) = \mathcal{Q}x\tau(A(x))$ for $\mathcal{Q} \in \{\forall, \exists\}$.

DEFINITION 2.7. The *first-order logic of* T, $\mathbf{QL}(T)$, consists of all first-order formulas A such that $T \vdash A^\tau$ for all first-order translations τ.

Since de Jongh's initial work, many notable results have been obtained in this area. Leivant [18] showed that $\mathbf{QL}(\mathrm{HA}) = \mathbf{IQC}$; van Oosten [19] gave a semantic proof of this fact (the idea of his construction will reappear in our construction in Section 5). De Jongh, Verbrugge, and Visser [16] consider a generalised version of de Jongh's theorem: given a (propositional or first-order) logic J and a theory T, we can consider the theory $T(J)$ obtained by closing T under J. We then say that T satisfies the *de Jongh property for* J if $\mathbf{PL}(T(J)) = J$ (or, $\mathbf{QL}(T(J)) = J$ if J is a first-order logic).

The main negative result concerning logics of set theory is due to Friedman and Ščedrov [7], and was also mentioned in the introduction. Here is a reformulation based on the terminology just introduced.

THEOREM 2.8 (Friedman and Ščedrov (1986)). *Let T be a set theory based on intuitionistic first-order logic that contains the axioms of extensionality, pairing and (finite) union, as well as the separation scheme. Then*, $\mathbf{IQC} \subsetneq \mathbf{QL}(T)$.

Passmann [20] showed that $\mathbf{PL}(\mathrm{IZF}) = \mathbf{IPC}$, and consequently, $\mathbf{PL}(\mathrm{CZF}) = \mathbf{IPC}$. Iemhoff and Passmann [11] analysed the logical structure of IKP and proved, among other things, that $\mathbf{QL}(\mathrm{IKP}) = \mathbf{IQC}$.

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

2.3. Admissible rules. We can further generalise our analysis of the logical structure of a given theory by not only considering its logically valid principles but also by determining its admissible rules.

DEFINITION 2.9. Let T be a theory in a language $\mathcal{L}_T$, and let A and B be propositional formulas. We say that a propositional rule A/B is *admissible in* T, written $A \mathrel{\sim_T} B$, if and only if $T \vdash A^\tau$ implies $T \vdash B^\tau$ for all propositional translations τ.

We say that a theory T has the disjunction property if $T \vdash \varphi \vee \psi$ implies $T \vdash \varphi$ or $T \vdash \psi$. The *restricted Visser's rules* $\{V_n\}_{n<\omega}$ are defined as follows and play a special role for admissibility (Iemhoff [9] proved that they form a so-called basis of the admissible rules of intuitionistic propositional logic):

$$\frac{\left(\bigwedge_{i=1}^{n}(p_i \to q_i)\right) \to (p_{n+1} \vee p_{n+2})}{\bigvee_{j=1}^{n+2}\left(\bigwedge_{i=1}^{n}(p_i \to q_i) \to p_j\right)}.$$

Denote by V_n^a the antecedent and by V_n^c the consequent of the rule. We will make use of the following result of Iemhoff [10] to determine admissible rules.

THEOREM 2.10 (Iemhoff [10, Theorem 3.9 and Corollary 3.10]). *Let T be a theory with the disjunction property. If the restricted Visser's rules are propositional admissible in T, then the propositional admissible rules of T are exactly the propositional admissible rules of intuitionistic propositional logic*, $\mathrel{\sim_T} = \mathrel{\sim_{\mathbf{IPC}}}$.

Visser [25] proved that the propositional admissible rules of Heyting Arithmetic HA are exactly the admissible rules of intuitionistic propositional logic **IPC**. Using realisability techniques, Carl, Galeotti, and Passmann [4] determined the propositional admissible rules of IKP to be exactly the admissible rules of propositional intuitionistic logic. Iemhoff and Passmann [12] proved that the propositional admissible rules of $\mathrm{CZF_{ER}}$ and $\mathrm{IZF_R}$ are the admissible rules of intuitionistic propositional logic by using a modification of the so-called blended models (earlier introduced by Passmann [20]).[1] It is possible to consider first-order admissible rules; van den Berg and Moerdijk [2] show that certain constructive principles are first-order admissible rules of CZF (calling them *derived* rules).

§3. Set register machines.

3.1. Definitions and basic properties. Let us begin with some intuition for set register machines (SRMs). A set register machine has a finite set of registers $R_0, \ldots, R_n$ on which it conducts computations. However, the registers do not contain natural numbers (as in the case of register machines) or ordinal numbers (as in the case of ordinal register or Turing machines) but rather arbitrary sets. Accordingly, SRMs use a different set of operations: for example, adding a set contained in a

[1] To obtain $\mathrm{CZF_{ER}}$ and $\mathrm{IZF_R}$, replace subset collection and (strong) collection by exponentiation and replacement, respectively.

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

register to another register, or removing a member of a set contained in a certain register.

We assume that $<_\tau$ is a global well-ordering such that $\mathrm{rank}(x) < \mathrm{rank}(y)$ implies $x <_\tau y$.[2] This means that we are working under the assumption of the global axiom of choice and extend our set-theoretical language with the symbol $<_\tau$. Note that this extended theory is conservative over ZFC (see Fraenkel [6, pp. 72–73]). The reason for using this theory as our meta-theory is that we want SRM-computations to be deterministic, and assuming a global well-ordering is a convenient way to achieve this. For a discussion of alternatives see Remark 3.3.

We will now first define programs by giving the permissible operations, and then computations for set register machines. While defining the permissible operations, we will directly give an intuitive description of what the operation does.

DEFINITION 3.1. A *set register program* p is a finite sequence $p = (p_0, \dots, p_{n-1})$, where each p_i is one of the following commands:

(i) "$R_i := \emptyset$": replace the content of the ith register with the empty set.
(ii) "$\mathtt{ADD}(i, j)$": replace the content of the jth register with $R_j \cup \{R_i\}$.
(iii) "$\mathtt{COPY}(i, j)$": replace the content of the jth register with R_i.
(iv) "$\mathtt{TAKE}(i, j)$": replace the content of the jth register with the $<_\tau$-least set contained in R_i, if R_i is non-empty.
(v) "$\mathtt{REMOVE}(i, j)$": replace the content of the jth register with the set $R_j \setminus \{R_i\}$.
(vi) "$\mathtt{IF}\ R_i = \emptyset\ \mathtt{THEN\ GO\ TO}\ k$": check whether the ith register is empty; if so, move to program line k, and, if not, move to the next line.
(vii) "$\mathtt{IF}\ R_i \in R_j\ \mathtt{THEN\ GO\ TO}\ k$": check whether $R_i \in R_j$; if so, move to program line k, and, if not, move to the next line.
(viii) "$\mathtt{POW}(i, j)$": replace the content of the jth register with the power set of R_i.

DEFINITION 3.2. Let p be a set register program and $k < \omega$ be the highest register index appearing in p. A *configuration of* p is a sequence $(l, r_0, \dots, r_k)$ consisting of the active program line $l < \omega$ and the current content r_i of register R_i. If $c = (l, r_0, \dots, r_k)$ is a configuration of p, then its successor configuration $c^+ = (l^+, r_0^+, \dots, r_k^+)$ is obtained as follows:

(i) If p_l is "$R_i := \emptyset$", then let $r_i^+ = \emptyset$, $r_n^+ = r_n$ for $n \neq i$, and $l^+ = l + 1$.
(ii) If p_l is "$\mathtt{ADD}(i, j)$", then let $r_j^+ = r_j \cup \{r_i\}$, $r_n^+ = r_n$ for $n \neq j$, and $l^+ = l + 1$.
(iii) If p_l is "$\mathtt{COPY}(i, j)$", then let $r_j^+ = r_i$, $r_n^+ = r_n$ for $n \neq j$, and $l^+ = l + 1$.
(iv) If p_l is "$\mathtt{TAKE}(i, j)$", then let r_j^+ be the $<_\tau$-minimal element of r_i (if that exists; if $r_i = \emptyset$, then $r_j^+ = r_j$), $r_n^+ = r_n$ for $n \neq j$, and $l^+ = l + 1$.
(v) If p_l is "$\mathtt{REMOVE}(i, j)$", then let $r_j^+ = r_j \setminus \{r_i\}$, $r_n^+ = r_n$ for $n \neq j$, and $l^+ = l + 1$.
(vi) If p_l is "$\mathtt{IF}\ R_i = \emptyset\ \mathtt{THEN\ GO\ TO}\ m$", then $r_i^+ = r_i$ for all $i \leq k$; and, if $r_i = \emptyset$, then $l^+ = m$; if $r_i \neq \emptyset$, then $l^+ = l + 1$.
(vii) If p_l is "$\mathtt{IF}\ R_i \in R_j\ \mathtt{THEN\ GO\ TO}\ m$", then $r_i^+ = r_i$ for all $i \leq k$; and, if $r_i \in r_j$, then $l^+ = m$; if $r_i \notin r_j$, then $l^+ = l + 1$.

[2] Whenever $<_\tau$ is a global well-ordering, we can assume that this is the case by defining $x <'_\tau y$ if and only if $\mathrm{rank}(x) < \mathrm{rank}(y)$ or $\mathrm{rank}(x) = \mathrm{rank}(y)$ and $x <_\tau y$. Note that $<'_\tau$ is again a well-order.

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

(viii) If p_l is "POW(i, j)", then $r_j^+ = \mathcal{P}(r_i)$, $r_n^+ = r_n$ for all $n \neq i$, and $l^+ = l + 1$.

A *computation of p with input $x_0, \dots, x_j$* is a sequence d of ordinal length $\alpha + 1$ consisting of configurations of p such that:

(i) $d_0 = (1, x_0, \dots, x_j, \emptyset, \dots, \emptyset)$,
(ii) if $\beta < \alpha$, then $d_{\beta+1} = d_\beta^+$,
(iii) if $\beta < \alpha$ is a limit, then $l_\beta = \liminf_{\gamma<\beta} l_\gamma$, and $r_\beta = \liminf_{\gamma<\beta} r_\gamma$, where the limes inferior of a sequence of sets is the set obtained from the limes inferior of the characteristic functions, and
(iv) d_α^+ is undefined (i.e., $l_\alpha > m$).

The notion of computability obtained by restricting Definitions 3.1 and 3.2 to clauses (i)–(vii) will be referred to as SRM; the full notion will be referred to as SRM$^+$. In other words, SRM$^+$ is obtained from SRM by adding the power set operation. We allow SRMs and SRM$^+$s to make use of finitely many set parameters which will be treated as additional input in a fixed register as specified in the program code.

REMARK 3.3. There are several alternatives for working with a global well-ordering function $<_\tau$: first, it is possible to develop a theory of non-deterministic SRMs, where the TAKE-command takes an arbitrary set. Second, SRMs could work on well-ordered sets (i.e., sets equipped with a well-order). This approach is not useful for SRM$^+$ as there is no canonical way in extending the well-ordering of a set to its power set (i.e., a certain degree of non-determinateness is introduced again). A third approach is to make computations dependent on a large enough well-ordering of some initial V_α. Finally, one could work in the constructible universe L where we have a Σ_1-definable well-ordering $<_L$. We will, in fact, consider this approach in Section 3.3 but for different reasons: for our main application, we need computations to be definable in the language of set theory without an additional symbol for the global well-ordering.

DEFINITION 3.4. A function f is SRM$^{(+)}$-*computable* if there is an SRM$^{(+)}$-program p, possibly with parameters, which computes $f(x)$ on input x. A predicate is called SRM$^{(+)}$-computable if its characteristic function is SRM$^{(+)}$-computable.

Note that every function with set-sized domain is SRM-computable. Clearly, if a function or predicate is SRM-computable, then it is also SRM$^+$-computable. The converse does not hold: consider, for example, the power set operation.

PROPOSITION 3.5. *Equality of sets is* SRM-*computable.*

PROOF. The following SRM-program computes whether the sets contained in registers R_0 and R_1 are equal: the program successively takes elements of the first set, checks whether they are contained in the second set, and removes the element from both sets. If both registers R_0 and R_1 are empty at the same time, then the original sets must have been equal. Otherwise, the original sets were not equal.

1: IF $R_0 = \emptyset$ THEN GO TO 3
2: GO TO 5
3: IF $R_1 = \emptyset$ THEN GO TO 11

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

4: GO TO 14
5: TAKE$(0, 2)$
6: REMOVE$(2, 0)$
7: IF $R_2 \in R_1$ THEN GO TO 9
8: GO TO 14
9: REMOVE$(2, 1)$
10: GO TO 1
11: $R_0 := \emptyset$
12: ADD$(0, 0)$
13: GO TO 15
14: $R_0 := \emptyset$

Note that the operation "GO TO i" is a shortcut for "IF $R_j = \emptyset$ THEN GO TO i" where j is chosen in such a way that the register R_j is not mentioned in any other instruction of the program. ⊣

In view of this proposition, we can use an operation "IF $R_i = R_j$ THEN GO TO k" by implementing the program of the proof of the proposition as a subroutine. The following lemma shows that many basic operations and predicates are SRM^+-computable.

LEMMA 3.6. *The following functions and predicates are* SRM^+*-computable*:

(i) *the binary union function* $(x, y) \mapsto x \cup y$,
(ii) *the intersection function* $(x, y) \mapsto x \cap y$,
(iii) *the singleton and pairing functions,* $x \mapsto \{x\}$ *and* $(x, y) \mapsto \{x, y\}$,
(iv) *the ordered pairing function* $(x, y) \mapsto \langle x, y \rangle$,
(v) *the first and second projections* $\langle x, y \rangle \mapsto x$, $\langle x, y \rangle \mapsto y$,
(vi) *the predicate "x is an ordered pair,"*
(vii) *the predicate "x is a function,"*
(viii) *the union of a set,* $x \mapsto \bigcup x$,
(ix) *the intersection of a set,* $x \mapsto \bigcap x$,
(x) *the function mapping a function to its domain* $f \mapsto \mathrm{dom}(f)$,
(xi) *function application* $(f, x) \mapsto f(x)$,
(xii) *the predicate "x is an ordinal,"*
(xiii) *the predicate "x is a sequence of ordinal length,"*
(xiv) *the function computing the* $<_\tau$*-least element* $x \in y$ *satisfying an* SRM^+*-computable predicate* $P(x)$,
(xv) *the* α*th projection on a sequence,* $\langle x_i \mid i < \beta \rangle \mapsto x_\alpha$,
(xvi) *the power set function,* $x \mapsto \mathcal{P}(x)$,
(xvii) *the predicate "x is the power set of y,"*
(xviii) *the limes inferior of a sequence of sets.*

PROOF. We will give explicit programs for the first few cases and then move to increasingly abstract descriptions of the desired programs:

(i) Observe that the following program computes the union of the sets in registers R_0 and R_1 by adding all elements of R_1 to R_0:

1: IF $R_1 = \emptyset$ THEN GO TO 6
2: TAKE$(1, 2)$

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

3: REMOVE$(2, 1)$
4: ADD$(2, 0)$
5: GO TO 1

(ii) Observe that the intersection of the sets contained in registers R_0 and R_1 can be computed as follows. Check for each element of R_1 whether it is contained in R_0 and, if so, save it into a register for the intersection:

1: IF $R_1 = \emptyset$ THEN GO TO 8
2: TAKE$(1, 2)$
3: REMOVE$(2, 1)$
4: IF $R_2 \in R_0$ THEN GO TO 6
5: GO TO 1
6: ADD$(2, 3)$
7: GO TO 1
8: COPY$(3, 0)$

(iii) The functions of (iii) can be easily implemented.
(iv) Recall that $\langle x, y\rangle = \{\{x\}, \{x, y\}\}$, and this can easily be computed.
(v) Note that $\bigcap\langle x, y\rangle = x$ and $\bigcup\langle x, y\rangle = \{x, y\}$. So we can construct the desired programs by combining the procedures from (i) and (ii) in a straightforward way.
(vi) We have to implement a procedure that checks whether x is an ordered pair: use (v) to compute the first and second projections of x, say, y and z. Then compute $\langle y, z\rangle$ with (iv) and check whether this equals x.
(vii) Check whether x consists of ordered pairs (using (vi)), and then check that x is functional with (v).
(viii) Use four registers: R_0 contains x, R_1 for the union of x, and R_2 and R_3 as auxiliary registers. Then proceed as follows: as long as R_0 is non-empty, take a set from R_0 and save it in R_2, then remove it from R_0. Then, as long as R_2 is non-empty, take an element of R_2 and save it in R_3, then remove it from R_2 and add it to R_1. Once R_0 is empty, we are done: copy our result from R_1 to R_0, and stop.
(ix) A similar procedure as in the previous item does the job.
(x) Take and remove elements from R_0 as long as it is non-empty. To each element, apply the first-projection from (v), and add it to R_1. Once R_0 is empty, R_1 contains the domain of x.
(xi) Search through f until a pair with first coordinate x is found. Then return the second projection of that pair.
(xii) Observe that it is straightforward to compute whether "x is a transitive set of transitive sets."
(xiii) Check whether x is a function whose domain is an ordinal.
(xiv) Given a procedure for checking P, take and remove elements from y until some x is found satisfying $P(x)$. By the definition of the TAKE-operation, this will be the $<_\tau$-minimal element of y satisfying P.
(xv) This is just function application.
(xvi) This is straightforward using the POW-operation.
(xvii) Again, straightforward using the POW-operation.

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

(xviii) Note that the limes inferior of a sequence of sets can be presented as follows:

$$\liminf_{\gamma<\alpha} x_\gamma = \bigcup_{\beta<\alpha} \bigcap_{\gamma\in[\beta+1,\alpha)} x_\gamma.$$

This can be straightforwardly implemented by combining the previous items of this lemma. ⊣

LEMMA 3.7. *Let $\varphi(\bar{x})$ be a Δ_0-formula. Then there is an* SRM *p such that $p(\ulcorner\varphi\urcorner, \bar{x}) = 1$ if $\mathrm{V} \vDash \varphi(\bar{x})$ and $p(\ulcorner\varphi\urcorner, \bar{x}) = 0$ if $\mathrm{V} \vDash \neg\varphi(\bar{x})$.*

PROOF. We construct a machine that recursively calls itself. For the base cases, let $p(\ulcorner x_i = x_j\urcorner, \bar{x})$ be the program that returns 1 if $x_i = x_j$ and 0 if $x_i \neq x_j$. Similarly, let $p(\ulcorner x_i \in x_j\urcorner, \bar{x})$ be the program that returns 1 if $x_i \in x_j$ and 0 if $x_i \notin x_j$. The cases for conjunction, disjunction, and implication are easily constructed by recursion. For the bounded existential quantifier, $\exists x \in a\, \varphi(x)$, the machine p conducts a search through a by consecutively taking and removing elements. If p finds some $b \in a$ such that $p(\ulcorner\varphi\urcorner, \langle b, a, x\rangle) = 1$, then p returns 1. If no such b is found, then a does not contain a witness for φ and p returns 0. The bounded universal quantifier can be implemented similarly with a bounded search. ⊣

The next theorem shows that moving from Ordinal Turing Machines to Set Register Machines does not increase the computational strength. We do not give a detailed proof since the result is not used in the remainder of this article.

THEOREM 3.8. *Ordinal Turing machines with parameters (OTMs) and set register machines with parameters (SRMs) can simulate each other.*

PROOF. For the first direction, recall that OTMs and ordinal register machines (ORMs) can simulate each other (e.g., Carl [3]). It will, therefore, be enough to show that SRMs simulate ORMs but, in fact, more is true: it is straightforward to see that every ORM-program can be executed by an SRM.

The other direction can be shown by a straightforward but tedious coding argument by using a large enough fragment of the well-order $<_\tau$ as a parameter (Carl, Galeotti, and Passmann [4] spell out a very similar argument in an appendix; Carl [3, Section 2.3.2 and Chapter 3] discusses codings as well). ⊣

3.2. Oracles and relative computability. As with other notions of computability, we can enrich SRM$^+$s with oracles. Let $O : \mathrm{V} \to \mathrm{V}$ be a partial class function. We obtain oracle SRM$^{+,O}$ by extending Definition 3.1 with the following operation:

> "ORACLE(i, j)": replace the contents of the jth register with the result of querying the oracle O with R_i.

We also extend Definition 3.2:

> If p_l is "ORACLE(i, j)," proceed as follows: if $O(r_i)$ is defined, let $r_j^+ = O(r_i)$, $r_n^+ = r_n$ for all $n \neq i$, and $l^+ = l + 1$. If $O(r_i)$ is undefined, let $r_j^+ = r_j$ for all $j \leq k$ and $l^+ = l$.

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

The evaluation function is chosen like this to ensure that any $\mathrm{SRM}^{+,O}$ loops whenever the oracle is queried on undefined input. This entails that the oracle is only queried on its domain within a successful computation. Given oracles, we can define a relative notion of computability.

DEFINITION 3.9. We say that a function f is SRM^{+}-*computable in* g if and only if there is an $\mathrm{SRM}^{+,g}$ program p that computes f.

A function is SRM^{+}-*computable* if and only if it is SRM^{+}-computable in the empty function. In fact, a function is SRM^{+}-*computable* if and only if it is SRM^{+}-computable in any set-sized function.

We will now work towards generalising a result of Kleene and Post [17], which will be useful later but is also interesting in its own regard.

PROPOSITION 3.10. *The class function* $V_{(\cdot)} : \mathrm{Ord} \to V, \alpha \mapsto V_\alpha$ *is* SRM^{+}-*computable.*

PROOF. An SRM^{+}-program does this by starting with the empty set and consecutively computing power sets while keeping the current rank in an auxiliary register. The program keeps computing until it reaches the desired α.

This procedure is implemented in the following program, where the input α is written into R_0; note that the initial configuration of all other registers is $\emptyset$. We use R_1 to count our current stage β and R_2 to save the current V_β.

1: IF $R_0 = R_1$ THEN GO TO 5
2: POW$(2, 2)$
3: ADD$(1, 1)$
4: GO TO 1

Note that the register R_0 remains unchanged, and the registers R_1 and R_2 are monotonically increasing. Therefore, the program does the job also at limit stages. ⊣

The following proposition can be anticipated from how the evaluation of the TAKE-operation was defined.

PROPOSITION 3.11. *The global well-ordering* $<_\tau$ *is* SRM^{+}-*decidable.*

PROOF. This is implemented by an SRM^{+} that does the following: given a and b, check whether $a = b$. If so, we are done. If not, compute $\{a, b\}$ and use the TAKE-operation to take a set $c \in \{a, b\}$. By the definition of the TAKE-operation, either $c = a$ and then $a <_\tau b$, or $c = b$ and then $b <_\tau a$. ⊣

By the α*th element of* V *according to* $<_\tau$, we denote the unique x such that the order type of $(\{y \mid y <_\tau x\}, <_\tau)$ is α.

PROPOSITION 3.12. *The bijective class function* $F_\tau : \mathrm{Ord} \to V$ *mapping* α *to the* α*th element of* V *according to* $<_\tau$ *is* SRM^{+}-*computable and so is its inverse.*

PROOF. Recall our assumption that $\mathrm{rank}(x) < \mathrm{rank}(y)$ implies $x <_\tau y$. Therefore, computing $<_\tau$ on some V_α means to compute an initial segment of $<_\tau$. We can therefore proceed as follows.

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

For the forward direction, use the POW-operation to compute $V_{\alpha+1}$. Then take and remove elements from $V_{\alpha+1}$ while running a counter until it reaches α. The last element taken is the set we were looking for.

For the other direction, given $a \in V$, compute a V_β such that $a \in V_\beta$. Then start a counter and successively take and remove elements from V_β until a is reached. The value of the counter is the ordinal α we are looking for. ⊣

PROPOSITION 3.13. *Let O be a (partial) class function. The $\mathrm{SRM}^{+,O}$ halting problem is $\mathrm{SRM}^{+,O}$ undecidable.*

PROOF. This is proved by contradiction with the usual diagonal argument. Assume that there is a machine p such that $p(x) = 1$ if and only if x is an SRM^+ that halts, and $p(x) = 0$ otherwise. Then define a machine q such that $q(x)$ does not halt if and only if $p(x) = 1$. Then, $p(q) = 1$ if and only if $q(q)$ does not halt if and only if $p(q) = 0$. A contradiction. ⊣

PROPOSITION 3.14. *Let O be a (partial) class function. Then there is an oracle $\tilde{O}$ such that there is an $\mathrm{SRM}^{+,\tilde{O}}$-program u which is universal for $\mathrm{SRM}^{+,O}$, i.e., $u(p, x)$ and $p(x)$ are both defined and equal whenever at least one of them is defined. Moreover, there is an $\mathrm{SRM}^{+,\tilde{O}}$-program c such that $c(p, x) = 1$ if x is a successful computation of p and $c(p, x) = 0$ otherwise. In particular, if O is the empty function, then $\tilde{O}$ can be taken empty as well.*

PROOF. Let $\tilde{O}$ be the function such that $\tilde{O}(x) = \langle 1, O(x)\rangle$ whenever $O(x)$ is defined and $\tilde{O}(x) = \langle 0, 0\rangle$ whenever $O(x)$ is undefined. Using Lemma 3.6 and $\tilde{O}$, it is straightforward (but tedious) to construct a program c such that $c(p, x) = 1$ if x is a successful computation of p and $c(p, x) = 0$ otherwise. Then note that $p(x)$ is defined if and only if there is a successful computation of p on input x. For this reason, the universal machine can be implemented as an unbounded search through V that stops if a successful computation for p on input x is found, and returns $p(x)$. In the case where O is the empty function, we can take $\tilde{O}$ to be the empty function as well because all SRM^+-operations are SRM^+-decidable. ⊣

It is possible to construct an $\mathrm{SRM}^{+,O}$-universal machine for $\mathrm{SRM}^{+,O}$, if one changes the definition of oracle evaluation in such a way that the universal machine can query the oracle without the risk of not halting.

Let $D(x, y)$ be a binary predicate in the language of set theory. Adapting from Kleene and Post [17], we write $D_z(x) := D(x, z)$ and define D^z to be the join of all D_y with $y \neq z$, as follows:

$$D^z(x, y) := \begin{cases} D(x, y), & \text{if } y \neq z, \\ 0, & \text{if } y = z. \end{cases}$$

The proof of the following theorem is a generalisation of a result by Kleene and Post [17, Theorem 2]; our proof will be a generalisation of their diagonal argument to the case of SRM^+.

THEOREM 3.15. *There is a set-theoretic predicate $D(x, y)$ such that D_z is not SRM^+-computable in D^z.*

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

PROOF. We define the predicate by informally describing a total $\mathrm{SRM}^{+,H}$-program that makes use of an oracle H for the SRM^{+}-halting problem.

Let R_{init} be an auxiliary register which is used to save an initial segment of the predicate we are defining. Let R_{stage} be an auxiliary register that contains an ordinal representing the current stage of the construction.

To ensure the non-computability desired in the theorem, we have to satisfy class-many conditions, for each SRM^{+}-program e (possibly with parameters) and set z:

$$\text{The program } e \text{ does not witness that } D_z \text{ is } \mathrm{SRM}^{+}\text{-computable in } D^z. \qquad (P_{e,z})$$

Apply the inverse of the Gödel pairing function to R_{stage} to obtain ordinals α and β. By Proposition 3.12, calculate $e := F_\tau^{-1}(\alpha)$ and $z := F_\tau^{-1}(\beta)$. We want to extend R_{init} in such a way that $P_{e,z}$ will hold. To this end, let x be the $<_\tau$-least set for which $R_{init}(x, z)$ is undefined. For convenience, let us say that E is a z-extension of R_{init} if $R_{init} \subseteq E$ and if $R_{init}(w, z)$ is undefined for some w then so is $E(w, z)$. There are two cases to consider.

CASE 1: There is a z-extension D_{init} of R_{init} such that there is a successful computation of e on input (x, z) using D_{init}^z as an oracle, i.e., the oracle is the predicate obtained from D_{init} by taking $D_{init}^z(w, y) = D_{init}(w, y)$ if $y \neq z$, and $D_{init}^z(w, z) = 0$ for all w. Note that our machine can decide whether such an extension exists by using the oracle for the SRM^{+}-halting problem. Let $y \in \{0, 1\}$ be the result of this computation. As D_{init} is a z-extension of R_{init}, it must be that $D_{init}(x, z)$ is undefined. We can therefore set $R_{init} := D_{init} \cup \{((x, z), 1 - y)\}$. This choice ensures that e does not witness that D_z is computable in D^z.

CASE 2: For all z-extensions D_{init} of R_{init} there is no successful computation of e on input (x, z) with D_{init}^z as oracle. In this case, we let $R_{init} := R_{init} \cup \{((x, y), 0)\}$. This (arbitrary) choice works because the final predicate D will be such that there is no successful computation of e on input (x, z) with oracle D^z: for contradiction, suppose there was such a successful computation c and consider the z-extension D_{init} of R_{init} given by $D_{init}(x, y) = D(x, y)$ for all (x, y), $y \neq z$, for which the oracle is called during the computation c. As $D_{init}^z(w, z)$ is defined for all w, all oracle calls during the computation c are still the same when using D_{init}^z instead of D^z. Hence, there is a successful computation c of e on input (x, z) with oracle D_{init}^z. But that is in contradiction to the assumption of this case.

The program defined this way will eventually give rise to a completely defined predicate D on $V \times V$. The value of $D(x, y)$ can be computed by running the procedure above until the value for (x, y) is known. ⊣

Note that the program described in the proof above does not use any parameters and can thus be coded as a natural number.

REMARK 3.16. In fact, Kleene and Post prove a stronger result which allows to locate D between any two Turing degrees. A similar result is possible here but we leave the proof to the interested reader as we do not need it.

3.3. Constructible SRMs. For our applications to the first-order logic of CZF, it will be important that we can express the predicate "$D(x, y)$ holds" in a way that only uses the language of set theory without introducing an extra relation symbol

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

into our language to refer to the global well-order. This means that we have to circumvent referring to $<_\tau$ as this is an extra symbol that cannot be defined in terms of a set-theoretic formula. Due to the following well-known fact, we will restrict our attention to constructible sets (for reference see, e.g., Jech [13, Theorem 13.18 and Lemma 13.19]):

FACT 3.17. *There is a* Σ_1*-definable well-ordering* $<_{\mathrm{L}}$ *of the constructible universe* L.

So if we restrict our attention to SRM^+s that work only on constructible sets, we can replace $<_\tau$ with $<_{\mathrm{L}}$ in Definition 3.2. The resulting notion of SRM will be called *constructible* SRM^+ and denoted, in short, by $\mathrm{SRM}^+_{\mathrm{L}}$. Note that all of the results obtained so far about SRM^+s can be relativised to L and thus transferred to $\mathrm{SRM}^+_{\mathrm{L}}$. In particular, we get the following versions of Lemma 3.7 and Theorem 3.15:

LEMMA 3.18. *Let* $\varphi(\bar{x})$ *be a* Δ_0*-formula. There is an* $\mathrm{SRM}_{\mathrm{L}}$*-program* p *such that* $p(\ulcorner\varphi\urcorner, \bar{x}) = 1$ *if* $L \vDash \varphi$ *and* $p(\ulcorner\varphi\urcorner, \bar{x}) = 0$ *if* $L \vDash \neg\varphi$.

COROLLARY 3.19. *There exists a non-*$\mathrm{SRM}^+_{\mathrm{L}}$*-computable set-theoretic predicate* $D(x, y)$*, expressible in the language of set theory, such that* D_z *is not* $\mathrm{SRM}^+_{\mathrm{L}}$*-computable in* D^z.

§4. Realisability. We will now define a notion of realisability based on SRM^+s, and observe a few proof-theoretic consequences for CZF.

DEFINITION 4.1. We define the realisability relation $\Vdash$ for an $\mathrm{SRM}^{(+),(O)}_{(\mathrm{L})}$ r recursively as follows:

(i) $r \Vdash a \in b$ if and only if $a \in b$;
(ii) $r \Vdash a = b$ if and only if $a = b$;
(iii) $r \Vdash \varphi_0 \wedge \varphi_1$ if and only if $r(0) \Vdash \varphi_0$ and $r(1) \Vdash \varphi_1$;
(iv) $r \Vdash \varphi_0 \vee \varphi_1$ if and only if $r(1) \Vdash \varphi_{r(0)}$;
(v) $r \Vdash \varphi_0 \to \varphi_1$ if and only if whenever $s \Vdash \varphi_0$, then $r(s) \Vdash \varphi_1$;
(vi) $r \Vdash \exists x \varphi(x)$ if and only if $r(1) \Vdash \varphi(r(0))$;
(vii) $r \Vdash \forall x \varphi(x)$ if and only if $r(a) \Vdash \varphi(a)$ for every set a.

We say that φ is SRM-realisable if and only if there is an SRM realising φ. Similarly, we say that φ is SRM^+-realisable if and only if there is an SRM^+ realising φ; and so for $\mathrm{SRM}^{+,O}$, $\mathrm{SRM}^+_{\mathrm{L}}$, and $\mathrm{SRM}^{+,O}_{\mathrm{L}}$.

This could be extended to infinitary languages as done by Carl, Galeotti, and Passmann [4]. Analogously to (i) and (ii), one could give realisability semantics to the global well-order $<_\tau$.

THEOREM 4.2. $\mathrm{SRM}^{(+),(O)}_{(\mathrm{L})}$*-realisability is sound for intuitionistic logic.*

PROOF. This is a standard argument and can be established, for example, by providing a realiser for every axiom in a Hilbert-style formalisation of **IQC** and showing that *modus ponens* is valid. The latter follows immediately from the definition of the realisability relation. ⊣

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

LEMMA 4.3. *Let $\varphi(\bar{x})$ be a Σ_1-formula. Then there is some realiser $r \Vdash \varphi(\bar{x})$ if and only if $\mathrm{V} \vDash \varphi(\bar{x})$.*

PROOF. This is a straightforward induction on Σ_1-formulas. We will prove a more intricate version of this lemma below (see Lemma 5.8). ⊣

THEOREM 4.4. *The axioms (and schemes) of extensionality, pairing, union, infinity, collection, $\in$-induction, and Δ_0-separation are* SRM*-realisable. The axiom of choice,* AC*, is* SRM*-realisable. The axioms of power set and strong collection are* SRM^+*-realisable. In conclusion,* IKP + AC *is* SRM*-realisable, and* CZF + PowerSet + AC *is* SRM^+*-realisable. Moreover,* IKP + AC *is* $\mathrm{SRM}_{\mathrm{L}}$*-realisable, and* CZF + PowerSet + AC *is* $\mathrm{SRM}^+_{\mathrm{L}}$*-realisable.*

PROOF. It is straightforward to construct a realiser for the extensionality axiom. For the empty set axiom, let r be an SRM that returns the empty set on input 0 and the identity function on input 1. Then $r(1) \Vdash \forall y(y \in r(0) \to \bot)$ because $\nVdash_w y \in \emptyset$ for all $w \in P$ and $y \in \mathrm{V}$. Hence, $r \Vdash \exists x \forall y(y \notin x)$. A realiser for the union axiom is an SRMr such that, for every $a \in \mathrm{V}$, $r(a)(0) = \bigcup a$, using Lemma 3.6, $r(a)(1)(x)(0) = \mathrm{id}$, and $r(a)(1)(x)(1) = \mathrm{id}$ for every x. The infinity axiom is realised by an SRMr with $r(0) = \omega$, $r(1)(x)(0) = \mathrm{id}$, and $r(1)(x)(1) = \mathrm{id}$ for every $x \in \mathrm{V}$. Using the power set operation provided by SRM^+-programs, it is straightforward to construct a realiser of the power set axiom. Note that the subset collection schema is a consequence of the power set axiom.

Let us consider Δ_0-separation next, i.e., the schema consisting of

$$\forall x \exists y \forall z(z \in y \leftrightarrow z \in x \wedge \varphi(x)),$$

where $\varphi(x)$ is a Δ_0-formula. By combining Lemmas 3.18 and 4.3, we know that $\Vdash \varphi(x)$ if and only if $p(\ulcorner \varphi \urcorner, x) = 1$, and $p(\ulcorner \varphi \urcorner, x) = 0$ in case $\nVdash \varphi(x)$. Hence, we can compute the witnessing set y by conducting a bounded search through x and collecting all $z \in x$ such that $p(\ulcorner \varphi \urcorner, z) = 1$. It is then trivial to realise $\forall z(z \in y \leftrightarrow z \in x \wedge \varphi(x))$ because φ is a Δ_0-formula.

Consider the schema of $\in$-induction next:

$$\forall x(\forall y \in x \varphi(z) \to \varphi(x)) \to \forall x \varphi(x).$$

An SRMr is a realiser for this if and only if, if $s \Vdash \forall x(\forall y \in x \varphi(z) \to \varphi(x))$, then $r(s) \Vdash \forall x \varphi(x)$. Now, in this situation, s allows us to iteratively construct realisers for every $x \in \mathrm{V}$ by successively building realisers for every V_α. Hence, given $x \in \mathrm{V}$, we just compute realisers until we reach x and then output the realiser for $\varphi(x)$.

Next, we consider the strong collection schema:

$$\forall x[(\forall y \in x \exists z \varphi(y, z)) \to \exists w(\forall y \in x \exists z \in w \varphi(y, z) \wedge \forall z \in w \exists y \in x \varphi(y, z))],$$

for all formulas $\varphi(x, y)$ for which w is not free. Given $x \in \mathrm{V}$, let $r(x)(s)$, for $s \Vdash \forall y \in x \exists z \varphi(z, y)$, be an SRM that computes a set consisting of all $s(y)(0)$ for every $y \in x$, and returns this set on input 0. Using s, it is straightforward to construct a realiser $r(x)(s)(1) \Vdash \forall y \in x \exists z \in r(x)(s)(0)\ \varphi(y, z) \wedge \forall z \in r(x)(s)(0)\ \exists y \in x \varphi(y, z))$.

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

Finally, consider the axiom of choice,

$$\forall x((\forall y \in x \exists z\ z \in y) \to \exists f \forall y \in x\ f(y) \in y).$$

This axiom states that whenever x consists of non-empty sets, then there is a choice function f on x. Using Lemma 3.6, it is straightforward to construct an SRM that computes such a choice function: for every element of $y \in x$, use the TAKE-operation to obtain some $z \in y$. Then add (x, y) to the register in which we build the choice function.

The corresponding results for $\mathrm{SRM}_{\mathrm{L}}$ and $\mathrm{SRM}_{\mathrm{L}}^{+}$ are obtained through relativisation and absoluteness properties (or by observing that the exact same realisers still do the job). ⊣

It turns out that IZF is not SRM^{+}-realisable.

THEOREM 4.5. *There is an instance of the separation axiom that is not* SRM^{+}*-realisable. In conclusion,* IZF *is not* SRM^{+}*-realisable.*

PROOF. Consider the predicate $H(x, y)$ expressing that "x is an SRM^{+} that halts on input y." One can easily construct a formula $\varphi(x, y)$ such that $\varphi(x, y)$ is realised if and only if $H(x, y)$ is true (see also the proof of Lemma 5.11 for a similar argument). Then let s be a realiser of the following instance of the separation axiom:

$$\forall x \forall y \forall z \exists w \forall u(u \in w \leftrightarrow (u \in z \wedge \varphi(x, y))).$$

We can then construct an $\mathrm{SRM}r$ that does the following. Given x and y, compute $w := s(x)(y)(1)(0)$ and return the result. By construction, $r(x, y) = 1$ just in case $H(x, y)$ holds, and $r(x, y) = 0$ otherwise. So r is an SRM^{+} solving the SRM^{+} halting problem but this is impossible (see Proposition 3.13). ⊣

In fact, we have just seen that CZF + PowerSet is SRM^{+}-realisable. The following proposition shows that we cannot be more fine-grained: if there is an SRM realising the exponentiation axiom (possibly using an oracle), then we can already compute power sets. Recall that the axiom of exponentiation is a consequence of subset collection (Fact 2.2).

PROPOSITION 4.6. *Let* r *be an* SRM*, possibly using an oracle, such that* r *realises the axiom of exponentiation, then there is an* SRM*, using* r *as an oracle, that computes power sets.*

PROOF. Let r be a realiser of the axiom of exponentiation:

$$\forall x \forall y \exists z \forall f(f \in z \leftrightarrow \text{“}f \text{ is a function from } x \text{ to } y\text{”}),$$

where "f is a function from x to y" is expressed as a Δ_0-formula. Then, given a set a, the set $b := r(a)(\{0, 1\})(0)$ contains all f for which there is a realiser of "f is a function from x to y." As this is a Δ_0-formula, Lemma 4.3 implies that b consists of all functions from a to 2. It is now easy to compute the power set of a as follows: for each element f of b, compute the set consisting of exactly those $x \in a$ for which $f(a) = 1$. This results in the power set of a because each subset of a gives rise to its characteristic function contained in b. ⊣

Our realisability semantics also allow to give an upper bound for Π_2-formulas provable in CZF in terms of the computable strength of SRM^{+}.

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

THEOREM 4.7. *Let φ be a Σ_1-formula. If* CZF $\vdash \forall x \exists y \varphi(x, y)$, *then there is an* SRM^+ p *such that* $\mathrm{V} \vDash \varphi(x, p(x))$.

PROOF. If CZF $\vdash \forall x \exists y \varphi(x, y)$, then, by Theorem 4.4, there exists an $\mathrm{SRM}^+ r \Vdash \forall x \exists y \varphi(x, y)$. Take $p(x)$ to be the SRM^+ to compute $r(x)(0)$. Then, for all x, $\varphi(x, p(x))$ is realisable. As φ is a Σ_1-formula, it follows with Lemma 4.3 that $\mathrm{V} \vDash \varphi(x, p(x))$. ⊣

Finally, we can use SRM^+-realisability to easily determine the admissible rules of CZF. A proof of Carl, Galeotti, and Passmann [4, Theorem 56] can be adapted to work here.

THEOREM 4.8. *The propositional admissible rules of* CZF *are exactly those of intuitionistic logic.*

PROOF. Using the fact that CZF is SRM^+-realisable, we can prove this with glued realisability using Theorem 2.10; almost exactly as we did in earlier joint work with Carl, Galeotti, and Passmann [4]. ⊣

§5. Beth realisability models.

5.1. Fallible Beth models. In this section, we will make use of so-called *fallible* Beth models because they satisfy a particular handy universal model theorem.

DEFINITION 5.1. A *fallible Beth frame* (P, U) consists of a tree P and an upwards closed set $U \subseteq P$ such that if every path through $p \in P$ meets U, then $p \in U$.

DEFINITION 5.2 (Fallible Beth model). A *fallible Beth model* (P, U, D, I) for first-order logic consists of a fallible Beth tree (P, U), domains D_p for $p \in P$, and an interpretation I_p of the language of first-order logic for each $p \in P$ such that:

(i) $I_v(R) \subseteq I_w(R)$ for all $w \geq v$,
(ii) $I_v(R) = D_v$ for all $v \in U$, and
(iii) if R is an n-ary relation symbol, $\bar{x} \in D_v^n$ and on every path through v there is some w such that $\bar{x} \in I_w(R)$, then $\bar{x} \in I_v(R)$.

A *Beth model* is a fallible Beth model where $U = \emptyset$. If $p \in P$, then a *bar for* p is a set $B \subseteq P$ such that every path through p meets B. A *U-bar for* p is a set $B \subseteq P$ such that $B \cup U$ is a bar for p.

DEFINITION 5.3. Let (P, U, D, I) be a fallible Beth model and $v \in P$. We define by recursion on sentences in the language of first-order logic:

(i) $v \Vdash \bot$ if and only if $v \in U$;
(ii) $v \Vdash R(d_1, \ldots, d_n)$ if and only if $(d_1, \ldots, d_n) \in I_v(R)$;
(iii) $v \Vdash A_0 \wedge A_1$ if and only if $v \Vdash A_0$ and $v \Vdash A_1$;
(iv) $v \Vdash A_0 \vee A_1$ if and only if there is a bar B for v such that for every $w \in B$, $w \Vdash A_0$ or $w \Vdash A_1$;
(v) $v \Vdash A_0 \to A_1$ if and only if for every $w \geq v$, if $w \Vdash A_0$, then $w \Vdash A_1$;
(vi) $v \Vdash \exists x A(x)$ if and only if there is a bar B for v such that for all $w \in B$, there is some $a \in D_w$ with $w \Vdash A(a)$;
(vii) $v \Vdash \forall x A(x)$ if and only if for every $w \geq v$ and $a \in D_w$, $w \Vdash A(a)$.

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

Note that, by this definition, if $v \in U$, then v forces every formula trivially, i.e., the relation $\Vdash$ trivialises in U. By definition of U, it follows that if $v \notin U$ and B is a U-bar for v, then $B \setminus U$ is non-empty. The following result of Troelstra and van Dalen [24, Chapter 13, Remark 2.6 and Theorem 2.8] will be a crucial ingredient of our proof.

THEOREM 5.4. *Let J be a recursively enumerable theory in intuitionistic first-order logic. Then there is a fallible Beth model $\mathcal{B}_J$ with constant domain ω, based on the full binary tree of height ω, such that $\mathcal{B} \Vdash A$ if and only if $J \vdash A$ for every sentence A of first-order logic.*

In what follows, we will refer to $\mathcal{B}_J$ as the *universal Beth model for J*.

5.2. Beth realisability models. Inspired by van Oosten [19], we now combine our notion of $\mathrm{SRM}_{\mathrm{L}}^{+,O}$-realisability with Beth semantics. To make coherent use of oracles, we need the following definition.

DEFINITION 5.5. Let P be a partial order. A *system of oracles* $(O_v)_{v\in P}$ consists of partial class functions $O_v : \mathrm{V} \to \mathrm{V}$ such that, for all $w \geq v$, we have that $\mathrm{dom}(O_v) \subseteq \mathrm{dom}(O_w)$ and $O_v(x) = O_w(x)$ for all $x \in \mathrm{dom}(O_v)$.

We need some notation to work with oracles. Given an $\mathrm{SRM}_{\mathrm{L}}^{+,O}$-program r, we write $r(x_1, \dots, x_n; O)$ for the result of the successful computation (if it exists) of r on input $x_1, \dots, x_n$ and oracle O. If we work with a system of oracles $(O_v)_{v\in P}$, we also write $r(x_1, \dots, x_n; v)$ to mean $r(x_1, \dots, x_n; O_v)$. Finally, we write $r(x_1, \dots, x_n)$ to mean $r(x_1, \dots, x_n; \emptyset)$, i.e., the output (if it exists) of r run with the empty oracle.

DEFINITION 5.6. Let (P, U) be a fallible Beth frame, $(O_v)_{v\in P}$ be a system of oracles. We define recursively for sentences φ and ψ in the language of set theory, for $a, b \in \mathrm{L}$, $v \in P$ and an $\mathrm{SRM}_{\mathrm{L}}^{+,O}$-program r:

(i) $r \Vdash_v \bot$ if and only if $v \in U$;
(ii) $r \Vdash_v a = b$ if and only if $a = b$ or $v \in U$;
(iii) $r \Vdash_v a \in b$ if and only if $a \in b$ or $v \in U$;
(iv) $r \Vdash_v \varphi \wedge \psi$ if and only if $r(0; v) \Vdash_v \varphi$ and $r(1; v) \Vdash_v \psi$;
(v) $r \Vdash_v \varphi \vee \psi$ if and only if there is a U-bar B for v such that, for every $w \in B$, either $r(0; w) = 0$ and $r(1; w) \Vdash_w \varphi$, or $r(0; w) = 1$ and $r(1; w) \Vdash \psi$;
(vi) $r \Vdash_v \varphi \to \psi$ if and only if for every $w \geq v$, if $s \Vdash_w \varphi$, then $r(s; w) \Vdash_w \psi$;
(vii) $r \Vdash_v \exists x \varphi(x)$ if and only if there is a U-bar B for v such that for all $w \in B$, $r(1; w) \Vdash_w \varphi(r(0; w))$;
(viii) $r \Vdash_v \forall x \varphi(x)$ if and only if for every a, $r(a; v) \Vdash_v \varphi(a)$.

If $v \in U$, then $r \Vdash_v \varphi$ for every realiser r and set-theoretic sentence φ. The following is established by a standard argument.

THEOREM 5.7. *Beth-realisability is sound for the axioms and rules of intuitionistic first-order logic.*

LEMMA 5.8. *Let $\varphi(\bar{x})$ be a Σ_1-formula and $v \notin U$. Then there is some realiser $r \Vdash_v \varphi(\bar{x})$ if and only if $L \vDash \varphi(\bar{x})$.*

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

PROOF. As $v \notin U$, we know that any U-bar B for v satisfies $B \setminus U \neq \emptyset$. We prove this by induction. The cases for equality and set-membership are trivial.

Suppose that $\Vdash_v \varphi(\bar{a}) \wedge \psi(\bar{a})$. By definition, this is equivalent to $\Vdash_v \varphi(\bar{a})$ and $\Vdash_v \psi(\bar{a})$. Applying the induction hypothesis, this holds if and only if $\mathrm{L} \vDash \varphi(\bar{a})$ and $\mathrm{L} \vDash \psi(\bar{a})$. This is, of course, equivalent to $\mathrm{L} \vDash \varphi(\bar{a}) \wedge \psi(\bar{a})$.

For disjunction, first suppose that $r \Vdash_v \varphi(\bar{a}) \vee \psi(\bar{a})$. By definition, this means that there is a U-bar B for v such that for all $w \in B$ we have either $r(0; w) = 0$ and $r(1; w) \Vdash_w \varphi(\bar{a})$, or $r(0; w) = 1$ and $r(1; w) \Vdash_w \psi(\bar{a})$. Recall that $B \setminus U$ is non-empty. So take any $w \in B \setminus U$, then $\Vdash_w \varphi(\bar{a})$ or $\Vdash_w \psi(\bar{a})$. By induction hypothesis, $\mathrm{L} \vDash \varphi(\bar{a})$ or $\mathrm{L} \vDash \psi(\bar{a})$. Hence $\mathrm{L} \vDash \varphi(\bar{a}) \vee \psi(\bar{a})$. Conversely, assume that $\mathrm{L} \vDash \varphi(\bar{a}) \vee \psi(\bar{a})$. Then $\mathrm{L} \vDash \varphi(\bar{a})$ or $\mathrm{L} \vDash \psi(\bar{a})$. It follows, by induction hypothesis, that $\Vdash_v \varphi(\bar{a})$ or $\Vdash_v \psi(\bar{a})$, but then $\Vdash_v \varphi(\bar{a}) \vee \psi(\bar{a})$.

For implication, assume that $r \Vdash_v \varphi \to \psi$. If $\mathrm{L} \nvDash \varphi$, then trivially $\mathrm{L} \vDash \varphi \to \psi$. So assume that $\mathrm{L} \vDash \varphi$. By induction hypothesis, we know that there is a realiser $s \Vdash_v \varphi$. Hence, $r(s) \Vdash_v \psi$. Applying the induction hypothesis once more, we get $\mathrm{L} \vDash \psi$. Conversely, assume that $\mathrm{L} \vDash \varphi \to \psi$. If $\mathrm{L} \nvDash \varphi$, then, by induction hypothesis, $\nVdash_w \varphi$ for all $w \geq v$. So $\Vdash_v \varphi \to \psi$ holds trivially. If $\mathrm{L} \vDash \varphi$, then $\mathrm{L} \vDash \psi$. So, by induction hypothesis, there is a realiser $s \Vdash_v \psi$. Hence, a realiser for $\varphi \to \psi$ is the SRM p that returns s on any input.

For bounded universal quantification, assume that $\mathrm{L} \vDash \forall x \in y \varphi(x)$. Then, by induction hypothesis, we can find a function $f : y \to \mathrm{L}$ such that $f(z) \Vdash_v \varphi(z)$. Let p be the SRM with parameter f that returns $f(z)$ on input z. Then $p \Vdash_v \forall x \in y \varphi(x)$. Conversely, note that $\Vdash_v \forall x \in y \varphi(x)$ entails that $\Vdash_v \varphi(x)$ for every $x \in y$. An application of the induction hypothesis yields $\mathrm{L} \vDash \forall x \in y \varphi(x)$.

For unbounded existential quantification, assume that $\mathrm{L} \vDash \exists x \varphi(x)$. Then there is some $a \in \mathrm{L}$ such that $\mathrm{L} \vDash \varphi(a)$. By induction hypothesis, there is a realiser $s \Vdash_v \varphi(a)$. Let p be an SRM such that $p(1) = s$ and $p(0) = a$ (by using, if necessary, parameter a). Then $p \Vdash_v \exists x \varphi(x)$. Conversely, if $p \Vdash_v \exists x \varphi(x)$, then there is a U-bar B for v such that for all $w \in B$, $p(1; w) \Vdash_w \varphi(p(0; w))$. Take any $w \in B$ and the induction hypothesis implies that $\mathrm{L} \vDash \varphi(p(0; w))$, and, hence, $\mathrm{L} \vDash \exists x \varphi(x)$. ⊣

THEOREM 5.9. *The Beth realisability model satisfies* CZF + PowerSet + AC.

PROOF. Realisers for the axioms and schemas can be constructed (almost exactly) as in the proof of Theorem 4.4. For the case of Δ_0-separation, observe that the use of Lemma 4.3 has to be replaced with Lemma 5.8. (Note that we only need to consider the cases for $v \notin U$, as the other case is trivial.) ⊣

5.3. Constructing a model for a given logic. The goal of this section is to construct a Beth-realisability model that matches the truth in the universal Beth model $\mathcal{B}_J = (P, U, D, I)$ for a given logic J. To begin with, we define the two systems of oracles $(F_v)_{v \in P}$ and $(G_v)_{v \in P}$. If a is a set, let $\mathrm{rank}_\omega(a)$ be the unique natural number such that $\mathrm{rank}(a) = \alpha + \mathrm{rank}_\omega(a)$ for a maximal (possibly 0) limit ordinal α.

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

(i) We define $F_v : \mathrm{V} \times \mathrm{V} \to \mathrm{V}$ by recursion on $v \in P$ (P being the binary tree of height ω) such that:

$$F_v(m, \langle b_0, \dots, b_n \rangle) = \begin{cases} a, & \text{if } m = \ulcorner \exists x A(x, y_0, \dots, y_n) \urcorner \\ & \quad \text{and } w \leq v \text{ is least such that} \\ & \quad a \in \omega \text{ is least with} \\ & \quad \mathcal{B}_J, w \Vdash A(a, \mathrm{rank}_\omega(b_0), \dots, \mathrm{rank}_\omega(b_n)), \\ i, & \text{if } m = \ulcorner (A_0 \vee A_1)(y_0, \dots, y_n) \urcorner \\ & \quad \text{and } w \leq v \text{ is least such that} \\ & \quad i \in \omega \text{ is least with} \\ & \quad \mathcal{B}_J, w \Vdash A_i(\mathrm{rank}_\omega(b_0), \dots, \mathrm{rank}_\omega(b_n)), \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

(ii) We define $G_v : \mathrm{V} \times \mathrm{V} \to \mathrm{V}$ such that:

$$G_v(a, b) = \begin{cases} 1, & \text{if } b = \langle i, b_0, \dots, b_n \rangle, \\ & \quad \mathcal{B}_J, v \Vdash P_i(\mathrm{rank}_\omega(b_0), \dots, \mathrm{rank}_\omega(b_n)), \\ & \quad \text{and } D(a, b) = 1, \\ 0, & \text{if } b = \langle i, b_0, \dots, b_n \rangle, \\ & \quad \mathcal{B}_J, v \Vdash P_i(\mathrm{rank}_\omega(b_0), \dots, \mathrm{rank}_\omega(b_n)), \\ & \quad \text{and } D(a, b) = 0, \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

LEMMA 5.10. *The sequences $(F_v)_{v \in P}$ and $(G_v)_{v \in P}$ form systems of oracles.*

From now on, we consider the Beth-realisability based on these systems of oracles. Note that, without loss of generality, we can combine two systems of oracles into one by, e.g., taking $O_v(\langle 0, x \rangle) = F_v(x)$ and $O_v(\langle 1, x \rangle) = G_v(x)$ for all $v \in P$.

LEMMA 5.11. *Let $v \notin U$. There is a negative formula $\psi(x, y)$ such that there is a realiser $r \Vdash_v \psi(x, y)$ is realised if and only if $D(x, y) = 1$.*

PROOF. Except for the power set case, every clause of the definition of successful computation (Definition 3.2), adapted for $\mathrm{SRM}^+_{\mathrm{L}}$, can be written as a Σ_1-formula. For the TAKE-operation, recall that $<_{\mathrm{L}}$ is Σ_1-definable. Now consider the predicate "$x = \mathcal{P}(y)$" which is needed for the POW-operation and can be formalised as "$\forall z (z \in x \leftrightarrow \forall w \in z\, w \in y)$." As the part in brackets is a Δ_0-formula, it follows with Lemma 5.8 that this predicate is realised if and only if it is true. Note, in particular, that also the successor case for the halting problem oracle is realised if and only if it is true in L. This is because the existence of a successful computation is absolute, as we have just seen.

Applying Lemma 5.8 once more, these observations show that we can construct a formula χ expressing "c is a successful computation of $D(x, y)$ with result 0" such that $\chi(c, x, y)$ is realised if and only if it is true in L. Take $\psi(x, y)$ to be $\neg \exists c \chi(c, x, y)$. It follows that $\psi(x, y)$ is realised if and only if $D(x, y) = 1$ because D halts on every input with either 0 or 1 as output. ⊣

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

LEMMA 5.12. *Let $P_i(y_0, \dots, y_n)$ be a predicate in the language of first-order logic. There is a set-theoretic formula $\varphi_i(y_0, \dots, y_n)$ and a realiser r such that for all $b_0, \dots, b_n \in \mathrm{L}$, $r(b_0, \dots, b_n) \Vdash_v \varphi_i(b_0, \dots, b_n)$ if and only if $\mathcal{B}_J, v \Vdash P_i(\mathrm{rank}_\omega(b_0), \dots, \mathrm{rank}_\omega(b_n))$.*

PROOF. Let $\psi(x, y)$ be the negative formula from Lemma 5.11 expressing that $D(x, y) = 1$. As ψ is negative, we know that, for every v and $a, b \in \mathrm{L}$, either $\Vdash_v \psi(a, b)$ or $\Vdash_v \neg\psi(a, b)$. Then take:

$$\varphi_i(y_0, \dots, y_n) \quad := \quad \forall x(\psi(x, \langle i, y_0, \dots, y_n\rangle) \vee \neg\psi(x, \langle i, y_0, \dots, y_n\rangle)).$$

Suppose there was a realiser $r \Vdash_v \varphi_i(b_0, \dots, b_n)$ but we also have that $\mathcal{B}_J, v \nVdash P_i(\mathrm{rank}_\omega(b_0), \dots, \mathrm{rank}_\omega(b_n))$. In this situation, we can decide $D_{\langle i, b_0, \dots, b_n\rangle}$ from r for every a: if $r(a, b_0, \dots, b_n)$ returns a realiser for $\psi(a, \langle i, b_0, \dots, b_n\rangle)$, then $D_{\langle i, b_0, \dots, b_n\rangle}(a) = 1$; if $r(a, b_0, \dots, b_n)$ returns a realiser for $\neg\psi(a, \langle i, b_0, \dots, b_n\rangle)$, then $D_{\langle i, b_0, \dots, b_n\rangle}(a) = 0$. However, by our assumption, $G_v(c, \langle i, b_0, \dots, b_n\rangle)$ is undefined for all $c \in \mathrm{L}$. This means that r cannot query the oracle G_v on elements of the form $(c, \langle i, b_0, \dots, b_n\rangle)$ because then the computation would not be successful. Hence, using r, we can construct a witnesses that $D_{\langle i, b_0, \dots, b_n\rangle}$ is computable in $D^{\langle i, b_0, \dots, b_n\rangle}$ but that is a contradiction to Theorem 3.19. (Note that F does not matter here because the information contained in F could be saved in a set-sized parameter.)

Conversely, assume that $\mathcal{B}_J, v \Vdash P_i(\mathrm{rank}_\omega(b_0), \dots, \mathrm{rank}_\omega(b_n))$. By definition of G, it follows that $G_v(a, \langle i, b_0, \dots, b_n\rangle)$ is defined for all $a \in \mathrm{L}$. Hence, a realiser for φ_i can be easily obtained by querying the oracle $G(a, \langle i, b_0, \dots, b_n\rangle)$: if the result is 1, then return a realiser of $\psi(a, \langle i, b_0, \dots, b_n\rangle)$. If the result is 0, then return a realiser of $\neg\psi(a, \langle i, b_0, \dots, b_n\rangle)$. In both cases, the computation of the corresponding realiser is trivial because the formulas are negative. ⊣

Let $\tau(P_i) = \varphi_i$ and extend τ to a translation of all formulas in the language of first-order logic in the obvious way. Note that the formulas φ_i are Π_3-formulas.

LEMMA 5.13. *Let $A(y_0, \dots, y_n)$ be a formula in the language of first-order logic. Then*:

(i) *If there is a realiser $r \Vdash_v A^\tau(b_0, \dots, b_n)$, then*

$$\mathcal{B}_J, v \Vdash A(\mathrm{rank}_\omega(b_0), \dots, \mathrm{rank}_\omega(b_n)).$$

(ii) *There is a realiser r_A such that for all $b_0, \dots, b_n \in \mathrm{L}$, if*

$$\mathcal{B}_J, v \Vdash A(\mathrm{rank}_\omega(b_0), \dots, \mathrm{rank}_\omega(b_n)),$$

then

$$r_A(b_0, \dots, b_n) \Vdash_v A^\tau(b_0, \dots, b_n).$$

PROOF. We prove (i) and (ii) simultaneously by induction so that both directions are available in the induction hypothesis. We begin with proving the cases for (i). The base case follows from Lemma 5.12. For conjunction, $A \wedge B$, note that $\Vdash_v A^\tau \wedge B^\tau$ entails $\Vdash_v A^\tau$ and $\Vdash_v B^\tau$. Hence, by induction hypothesis, $\mathcal{B}_J, v \Vdash A$ and $\mathcal{B}_J, v \Vdash B$. So, $\mathcal{B}_J, v \Vdash A \wedge B$. For disjunction, $A \vee B$, we have that $r \Vdash_v A^\tau \vee B^\tau$ entails that there is a U-bar B for v such that for every $w \in B$, either $r^w(0) = 0$ and $r^w(1) \Vdash_w A^\tau$ or $r^w(0) = 1$ and $r^w(1) \Vdash_w B^\tau$. By induction hypothesis, this means that there is a

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

U-bar B for v such that for every $w \in B$, $w \Vdash A$ or $w \Vdash B$. Hence $v \Vdash A \vee B$. The case for implication is similar (making use of (ii) as well), and the cases for universal and existential quantification follow with the induction hypothesis.

Regarding the cases for (ii), we recursively construct the required realisers $r_A(b_0, \dots, b_n)$, uniform in $b_0, \dots, b_n \in \mathrm{L}$, for each formula A. Once more, the base case, $r_{P_i}(y_0, \dots, y_n)$, was established in Lemma 5.12. To keep notation light, we will write $\bar{y}$ for $y_0, \dots, y_n$ (or, potentially, a subsequence of this), and similarly for $\bar{b}$.

For conjunction $(A \wedge B)(\bar{y})$, take $r_{(A\wedge B)(\bar{y})}(\bar{b})(0) = r_A(\bar{b})$ and $r_{(A\wedge B)(\bar{y})}(\bar{b})(1) = r_B(\bar{b})$. An application of the induction hypothesis shows that $r_{(A\wedge B)(\bar{y})}$ does the job.

For implication $(A \to B)(\bar{y})$, we know by our induction hypothesis—for both (i) and (ii)—that $r_{B(\bar{y})}(\bar{b}) \Vdash_w B(\bar{b})$ if and only if $w \Vdash B(\bar{b})$ for all $w \geq v$. Hence, let $r_{A\to B(\bar{y})}(\bar{b}, s) = r_B(\bar{b})$. It is straightforward to check that this does the job.

For disjunction, define $r_{A\vee B}(\bar{y})$ to be the $\mathrm{SRM}^{+,O}$ that, on input $\bar{b}$, returns a code s for an $\mathrm{SRM}^{+,O}$ with parameters $\bar{b}$ that does the following. On input 0, s calls the oracle F on $(\ulcorner (A \vee B)(\bar{y}) \urcorner, \langle \bar{b} \rangle)$ and returns this value. On input 1, s returns $r_A(\bar{b})$ if $F(\ulcorner (A \vee B)(\bar{y}) \urcorner, \langle \bar{b} \rangle) = 0$ and it returns $r_B(\bar{b})$ otherwise. To see that $r_{(A\vee B)(\bar{y})}$ does the job, assume that there is a U-bar B such that for every $w \in B$, $w \Vdash A(\bar{b})$ or $w \Vdash B(\bar{b})$. Equivalently, by induction hypothesis, for every $w \in B$, $r_A(\bar{b}; w) \Vdash_w A(\mathrm{rank}_\omega(b_0), \dots, \mathrm{rank}_\omega(b_n))$ or $r_B(\bar{b}; w) \Vdash_w B(\mathrm{rank}_\omega(b_0), \dots, \mathrm{rank}_\omega(b_n))$. By definition of $r_{(A\vee B)(\bar{y})}$, it follows that $r_{(A\vee B)(\bar{y})}(\bar{b}; w)(1) = r_A$ or $r_{(A\vee B)(\bar{y})}(\bar{b}; w)(1) = r_B$. In conclusion, $r_{(A\vee B)(\bar{y})}(\bar{b}; w) \Vdash_w (A \vee B)(\bar{b})$.

For existential quantification, define $r_{\exists x A(x,\bar{y})}$ to be the function that, on input $\bar{b}$, calls the oracle F on input $(\ulcorner \exists x A(x, y) \urcorner, \langle \bar{b} \rangle)$. Let the result of this query be $n \in \omega$. Then let $r_{\exists x A(x,\bar{y})}(0) = n$ and $r_{\exists x A(x,\bar{y})}(1) = r_{A(n,\bar{y})}$. Note here that we do not require the use of parameters because the realiser $r_{A(n,\bar{y})}$ is uniform in $n, \bar{y}$. To check that $r_{\exists x A(x,\bar{y})}$ does the job, let $\bar{b} \in \mathrm{L}$ and assume that there is a U-bar B for v such that, for every $w \in B$, there is some $n_w \in \omega$ such that $w \Vdash A(n_w, \mathrm{rank}_\omega(\bar{b}))$. By induction hypothesis, it follows that $r_{A(n_w,\bar{y})}(\bar{b}; w) \Vdash_w A^\tau(n_w, \bar{b})$ (as $\mathcal{B}_J$ has constant domain ω and $\mathrm{rank}_\omega(n_w) = n_w$), i.e., $r_{\exists x A(x,\bar{y})}(\bar{b}, 0; w) \Vdash_w A^\tau(r_{\exists x A(x,\bar{y})}(\bar{b}, 1; w), \bar{b})$. Hence, $r_{\exists x A(x,\bar{y})}(\bar{b}; v) \Vdash_v \exists x A^\tau(x, \bar{b})$.

For universal quantification, define $r_{\forall x A(x,\bar{y})}(\bar{y})$ to be the function that returns $r_{A(x,\bar{y})}(x, \bar{y})$. ⊣

If J is a set of formulas in first-order logic, we write J^τ for the image of J under τ (i.e., $J^\tau = \tau[J]$).

THEOREM 5.14. *Let J be a recursively enumerable theory in intuitionistic first-order logic, and $T \subseteq \mathrm{CZF} + \mathrm{PowerSet} + \mathrm{AC}$. Then $T + J^\tau \vdash A^\tau$ if and only if $J \vdash_{\mathbf{IQC}} A$.*

PROOF. The backwards direction is straightforward with the soundness of the Beth realisability model. For the forward direction, assume that $J \nvdash A$. Then, by Theorem 5.4, we know that $\mathcal{B}_J \nVdash A$. In this situation, Lemma 5.13 implies that there is no realiser of A^τ. But the same lemma implies that B^τ is realised for every $B \in J$. Hence, $T + J^\tau \nvdash A^\tau$. ⊣

The following corollary follows immediately by taking $J = \emptyset$.

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

COROLLARY 5.15. *Let* $T \subseteq \mathrm{CZF} + \mathrm{PowerSet} + \mathrm{AC}$ *be a set theory. Then the first-order logic of* T *is intuitionistic first-order logic,* $\mathbf{QL}(T) = \mathbf{IQC}$. *In particular,* $\mathbf{QL}(\mathrm{CZF}) = \mathbf{IQC}$.

REMARK 5.16. Rathjen [21] points out that "the combination of CZF and the general axiom of choice has no constructive justification in Martin-Löf type theory." In contrast, our results show that the combination of CZF and the axiom of choice is innocent *on a logical level* in that adding the axiom of choice does not result in an increase of logical strength: $\mathbf{QL}(\mathrm{CZF} + \mathrm{AC}) = \mathbf{QL}(\mathrm{CZF}) = \mathbf{IQC}$. Note, of course, that $\mathrm{CZF} + \mathrm{AC}$ satisfies the law of excluded middle for Δ_0-formulas. This follows from the proof of Diaconescu's theorem (see Section 2) which only requires Δ_0-separation to prove the law of excluded middle for Δ_0-formulas. Such theories satisfying the law of excluded middle for Δ_0-formulas but not in general are sometimes called semi-intuitionistic.

Acknowledgments. I am thankful for the very helpful remarks of an anonymous reviewer. Moreover, I would like to thank Merlin Carl, Lorenzo Galeotti, Rosalie Iemhoff, Benedikt Löwe, Benno van den Berg, and Ned Wontner for helpful discussions. I thank Daniël Otten for spotting a few typos. This research was supported by a doctoral scholarship of the *Studienstiftung des deutschen Volkes* (German Academic Scholarship Foundation).

REFERENCES

[1] P. ACZEL and M. RATHJEN, *Notes on constructive set theory*, 2010. http://www1.maths.leeds.ac.uk/∼rathjen/book.pdf.

[2] B. VAN DEN BERG and I. MOERDIJK, *Derived rules for predicative set theory: An application of sheaves*. ***Annals of Pure and Applied Logic***, vol. 163 (2012), no. 10, pp. 1367–1383.

[3] M. CARL, ***Ordinal Computability: An Introduction to Infinitary Machines***, De Gruyter Series in Logic and Its Applications, vol. 9, De Gruyter, Berlin, 2020.

[4] M. CARL, L. GALEOTTI, and R. PASSMANN, *Realisability for infinitary intuitionistic set theory*, preprint, 2021, arXiv:2009.12172.

[5] R. DIACONESCU, *Axiom of choice and complementation*. ***Proceedings of the American Mathematical Society***, vol. 51 (1975), pp. 176–178.

[6] A. A. FRAENKEL, Y. BAR-HILLEL, and A. LEVY, ***Foundations of Set Theory***, Elsevier, Amsterdam, 1973.

[7] H. M. FRIEDMAN and A. ŠČEDROV, *On the quantificational logic of intuitionistic set theory*. ***Mathematical Proceedings of the Cambridge Philosophical Society***, vol. 99 (1986), no. 1, pp. 5–10.

[8] N. GOODMAN and J. MYHILL, *Choice implies excluded middle*. ***Mathematical Logic Quarterly***, vol. 24 (1978), nos. 25–30, pp. 461–461.

[9] R. IEMHOFF, *On the admissible rules of intuitionistic propositional logic*, this JOURNAL, vol. 66 (2001), no. 1, pp. 281–294.

[10] ———, *Intermediate logics and Visser's rules*. ***Notre Dame Journal of Formal Logic***, vol. 46 (2005), no. 1, pp. 65–81.

[11] R. IEMHOFF and R. PASSMANN, *Logics of intuitionistic Kripke–Platek set theory*. ***Annals of Pure and Applied Logic***, vol. 172 (2021), no. 10, Article no. 103014, 22 pp.

[12] ———, *Logics and admissible rules of constructive set theory*, submitted, 2022.

[13] T. JECH, ***Set Theory: The Third Millennium Edition, Revised and Expanded***, Springer Monographs in Mathematics, Springer, Berlin, 2003.

[14] D. DE JONGH, *The maximality of the intuitionistic predicate calculus with respect to Heyting's arithmetic,* unpublished article with abstract appearing in [15], 1968.

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press

[15] ———, *The maximality of the intuitionistic predicate calculus with respect to Heyting's arithmetic*, this JOURNAL, vol. 35 (1970), no. 4, p. 606.

[16] D. DE JONGH, R. VERBRUGGE, and A. VISSER, *Intermediate logics and the de Jongh property*. ***Archive for Mathematical Logic***, vol. 50 (2011), no. 1, pp. 197–213.

[17] S. C. KLEENE and E. L. POST, *The upper semi-lattice of degrees of recursive unsolvability*. ***Annals of Mathematics. Second Series***, vol. 59 (1954), pp. 379–407.

[18] D. LEIVANT, ***Absoluteness of Intuitionistic Logic***, ILLC Historical Dissertations Series (HDS), vol. 13, Institute for Logic, Language and Computation, University of Amsterdam, Amsterdam, 1979.

[19] J. VAN OOSTEN, *A semantical proof of de Jongh's theorem*. ***Archive for Mathematical Logic***, vol. 31 (1991), no. 2, pp. 105–114.

[20] R. PASSMANN, *De Jongh's theorem for intuitionistic Zermelo–Fraenkel set theory*, ***28th EACSL Annual Conference on Computer Science Logic, CSL 2020*** (M. Fernández and A. Muscholl, editors), LIPIcs, vol. 152, Schloss Dagstuhl—Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Saarbrücken/Wader, 2020, pp. 33:1–33:16.

[21] M. RATHJEN, *Choice principles in constructive and classical set theories*, ***Logic Colloquium '02*** (Z. Chatzidakis, P. Koepke, and W. Pohlers, editors), Lecture Notes in Logic, vol. 27, Cambridge University Press, Cambridge, 2006, pp. 299–326.

[22] ———, *The formulae-as-classes interpretation of constructive set theory*, ***Proof Technology and Computation*** (H. Schwichtenberg and K. Spies, editors), NATO Science Series, III: Computer and Systems Sciences, vol. 200, IOS, Amsterdam, 2006, pp. 279–322.

[23] L. H. THARP, *A quasi-intuitionistic set theory*, this JOURNAL, vol. 36 (1971), pp. 456–460.

[24] A. S. TROELSTRA and D. VAN DALEN, ***Constructivism in Mathematics: An Introduction,*** *vol. II*, Studies in Logic and the Foundations of Mathematics, vol. 123, North-Holland, Amsterdam, 1988.

[25] A. VISSER, *Rules and arithmetics*. ***Notre Dame Journal of Formal Logic***, vol. 40 (1999), no. 1, pp. 116–140, Special issue in honor and memory of George S. Boolos.

INSTITUTE FOR LOGIC LANGUAGE AND COMPUTATION, FACULTY OF SCIENCE
UNIVERSITY OF AMSTERDAM
P.O. BOX 94242, 1090 GE AMSTERDAM
THE NETHERLANDS
E-mail: r.passmann@uva.nl

https://doi.org/10.1017/jsl.2022.51 Published online by Cambridge University Press