THE DIVERSITY OF CATEGORICITY WITHOUT DELAY

ISKANDER S. KALIMULLIN, ALEXANDER G. MELNIKOV, AND KENG MENG NG.

We continue the systematic study of fully primitive recursive structures (initiated in [7]). Informally, an algebraic structure is fully primitive recursive (f.p.r.) if more of the structure can be computed "now", i.e. without any unbounded delay. Such structures tend to have algorithmically feasible presentations (in the sense of [5]). Eliminating the unbounded search is one of the key difficulties we have to face when trying to transform a Turing computable (constructive) structure (in the sense of [3, 6]) into a polynomial-time one (in the sense of [5, 4]). Our approach is designed to isolate and study this major difficulty. Formally, $\mathcal{B} = (\omega; f_1, \dots, f_k, P_1, \dots, P_n)$ is a f.p.r. presentation of a countably infinite $\mathcal{A} = (A; g_1, \dots, g_k, R_1, \dots, R_n)$ if $\mathcal{B} \cong \mathcal{A}$ and the functions $f_1, \dots, f_k$ and $P_1, \dots, P_n$ are primitive recursive. We note that Alaev [1] has recently suggested a similar approach.

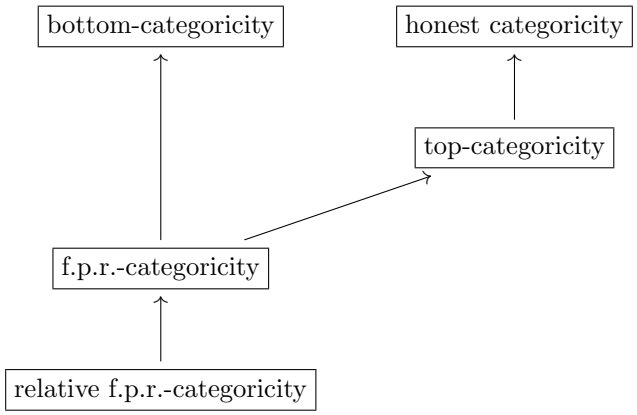
Recall that a countably infinite structure is computably categorical (autostable) if it has a unique (Turing) computable presentation up to computable isomorphism [6, 3]. It is natural to view f.p.r. structures under fully primitive recursive (f.p.r.) isomorphism, i.e. under primitive recursive isomorphism having primitive recursive inverse. A countably infinite structure is **f.p.r.-categorical** if it has a unique f.p.r. presentation up to f.p.r. isomorphism. Remarkably, f.p.r.-categoricity does not imply computable categoricity in general [7], but in common algebraic classes f.p.r.-categorical structures tend to be computably categorical and trivial. Many algebraically natural f.p.r. structures satisfy some weaker property which resembles f.p.r.-categoricity. We list 5 such properties below, each of these 5 properties naturally and frequently occur in standard algebraic classes. Our ultimate goal is to compare these properties, but this is not the main point of this note. It is more important that *properties of this kind have never been seen in computable structure theory*, and the study of these new concepts requires novel ideas and techniques.

Let $\mathbb{FPR}(\mathcal{A})$ be the collection of all f.p.r. presentations of a countably infinite structure $\mathcal{A}$. Note that the inverse of a primitive recursive function does not have to be primitive recursive. For $\mathcal{A}_1, \mathcal{A}_2 \in \mathbb{FPR}(\mathcal{A})$, write $\mathcal{A}_1 \leq_{pr} \mathcal{A}_2$ if there exists a primitive recursive isomorphism from $\mathcal{A}_1$ onto $\mathcal{A}_2$. We say that $\mathcal{A}$ is **bottom-categorical** if $\mathbb{FPR}(\mathcal{A})$ has the $\leq_{pr}$-least element. A typical example of a bottom-categorcial structure is (ω, S) (where $S(x) = x + 1$). The same applies to any other finitely generated structure in a finite functional language. We say that $\mathcal{A}$ is **top-categorical** if $\mathbb{FPR}(\mathcal{A})$ has the $\leq_{pr}$-greatest element. An example of a top-categorcial structure is the dense linear order $(\mathbb{Q}, <)$. The same can be said about the atomless Boolean algebra[1].

[1]Of course $(\mathbb{FPR}(\mathcal{A}), \leq_{pr})$ does not have to have the least of the greatest element. Nonetheless, these two properties seem to be closest to f.p.r.-categoricity among other naturally occurring features of $(\mathbb{FPR}(\mathcal{A}), \leq_{pr})$.

An alternate way of comparing elements in $\mathbb{FPR}(\mathcal{A})$ uses *honest* computable functions. A computable total function is *honest* [8] if its graph is primitive recursive[2]. We say that a fully primitive recursive structure $\mathcal{A}$ is **honestly categorical** if for each $\mathcal{A}_1, \mathcal{A}_2 \in \mathbb{FPR}(\mathcal{A})$ there exists a honest isomorphism from $\mathcal{A}_1$ onto $\mathcal{A}_2$. The associated binary relation of being honestly isomorphic is symmetric, but it is not transitive in general. In some sense honest categoricity is a non-deterministic version of f.p.r. categoricity. Indeed, we can "instantly" decide whether $f(x) = y$, but given x finding $f(x)$ may involve a search through the whole structure. In many standard classes (including e.g. linear orders) honest categoricity is equivalent to the usual computable categoricity.

Finally, the strongest possible notion is that of **relative f.p.r. categoricity** which says that there exists a pair of oracle primitive recursive schemata $\mathcal{P}_+$ and $\mathcal{P}_-$ such that for any isomorphic copy $\mathcal{B}$ (upon ω) of the f.p.r. structure $\mathcal{A}$ the maps $\mathcal{P}_+^{\mathcal{B}} : \mathcal{A} \to \mathcal{B}$ and $\mathcal{P}_-^{\mathcal{B}} : \mathcal{B} \to \mathcal{A}$ are isomorphisms that are inverses of each other[3]. All algebraically natural examples of f.p.r. categorical structures are relatively f.p.r.-categorical [7]. We summarise all notions in one diagram below.



The first main result justifies the diagram.

Theorem 0.1. *All implications in the diagram above are proper. Furthermore, incompatible nodes correspond to independent notions.*

An alternate way to compare f.p.r. structures uses relative computation and the primitive recursive jump $0'_{PR}$ (to be defined). This is the direct analogy with the situation in computable structure theory where $0^{(n)}$-categorical structures have been studied extensively [3, 2].

Theorem 0.2. For every $n > 0$ there exists a fully primitive recursive structure which is fully $0^{(n)}_{PR}$-categorical but not fully $0^{(n-1)}_{PR}$-categorical.

The terminology is self-explanatory and should be clear to any expert in computable structure theory. If the reader is unfamiliar with the standard terminology

[2]According to [8], the complexity of such a function comes from its speed of growth and thus it is "honest".

[3]Here a structure is identified with its open diagram, and an open diagram is identified with its characteristic function. An oracle primitive recursive schema will be defined later.

of the field, they should wait until Section 1.2 where all these terms will be introduced[4].

1. Proofs and proof sketches

1.1. **Proof of Theorem 0.1.** First, note that all the implications illustrated in the diagram are fairly obvious. Theorem 0.1 follows from the four claims below.

Claim 1.1. *In general, f.p.r.-categoricity does not imply relative f.p.r. categoricity.*

Proof of Claim. There exists a f.p.r.-categorical structure that is not computably categorial [7]. Relative f.p.r.-categoricity implies relative computable categoricity and thus computable categoricity as well. □

Claim 1.2. *Bottom categoricity does not imply honest categoricity.*

Proof of Claim. As noted in the introduction, (ω, S) is bottom-categorical. We build two f.p.r. copies of (ω, S), $\mathcal{A}$ and $\mathcal{B}$. To diagonalize against the e'th potential primitive recursive graph, start building both $\mathcal{A}$ and $\mathcal{B}$ to consist of an initial segment and a disjoint region that will temporarily be disconnected from the origin 0 (in both copies). Pick a pair of points (x, y) from these disjoint regions ($x \in \mathcal{A}$ and $y \in \mathcal{B}$) and wait for the e'th primitive recursive algorithm p_e to make a decision. Depending on the outcome, connect the regions to the initial segment so that the opposite holds. □

Claim 1.3. *Top categoricity does not imply bottom categoricity.*

Proof of Claim. Given an arbitrary f.p.r. presentation $\mathcal{A} \cong (\mathbb{Q}, <)$ (which is top-categorical) we build $\mathcal{B}$ and make sure the e'th primitive recursive function $p_e : \mathcal{A} \to \mathcal{B}$ is not an isomorphism. We wait for $p_e(x)$ to halt, and then freeze the interval $[p_e(x), +\infty)$ in $\mathcal{B}$ (i.e., enumerate $\mathcal{B}$ outside this interval). In contrast, make sure at least one extra point is enumerated into the interval $[x, +\infty)$ in $\mathcal{A}$. Wait for p_e to halt on all these points. □

Claim 1.4. *Honest categoricity does not imply top categoricity.*

Proof of Claim. Once the reader sees the main idea they should be able to reconstruct the routine standard technical details (some of which will be omitted).

We build a f.p.r. M in a finite language consisting of unary functional symbols f, g, h. We need to meet

$$R_{e,k} : \ M \cong P_e \cong P_k \implies P_e \cong_{honest} P_k,$$

where $(P_i)_{i\in\omega}$ is the computable listing of all f.p.r. structures (upon initial segments of ω), and $P_e \cong_{honest} P_k$ is self-explanatory. We also need to satisfy

$$S_e : \ M \cong P_e \implies (\exists B_e)(\forall i)\ p_i : B_e \not\to P_e,$$

[4]In fact, in Section 1.2 we will prove more than is stated in Theorem 0.2. More specifically, we will show that the PR-degree any honest function can be realized as the f.p.r.-degree of categoricity of some structure (all undefined terms will be clarified). Remarkably, for any n, the PR-degree of $0^{(n)}_{PR}$ is honest. We also note that the structures witnessing Theorem 0.2 are finitely generated in a finite functional language and thus are bottom-categorical. It is unclear whether there is any non-trivial relationship between fully $0^{(n)}_{PR}$-categoricity and any of the properties of $(\mathbb{FPR}, \leq_{pr})$ described above, but we conjecture that the approach via the primitive jump is mostly independent from the approach taken in Theorem 0.1.

where $(p_i)_{i\in\omega}$ is the computable listing of all primitive recursive functions, and $p_i : B_e \not\to P_e$ indicates that p_i is not an isomorphism from B_e to P_e. For convenience, we further split S_e into sub-requirements $S_{e,i}$, one for each p_i. The sub-requirements $\{S_{e,i}\}_{i\in\omega}$ will share the same B_e which will be built in the construction. The basic strategy for $S_{e,i}$ will be using a *special component* which is labeled by a loop of a specific length. More formally, a sequence of the form $x, f(x), \dots, f^u(x) = x$ where u is a number specifically reserved for the requirement in the construction. For future convenience, using another unary functional symbol g we ensure that $g(y) = x$ for any y in this special f-loop. Apart from the f-loop, the special component will contain an extra auxiliary h-loop (for another unary h) of length 2. This 2-loop will be disjoint from the rest of the special component, i.e. it will not contain $f^i(x)$ for any i) For each z in the auxiliary 2-loop, we define $g(z) = x$. In other words, x is the "root" of the special component, and every other element can be promptly sent to the root via g.

Our structure M will consist of infinitely many special components, one for each requirement. In the construction we may be (temporarily) dealing with a special component having no auxiliary h-loop, this incomplete component will be denoted $\boxed{1}$. The full special component with the auxiliary h-loop will be denoted $\boxed{2}$. (Note the component also depends on e.) We also note that the strategy below will have intermediate substages at which we will be waiting. This waiting will be implemented by infinite unbounded copying a special, smallest component reserved for this copy-delay reason (with the length of the f-loop equal to 2, say, and having no auxiliary h-loop). With the exception of this smallest special component, any other special component will appear in M exactly once.

The strategy for $S_{e,i}$ is:

1. Introduce $\boxed{1}$ into M and $\boxed{2}$ into B_e, but temporarily keep $\boxed{2}$ out of M.
2. Wait for either $p_i : \boxed{2} \to P_e$ to halt or for P_e to illustrate $M \ncong P_e$.
3. Transform $\boxed{1}$ into $\boxed{2}$ in M.

Note that it could be the case that $P_e \ncong M$. If we have a prompt evidence of this fact, we will not transform $\boxed{1}$ into $\boxed{2}$ in M. Also observe that a special component is designed so that we can promptly pass to the "root" of the component from any other node of it (using g). Note that in either case p_e cannot be an isomorphism, and we must see this in a finite number of stages.

The strategy for $R_{e,k}$ is reduced to forcing large loops to appear late in both P_e and P_k. In fact, we work with each P_e (individually). In the construction we will explicitly ensure that an element of index n in P_e cannot be contained in a special component of size larger than $s(e) + n$ (we mean the size of its f-loop), where $s(e)$ is the stage of the construction at which P_e is first monitored. This is done by extra delaying the enumeration of M via promptly cloning the smallest component. As usual, if P_e reveals itself too fast we make sure $P_e \ncong M$ by keeping some sizes of f-loops temporarily or permanently outside of M. This way we can ensure that P_e reveals large f-loops slowly (see [7] for a detailed discussion of this and similar strategies).

The construction is standard. The verification of S_e has already been explained above, we discuss why all $R_{e,k}$ are met. Given P_e and P_k isomorphic to M, we can fix the stages $s(e)$ and $s(k)$ at which P_i and P_j are first attended in the construction. We define an isomorphism $u : P_e \to P_k$ such that the graph of u is primitive

recursive, as follows. If $x \in P_e$ and $y \in P_k$ and we evaluate f on the components for at least $s(e)+s(k)+x+y$ stages we must fully reveal the respective components of x and y. If the components are not isomorphic then we say "no" (i.e., $u(x) \neq y$). If the components are isomorphic but either x and y have different locations in their components or $u(x) \neq y$ has already been found, we also "no". Otherwise we have just discovered that y can be made the image of x and then we say "yes" (thus set $u(x) = y$). □

1.2. **Proof of Theorem 0.2.** Recall that a function $t : \omega \to \omega$ is *honest* if the graph of t is primitive recursive. Given a total function f, we may define f-primitive recursive schema by adding f to the elementary functions in the basis of recursion. A total function g is primitively recursively reducible to a function f ($g \leqslant_{PR} f$) if $g = \Phi^f$ for some f-primitive recursive schema Φ^f. The relations $g \equiv_{PR} f$ and $g <_{PR} f$ as well as the notion of primitive recursive (PR-) degree are defined by the standard way. The PR-degree of a function f is denoted by $\deg_{PR}(f)$. For a total function f let $\{\Phi^f_n\}$ be the Gödel numbering of all f-primitive recursive schemata for functions with one variable. Define the primitive recursive jump f'_{PR} as the function $f'_{PR}(n, x) = \Phi^f_n(x)$. It is easy to check that $f \leqslant_{PR} g \implies f'_{PR} \leqslant_{PR} g'_{PR}$ and $f <_{PR} f'_{PR}$.

Definition 1.5. For a total function f and a f.p.r. structure $\mathcal{A}$ we say that $\mathcal{A}$ is fully f-categorical if for every f.p.r. copy $\mathcal{B} \cong \mathcal{A}$ there is an isomorphism h from $\mathcal{B}$ onto $\mathcal{A}$ such that $h \leqslant_{PR} f$ and $h^{-1} \leqslant_{PR} f$. We say that $\mathcal{A}$ has PR-degree of categoricity $\deg_{PR}(f)$ if for all total g such that $\mathcal{A}$ is fully g-categorical we have $f \leqslant_{PR} g$.

To prove Theorem 0.2 it is sufficient to construct, for every n, an example of a f.p.r. structure whose PR-degree of f.p.r. categoricity is $\emptyset^{(n)}_{PR}$. In fact, we will prove more. We will show that the PR-degree of a honest function is the PR-degree of categoricity of some f.p.r. structure, and we will also show that the PR-degrees $\emptyset'_{PR}, \emptyset''_{PR}, \emptyset'''_{PR}, \ldots$ contain honest functions.

Definition 1.6. A partial function $f(\vec{x})[s]$ is a primitive recursive approximation of a total function $f(\vec{x})$ if

(1) if $f(\vec{x})[s] \downarrow$ then $f(\vec{x})[s] = f(x) \leqslant s$ and $f(\vec{x})[s+1] \downarrow$;
(2) for every $\vec{x}$ there is a stage s such that $f(\vec{x})[s] \downarrow$;
(3) the function

$$f'(\vec{x}, s) = \begin{cases} -1, & \text{if } f(\vec{x})[s] \uparrow; \\ f(\vec{x})[s], & \text{if } f(\vec{x})[s] \downarrow; \end{cases}$$

is primitive recursive.

The function $t(\vec{x}) = \min\{s : f(\vec{x})[s] \downarrow\}$ will be called *the time function* for $f(\vec{x})$.

Note that a function $t(\vec{x})$ is the time function for some function $f(\vec{x})$ with respect to a primitive recursive approximation $f(\vec{x})[s]$ if and only if the function t is honest. Moreover, if $t(\vec{x})$ is honest then $t(\vec{x})$ is a time function for itself with respect to the primitive recursive approximation $t(\vec{x})[s]$ such that $t(\vec{x})[s] \downarrow \iff (\exists y \leqslant s)[(\vec{x}, y) \in \text{graph } t]$.

Lemma 1.7. *Let $f(\vec{x})[s]$ be a primitive recursive approximation for $f(\vec{x})$ with the corresponding time function $t(\vec{x})$. There is a primitive recursive function $i(n)$ such*

that for every n the function $\Phi^t_{i(n)}(x)$ is the time function for $\Phi^f_n(x)$ with respect to the uniformly primitive recursive approximation $\Phi^f_n(x)[s]$ as defined above.

The proof of the lemma above is a straightforward analysis of the recursive definition of $\Phi^f_n(x)$.

Proposition 1.8. *If $t(\vec{x})$ is honest and $i(n)$ is from Lemma 1.7 with $f = t$ then the function $u(n, x) = \Phi^t_{i(n)}(x)$ is honest and $u \equiv_{PR} t'_{PR}$. Therefore, the primitive recursive jump of honest primitive recursive degree is a honest primitive recursive degree.*

Proof. Follows from $u(n, x) = t'_{PR}(i(n), x)$ and $t'_{PR}(n, x) = \Phi^t_n(x)[u(n, x)]$. □

Thus, Theorem 0.2 follows from the proposition below.

Proposition 1.9. *For every honest function t there exists a f.p.r. rigid structure $\mathcal{A}$ such that:*

1. *for every f.p.r. copy $\mathcal{B} \cong \mathcal{A}$ the isomorphism h from $\mathcal{B}$ onto $\mathcal{A}$ is PR-reducible to t;*
2. *for every f.p.r. copy $\mathcal{B} \cong \mathcal{A}$ the isomorphism h from $\mathcal{B}$ onto $\mathcal{A}$ has a primitive recursive inverse h^{-1};*
3. *there is a f.p.r. copy $\mathcal{C} \cong \mathcal{A}$ such that for the isomorphism g from $\mathcal{C}$ onto $\mathcal{A}$ we have $t \leqslant_{PR} g$.*

We also note that the language of the structure is finite and functional.

Proof. The functional signature of the structure $\mathcal{A} = \mathcal{A}_t$ consists of a constant o and two unary functions s and c. The universe of the structure is a disjoint union of c-cycles of finite length $C^0, C^1, C^2, \dots$ The s-function maps all elements of the cycle C^k to a fixed element $o_{k+1} \in C^{k+1}$. The values of the s-function together with o form the *ω-chain*

$$o_0 = o, o_1 = s(o), o_2 = s(s(o)), \dots, o_k = s^k(o), o_{k+1} = s^{k+1}(o), \dots .$$

For an element $x \in \mathcal{A}$ let $N(x)$ denote the least number $s > 0$ such that $c^s(x) = x$. Then let $\ell(k) = N(s^k(o))$ be the size of the c-cycle containing the k-th element of the ω-chain $s^k(o)$. It is clear that the function $\ell : \omega \to \omega \setminus \{0\}$ uniquely determines the isomorphic type of $\mathcal{A}$. The structure $\mathcal{A}$ has a f.p.r. copy if and only if the function ℓ is honest. Furthermore, if ℓ is honest we can identify $\mathcal{A}$ with its *canonical* f.p.r. copy (ω, o, s, c) where we can primitive recursively calculate $a(k)$ and $b(k) < \ell(a(k))$ such that $k = c^{b(k)}(s^{a(k)}(o))$ for every $k \in \omega$. Then for every f.p.r. copy $\mathcal{B}$ of $\mathcal{A}$ the isomorphism h^{-1} from $\mathcal{A}$ onto $\mathcal{B}$ is primitive recursive. Let $\mathcal{C}$ be an f.p.r. copy of $\mathcal{A}$ in which the odd numbers are reserved for the elements of $c^{\ell(k)-1}(s^k(o))$. Namely we have $2n + 1 = c^{\ell(n)-1}_{\mathcal{C}}(s^n_{\mathcal{C}}(o_{\mathcal{C}}))$ for every n. Then for the isomorphism g from $\mathcal{C}$ onto $\mathcal{A}$ we have $\ell(n) = b(g(2n + 1)) + 1$ for every $n \in \omega$, so that $\ell \leqslant_{PR} g$. We need to satisfy the requirements

$$P_n : \mathcal{B}_n \cong \mathcal{A} \implies (\exists h_n)[h_n \text{ — p.r. isomorphism from } \mathcal{B}_n \text{ onto } \mathcal{A}],$$

where $\mathcal{B}_n = (\omega, o_n, s_n, c_n)$ is the n-th f.p.r. structure. The functions s_n and c_n given by their primitive recursive schemas. Then s_n and c_n have primitive recursive time functions. Let $N_n(x)$ be the least number $s > 0$ such that $c^s_n(x) = x$. To meet P_n we define the structure $\mathcal{A}$ in such a way that $\ell(2^{n+1}(2m+1)) \in \{3n+1, 3n+2\}$. The

choice between $3n+1$ and $3n+2$ depends only on the behaviour of the f.p.r. structure $\mathcal{B}_n$.

We define the primitive recursive function $g(n,m)$ as the greatest $k \leqslant m$ such that for all $x < k$

(1) the primitive recursive approximations $c_n^j(s_n^i(x))[m]$ are defined for every $i \leqslant 2^{n+2}(2x+2)$ and $j \leqslant 3n+2$;
(2) there is a nonzero $i \leqslant 2^{n+2}$, such that $N_n(s_n^i(x)) \in \{3n+1, 3n+2\}$; and
(3) for the least nonzero $i \leqslant 2^{n+2}$ from (2) there are u and v such that $u \leqslant x+1$, $0 < v \leqslant x$ and $N_n(s_n^{i+2^{n+2}u}(x)) = N_n(s_n^{i+2^{n+2}(u+v)}(x)) = 3n+1$.

Now let

$$f(n,m) = \begin{cases} 3n+1, & \text{if } f(n,m') = 3n+2 \text{ for all } m' \in \Big[m \dot{-} g(n,m), m\Big); \\ 3n+2, & \text{otherwise.} \end{cases}$$

Let $\mathcal{A}$ be the canonical structure (ω, o, s, c) for the honest size function

$$\ell(k) = \begin{cases} 3t(n)+3, & \text{if } k = 2n+1; \\ f(n,m), & \text{if } k = 2^{n+1}(2m+1). \end{cases}$$

Note that $\ell \equiv_{PR} t$. Fix an f.p.r. copy $\mathcal{C}$ of $\mathcal{A}$ such that $2n+1 = c_{\mathcal{C}}^{\ell(n)-1}(s_{\mathcal{C}}^n(o_{\mathcal{C}}))$ for every n. Then by the arguments above for the isomorphism g from $\mathcal{C}$ onto $\mathcal{A}$ we have $\ell \leqslant_{PR} g$ and, hence, $t \leqslant_{PR} g$.

To prove that each requirement P_n is satisfied suppose that $\mathcal{B}_n = (\omega, o_n, s_n, c_n) \cong \mathcal{A} = (\omega, o, s, c)$. Then $\lim_m g(n,m) = \infty$. Indeed, if $\lim_m g(n,m) = x < \infty$ then for the integer x one of conditions (2) and (3) does not hold. But (2) should hold for every x since $2^{n+1}(2(m+1)+1) - 2^{n+1}(2m+1) = 2^{n+2}$ and $\ell(2^{n+1}(2m+1)) \in \{3n+1, 3n+2\}$. If (3) fails for x then there exists some $u \leqslant x+1$ such that $N_n(s_n^{2^{n+2}v}(s_n^{i+2^{n+2}u}(x))) = N_n(s_n^{i+2^{n+2}(u+v)}(x)) = 3n+2$ for every $v \leqslant x$. Since $\mathcal{B}_n \cong A$ we have $y = s_n^{i+2^{n+2}u}(x) = s_n^{2^{n+1}(2m+1)}(o_n)$ for some m. Then for every $v \leqslant x$ $3n+2 = N_n(s_n^{2^{n+2}v}(y)) = N_n(s_n^{2^{n+1}(2(m+v)+1)}(o_n))$, and thus $\ell(2^{n+1}(2(m+v)+1)) = f(n, m+v) = 3n+2$ for every $v \leqslant x$. This is impossible since by the definition of f for every $m \in \omega$ there is an $v \leqslant x = \lim_m g(n,m)$ such that $f(n, m+v) = 3n+1$. To calculate $h_n(x)$ it is enough to find integers a and b such that $x = c_n^a(s_n^b(o_n))$. At first we use time functions for c_n and s_n to get an integer m such that $g(n,m) > x$. By (2) and (3) there are $i < 2^{n+2}$, $u \leqslant x+1$ and $0 < v \leqslant x$ such that $N_n(s_n^{i+2^{n+2}u}(x)) = N_n(s_n^{i+2^{n+2}(u+v)}(x)) = 3n+1$. Then the element $y = s_n^{i+2^{n+2}u}(x)$ should have the form $y = s_n^{2^{n+1}(2s+1)}(o_n)$, and the element $z = s_n^{i+2^{n+2}(u+v)}(x)$ should have the form $z = s_n^{2^{n+1}(2(s+v)+1)}(o_n)$. Therefore, we have

$$\ell(2^{n+1}(s+1)) = \ell(2^{n+1}(2(s+v)+1)) = f(n,s) = f(n,s+v) = 3n+1.$$

By the definition of f we have $s \notin \left[s+v \dot{-} g(n,s+v), s+v\right)$ and, hence, $g(n,s+v) < v$. Then $g(n,s) \leqslant g(n,s+v) < v \leqslant x < g(n,m)$, so that $s < m$ and, hence, $b < m$. Thus, $h_n(x) = c^a(s^b(o))$ for integers a and b such that $a < \ell(b)$, $b < m$ and $x = c_n^a(s_n^b(o_n))$. This shows that $h_n \leqslant_{PR} \ell \leqslant_{PR} t$. □

References

[1] P. Alaev. Existence and uniqueness of polinomial-time presentations of structures. *Submitted*.

[2] C. Ash. Recursive labeling systems and stability of recursive structures in hyperarithmetical degrees. *Trans. Amer. Math. Soc.*, 298:497–514, 1986.

[3] C. Ash and J. Knight. *Computable structures and the hyperarithmetical hierarchy*, volume 144 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam, 2000.

[4] Douglas Cenzer, Rodney G. Downey, Jeffrey B. Remmel, and Zia Uddin. Space complexity of abelian groups. *Arch. Math. Log.*, 48(1):115–140, 2009.

[5] Douglas A. Cenzer and Jeffrey B. Remmel. Polynomial-time versus recursive models. *Ann. Pure Appl. Logic*, 54(1):17–58, 1991.

[6] Y. Ershov and S. Goncharov. *Constructive models*. Siberian School of Algebra and Logic. Consultants Bureau, New York, 2000.

[7] I. Kalimullin, A. Melnikov, and K.M. Ng. Algebraic structures computable without delay. *Submitted*.

[8] L. Kristiansen. *Papers on Subrecursion Theory, Dr Scient Thesis, Research report 217*. PhD thesis, University of Oslo, 1996.