Sub-Computabilities

Fabien Givors, Grégory Lafitte

Sub-Computabilities

Fabien Givors and Gregory Lafitte*

Laboratoire d'Informatique Fondamentale de Marseille (LIF),
CNRS – Aix-Marseille Université,
39, rue F. Joliot-Curie, 13453 Marseille Cedex 13, France
{Fabien.Givors,Gregory.Lafitte}@lif.univ-mrs.fr

Abstract. Every recursively enumerable set of integers (r.e. set) is enumerable by a primitive recursive function. But if the enumeration is required to be one-one, only a proper subset of all r.e. sets qualify. Starting from a collection of total recursive functions containing the primitive recursive functions, we thus define a *sub-computability* as an enumeration of the r.e. sets that are themselves one-one enumerable by total functions of the given collection. Notions similar to the classical computability ones are introduced and variants of the classical theorems are shown. We also introduce sub-reducibilities and study the related completeness notions. One of the striking results is the existence of natural (recursive) sets which play the role of low (non-recursive) solutions to Post's problem for these sub-reducibilities. The similarity between sub-computabilities and (complete) computability is surprising, since there are so many missing r.e. sets in sub-computabilities. They can be seen as toy models of computability.

Introduction

Many proofs in (basic and also more involved) computability rely on the algebraic structure of the enumeration of r.e. sets, partial functions, *etc.*, and not really *per se* on the notion of "being computable". The structure is provided by the properties of an acceptable enumeration, and consequences of being acceptable, *e.g.*, Kleene's second recursion theorem. One motivation behind the work of this article is to develop results similar to the classical computability ones (from basic results to Turing completeness issues) but in a setting verifying only a proper subset of the elementary properties of classical computability. In our case, this translates as the quest of developing computabilities with most of the nooks and crannies of classical computability without being all of computability. Here, a computability is meant to be collections of sets and functions which portray what we consider in this setting to be the r.e. sets and partial computable functions. Sub-computabilities are computabilities where these collections are a proper subset of the classical ones and is the focus of this article. Higher

* The research presented in this paper has been made possible by the support of the French ANR grants *NAFIT* (ANR-08-DEFIS-008-01) and *EMC* (ANR-09-BLAN-0164-01).

computability could also be cast in this setting and it will be the subject of a subsequent article. Our setting can be seen as a toy model for computability, not based on models of machines, but on the algebraic structure of computability.

A sub-computability c is a pair $(\mathbf{\Phi}^{\mathsf{c}}_{\cdot},\mathbf{W}^{\mathsf{c}}_{\cdot})$ of enumerations of a sufficiently closed collection $\mathbf{\mathfrak{F}}_{\mathsf{c}}$ (called the foundation of c) of total recursive functions (called c-*fundamental*), and of a collection $\mathbf{\mathfrak{E}}_{\mathsf{c}}$ (called the support of c) of r.e. sets (called c-*enumerable*), which are one-one enumerated[1] by functions in $\mathbf{\mathfrak{F}}_{\mathsf{c}}$. These fundamental functions somehow measure the effectiveness of the constructions used in computability. A partial recursive function is said to be *somewhat* c-*computable* if its graph is c-enumerable. Building on an enumeration $\mathbf{\Phi}^{\mathsf{c}}_{\cdot}$, we can respectively build canonical enumerations $\mathbf{W}^{\mathsf{c}}_{\cdot}$ (resp. $\varphi^{\mathsf{c}}_{\cdot}$) of c-enumerable sets (resp. somewhat c-computable functions). Since the collection of c-enumerable sets (and somewhat c-computable functions) is completely and uniquely determined by the collection $\mathbf{\mathfrak{F}}_{\mathsf{c}}$, we will identify c with the foundation ($\mathsf{c} = \mathbf{\mathfrak{F}}_{\mathsf{c}}$). What could vary is the enumerations of $\mathbf{\mathfrak{F}}_{\mathsf{c}}$ and $\mathbf{\mathfrak{E}}_{\mathsf{c}}$, but we will see with the isomorphism theorem *à la* Rogers (Theorem 5) that there is no confusion here.

An example of a sub-computability is p, the r.e. sets one-one enumerated by primitive recursive functions. Koz'minyh [2] in 1972 proved a key lemma on p. We generalize this lemma (Lemma 1) to any sub-computability. It gives insights into the behavior of c-enumerable sets and somewhat c-computable functions. Its corollaries make classical constructions possible, even if we do not have the general μ recursion operator. The (primitive recursive) effectivity of these corollaries (and of most of our theorems) is especially useful.

Other sub-computability foundations (and thus sub-computabilities) stem from the field of *subrecursion*. It provides a great number of examples of closed collections of ever more increasing total functions which do not contain all of the total recursive functions. This study is strongly entangled with proof theory and provides a clear picture of "what is provable and what is not" in a theory for which one is able to construct an *ordinal analysis*. In particular, proof theory has identified for many theories T the set of total recursive functions whose totality is provable in T. This connection gives another motivation to our work.

Studying sub-computabilities amounts to classifying r.e. sets by measuring the difficulty of enumerating them by way of subrecursion. On that quest, one stumbles with the well-known at first surprising result that all r.e. sets are enumerable by primitive recursive functions. But if the enumeration is required to be one-one, the triviality evaporates: some r.e. sets are not one-one enumerable by primitive recursive functions, *e.g.*, the Ackermann function's range, but still many r.e. sets are primitively one-one enumerable, *e.g.*, the graph of the characteristic function of the range of the Ackermann function or the graph of a universal function.

If a set of integers is enumerable in a sub-computability, then it is recursively enumerable. If it is not enumerable, then it is either not recursively enumerable, or not feasibly enumerable in this sub-computability and necessitates more *power* to be seen as effectively enumerable. Thus, a sub-computability is an ap-

[1] The complete definition (Def. 2) also incorporates finite sets and $\varnothing$.

proximation of the classical computability: we have the same kind of results than in computability but with variations on the notions used; Classical (complete) computability is the *union* of all sub-computabilities.

For example, Post's theorem —stating that a set is recursive if and only if it is both recursively enumerable and co-recursively enumerable— does not hold anymore, leading the way to more restrictive notions of "being recursive". Another example is Kleene's second recursion theorem, which only partially holds in various ways in our sub-computabilities (Theorem 3).

We also define reducibilities which are refinements of Turing (or stronger) reducibilities and yield a structure of the associated degrees that looks very similar to the classical structure of the Turing degrees. Two of the interesting aspects of this study are the simplicity of some of the proofs of the degree structure, and the existence of natural (recursive) sets which play the role of low (non-recursive) solutions to Post's problem for these sub-reducibilities. To have this setting as a *real* toy model for computability, it would be nice to be able to have ways of transferring those results back in classical computability.

Our study is different from other *subrecursive degree* theories, especially when considering the fact that our objects of study are not necessarily recursive. Nevertheless, one of our future goals is to build bridges with these theories, *e.g.*, *honest elementary degrees* (see [3–5]), to be able to use their techniques and have their applications, especially the minimal pairs result with a proof-theoretical twist in [6].

1 Sub-computability basics

We consider a *sub-computability* $\mathtt{C}$ to be a pair $(\mathbf{\Phi}^{\mathtt{C}}_{\cdot},\mathbf{W}^{\mathtt{C}}_{\cdot})$ of enumerations of the *foundation* $\mathfrak{F}_{\mathtt{C}}$ and of the *support* $\mathfrak{E}_{\mathtt{C}}$ of the sub-computability. A support $\mathfrak{E}_{\mathtt{C}}$ is a collection of r.e. sets (the $\mathtt{C}$-*enumerable* sets). It is constituted by sets one-one enumerated by total recursive functions (the $\mathtt{C}$-*fundamental* functions) belonging to $\mathfrak{F}_{\mathtt{C}}$. Since the $\mathfrak{E}_{\mathtt{C}}$ is uniquely determined by this foundation, $\mathtt{C}$ will denote both the sub-computability and the foundation. It will always be easy to distinguish between those two uses of notation.

We start with the definition of a *sub-computability foundation*.

Definition 1. A *sub-computability foundation* $\mathtt{C}$ is a set of recursive total functions, called the *fundamental* functions, containing (at least) the primitive recursive initial functions, closed by composition and primitive recursion, along with a coding scheme providing fundamental-functions indices in $\mathbb{N}$ for the fundamental functions, such that we can primitively compute a recursive-functions index from any fundamental-functions index, and such that the characteristic function of the set $\{x : x \text{ is an index for fundamental } f\}$ is primitive recursive, with a primitive recursive padding function $\mathfrak{p}$ ($\mathfrak{p}^n(x)$ gives an index for the fundamental function for which x is an index, which is different from all n *previous* indices, $x, \mathfrak{p}(x), \ldots, \mathfrak{p}^{n-1}(x)$), and with a primitive recursive composition function[2] $\mathfrak{C}$.

[2] $\mathfrak{C}(x, y)$ gives an index for the fundamental function which is the composition of the two fundamental functions for which x and y are respectively indices.

$\mathsf{c}[F]$ designates $\mathsf{c} \cup F$ (the foundation c to which the functions of F are added) closed under composition and primitive recursion. $\mathsf{c}[f]$, $\mathsf{c}[f, g]$, *etc.*, designate respectively $\mathsf{c}[\{f\}]$, $\mathsf{c}[\{f, g\}]$, *etc.*

$\mathbf{\Phi}^{\mathsf{c}}_e(x)$ designates the computation on x of the c-fundamental function whose index is e.

On top of foundations, we build *sub-computabilities*.

Definition 2. A set of integers X is a c-*enumerable* set if there is a function $f \in \mathsf{c}$ such that the maximal initial one-one part[3] of f enumerates X.

A *sub-computability* is a pair $(\mathbf{\Phi}^{\mathsf{c}}_{\cdot}, \mathbf{W}^{\mathsf{c}}_{\cdot})$ of enumerations of a foundation c and of a collection $\mathbf{\Xi}_{\mathsf{c}}$ of recursively enumerable sets, the c-enumerable sets. $\mathbf{\Xi}_{\mathsf{c}}$ is called the *support* of the sub-computability. c will also denote the sub-computability.

In the following, we will denote by p the collection of primitive recursive functions and the sub-computability based on this foundation.

A partial function is *somewhat* c-*computable* if its graph is c-enumerable.

A partial somewhat c-computable function f is said to converge on x if $f(x)$ is defined. It is denoted by $f(x)\downarrow$. Otherwise, it is said to diverge on x and is denoted by $f(x)\uparrow$.

e is a c-*index* for a c-enumerable set W if e is a fundamental index for a c-fundamental function that one-one maximally initially enumerates W. We denote W by $\mathbf{W}^{\mathsf{c}}_e$.

e is a c-*index* for a somewhat c-computable function f if e is a c-index for the graph of f. We denote $f(x)$ by $\varphi^{\mathsf{c}}_e(x)$.

Notice that the honest functions (see for example [3]) are somehow defined in a similar way, but with an elementary foundation (which does not make possible enumerations of non recursive graphs). Notice also that there are indices for the empty set (and thus the nowhere-defined function).

Without the c, $(W_e)_{e\in\mathbb{N}}$ denotes the classical enumeration of recursively enumerable sets, and $(\varphi)_{e\in\mathbb{N}}$ the classical enumeration of recursive functions.

Definition 3. A *universal* function $\mathbf{U}_{\mathsf{c}}$ over a sub-computability support c is a partial function $(c, x) \mapsto \mathbf{U}_{\mathsf{c}}(c, x)$ such that for each somewhat c-computable function φ^{c}_e, for all x, if $\varphi^{\mathsf{c}}_e(x)\uparrow$, then $\mathbf{U}_{\mathsf{c}}(e, x)\uparrow$, else $\mathbf{U}_{\mathsf{c}}(e, x)\downarrow$ and $= \varphi^{\mathsf{c}}_e(x)$.

A c-*fundamentally universal* function $\mathbf{U}^{\mathbf{\Phi}}_{\mathsf{c}}$ is a function $(c, x) \mapsto \mathbf{U}^{\mathbf{\Phi}}_{\mathsf{c}}(c, x)$ such that for each c-fundamental function $\mathbf{\Phi}^{\mathsf{c}}_e$, for all x, $\mathbf{\Phi}^{\mathsf{c}}_e(x) = \mathbf{U}^{\mathbf{\Phi}}_{\mathsf{c}}(e, x)$.

As we will see later, a universal function over a sub-computability support c will *surprisingly* be somewhat c-computable. The notation $\mathbf{U}_{\mathsf{c}}$ will at the same time designate the afore defined partial function $\mathbf{U}_{\mathsf{c}}$ and its c-index.

---

[3] f is said to *one-one maximally initially enumerate* (in short, *one-one enumerate*) X if $X = \begin{cases} f\text{“}Y & \text{if } Y = \mathbb{N}, \\ f\text{“}Y \setminus \{f(\max(Y)\} & \text{otherwise, that is } Y \text{ is finite,} \end{cases}$ where Y is the greatest initial subset of $\mathbb{N}$ (Y is equal to $\mathbb{N}$ or to some $n \in \mathbb{N}$) such that $f \upharpoonright Y$ is one-one.

Definition 2 involving discarding the last enumerated element of the finite set is a better choice than having the empty set represented artificially because we want the set of indices of nowhere-defined somewhat c-computable functions to be as less[4] c-*computable* as possible.

Our enumerability notion sticks well with the intuition adopted in classical computability: a set is c-enumerable if and only if it is the domain of a somewhat c-computable function. This property and Definition 2 call for the following result: a c-enumerable set W, for which x is a c-index, has infinitely many other indices computable from x in a p-fundamental computable way. To prove it, we naturally use the padding function $\boldsymbol{\mathfrak{y}}$ for the indices of the c-fundamental functions. The s-m-n theorem also still holds for fundamental functions.

Universal functions are absent of the fundamental functions. Therefore, even if we have s-m-n, we cannot prove Kleene's second recursion theorem for fundamental functions. It will however partially hold for somewhat c-computable functions (Theorem 3), for which s-m-n does not hold anymore.

Koz'minyh [2] in 1972 proved the following lemma (for the p part), which constitutes a mile stone for understanding sub-computabilities. We call it the *heredity lemma*.

Lemma 1. *If W_e is an infinite c-enumerable set, and $W_i \supset W_e$ is recursively enumerable, then W_i is c-enumerable.*

The following useful corollaries come to mind when proving this lemma. First, there exists a primitive recursive function $\mathbf{H}_{\mathsf{c}}$ such that $\mathbf{W}^{\mathsf{c}}_{\mathbf{H}_{\mathsf{c}}(\langle i,j\rangle)} = \mathbf{W}^{\mathsf{c}}_i \cup \mathbf{W}^{\mathsf{c}}_j$. Also, if X is a c-enumerable set and g a c-fundamental function, then the following function is somewhat c-computable: $f(\langle x, y\rangle) = g(\langle x, y\rangle)$ if $x \in X$, $\uparrow$ otherwise.

The first idea to build sub-computabilities is to define a set of total recursive functions, called *fundamental*. From these total functions, we can define a collection of sets, called *support*, that can each be enumerated in a non-repetitive way[5]. Finally, thinking of this support as a collection of graphs of partial functions gives rise to a notion of *somewhat* computable functions. We review these notions below.

c-fundamental functions As they will be our raw material, it is important to see what are their capabilities, and limitations. In general, there will not be any universal function among the fundamental functions. However, thanks to the Normal Form Theorem, we know we can primitively simulate any finite number of step of any recursive function φ_e: there is a *primitive recursive*, function $\mathrm{sim}_{\mathsf{p}}$ such that $\mathrm{sim}_{\mathsf{p}}(e, i, s) = \langle e, i, \varphi_e(i)\rangle + 1$ for at most a unique s, and 0 otherwise.

[4] It is not χ-c-computable, but nonetheless c-enumerable and weakly-c-computable. (See Def. 4.)

[5] As indicates the Normal Form theorem, every recursively enumerable set is enumerable by a primitive recursive function. For example, $f_e : \langle n, s\rangle \mapsto \varphi_e(n)$ if it converges in less than s steps, $\varphi_e(0)$ otherwise. However, such a function will not be one-one in general.

c-*computable sets* The recursiveness of a set in classical computability has many characterizations. These definitions are no longer equivalent in sub-computabilities.

Definition 4. Let E be a set of integers, E is weakly c-computable if E and $\overline{E}$ are c-enumerable. E is somewhat c-computable if χ_E is somewhat c-computable. E is strongly c-enumerable if E is c-enumerable, enumerated by an increasing one-one c-fundamental function $\mathbf{\Phi}^{\mathsf{c}}_e$. E is strongly c-computable if E and $\overline{E}$ are strongly c-enumerable and χ-c-computable if E has a c-fundamental characteristic function.

It is easy to see that each of these *sub-recursive* notions implies classical computability: for a set, being strongly c-enumerable, strongly c-computable, weakly c-computable, somewhat c-computable or χ-c-computable implies being recursive. There are also straightforward implications between these sub-recursive notions: if E is a strongly c-enumerable (resp. strongly c-computable) set, then E is c-enumerable (resp. weakly c-computable) and χ-c-computable. Then, if E is strongly c-computable then E is weakly c-computable and χ-c-computable. If E is a weakly c-computable set or a χ-c-computable set, then E is somewhat c-computable. Hence, all our sub-recursive notions for sets imply being somewhat c-computable.

Notice that, as in the classical setting, every infinite c-enumerable set A contains an infinite strongly c-enumerable set E. Moreover, E is χ-c-computable.

Somewhat c-*computable functions* First, recall that a partial function is *somewhat* c-*computable* if its graph is c-enumerable. Using our *simulation* function $\mathsf{sim}_{\mathsf{p}}$, it is not too difficult to create a one-one enumeration of the graph of an universal function for all recursive functions. Hence, it is easy to prove, as a corollary of the heredity lemma (Lemma 1), that there is a somewhat p-computable function $\mathbf{U}_{T,\mathsf{p}}$ universal for all recursive functions. The following theorem legitimates the notion of an effective enumeration of somewhat c-computable functions:

Theorem 2. *There exists a somewhat* p*-computable universal function* $\mathbf{U}_{\mathsf{c}}$ *over* c.

Remarkable functions There is a universal function among the somewhat c-computable functions. However, not all recursive functions belong to that class. As expected, the Ackermann function is not part of the fundamental functions of our primitive computability. It is not even somewhat p-computable. This result may seem contradictory with Theorem 2. It just shows again that enumerating all functions in one is easier than enumerating some. It also shows that the somewhat computable functions are not closed by composition. Nevertheless, the Ackermann function range is χ-p-computable, because its inverse does belong to p. Thus, the Ackermann function is linked to somewhat p-computable functions: if a recursive function f is somewhat p-computable, then it is less than the Ackermann function on an infinite subset of the domain.

Halting problem for different classes of function of sub-computabilities The halting problem is central in classical computability and is linked to the diagonal set $\mathcal{K} = \{e : \varphi_e(e)\downarrow\}$. In our sub-computabilities, we can define a couple of different $\mathcal{K}$'s depending on the set of functions we are interested in (all recursive functions: $\mathcal{K} = \{e : \varphi_e(e)\downarrow\}$, somewhat computable (partial) functions: $\mathcal{K}_{\mathsf{c}} = \{e : \mathcal{U}_{\mathsf{c}}(e,e)\downarrow\} = \{e : \varphi^{\mathsf{c}}_e(e)\downarrow\}$, fundamental functions: $\mathcal{K}^{\Phi}_{\mathsf{c}} = \{e : \mathcal{U}^{\Phi}_{\mathsf{c}}(e,e) > 0\} = \{e : \Phi^{\mathsf{c}}_e(e) > 0\}$). Notice that the *hardness* of the different halting problems gives more sense to the distinctions between our sub-computability notions.

$\mathcal{K}$ is p-enumerable but not strongly p-enumerable. For every sub-computability c, $\overline{\mathcal{K}}$ is not c-enumerable, the set $\mathcal{K}^{\Phi}_{\mathsf{c}}$ is weakly p-computable but not χ-c-computable and the set $\mathcal{K}_{\mathsf{c}}$ is p-enumerable but not strongly p-enumerable, nor recursive and $\overline{\mathcal{K}_{\mathsf{c}}}$ is not c-enumerable. Finally, the diagonal set on somewhat functions is complete: $\mathcal{K}_{\mathsf{c}} \equiv_m \mathcal{K}$. The proof[6] of the Turing (many-one really) completeness uses the Half recursion theorem (Theorem 3).

Strong c-enumerability implies χ-c-computability, thus we have that there exists a weakly c-computable but not strongly c-enumerable set. We summarize the properties of these sets in table 1.

	r.e	co-r.e	p-r.e	s-c-r.e	co-c-r.e	co-s-c-r.e	w-c-comp	s-c-comp	χ-c-comp
$\mathcal{K}$	✓	×	✓	×	×	×	×	×	×
$\mathcal{K}^{\Phi}_{\mathsf{c}}$	✓	✓	✓	×	✓	×	✓	×	×
$\mathcal{K}_{\mathsf{c}}$	✓	×	✓	×	×	×	×	×	×

Table 1: Properties in c of particular sets

c-computabilities' specimens It is possible to exhibit an infinite family of different c-computabilities using a natural extension of the primitive recursive recursion: α-recursion[7]. The different growing hierarchies give us an easy way to exhibit functions that require a given α-recursion operator. For example, the Ackermann function does not belong to p, but it belongs to[8] $\mathsf{c}_{\omega^\omega}$. And Goodstein's sequences

[6] Let a be the c-fund. index of a nowhere null function. Show that there is $f_x \in \mathsf{c}$ such that $\varphi^{\mathsf{c}}_{f_x(e)} : y \mapsto \begin{cases} \varphi_x(x) & \text{if } y = a \text{ or } y = e \\ 0 & \text{otherwise.} \end{cases}$. Use the set $A = \left\{\mathfrak{y}^{(n)}(z) : z \geqslant 1\right\}$ to apply the Half recursion part of Theorem 3 and effectively obtain a fixed-point n of f_x. Notice that $\varphi^{\mathsf{c}}_{f_x(n)} \cong \varphi^{\mathsf{c}}_n$ since $a \notin A$. Then, $n \in \mathcal{K}_{\mathsf{c}}$ iff $x \in \mathcal{K}$, for n c-computable from x.

[7] Let $\langle A, \lhd\rangle$ an elementary ordinal representation system (EORS) and $\alpha \in A$ such that $\overline{0} \lhd \alpha$. A function f on natural numbers is called *α-recursive* if it can be generated like the primitive recursive functions albeit plus the following operation:

$$f(m,\boldsymbol{n}) = \begin{cases} h(m,\boldsymbol{n},f(\theta(m,\boldsymbol{n}),\boldsymbol{n})) & \text{if } \overline{0} \lhd m \lhd \alpha \text{ and } \theta(m,n) \lhd m, \\ g(m,\boldsymbol{n}) & \text{otherwise,} \end{cases}$$

where g,h,θ are α-recursive. See [1, 10] for more details.

[8] Denote by c_α the set of total α-recursive functions based on EORS' provided by proof theory.

function will only appear in c_{ϵ_0}. Using fundamental sequences, we can easily design other such examples up to Γ_0 thanks to Veblen's hierarchy (See [12]) and even beyond.

2 Common sub-computability

Apart from the particular functions (like the diagonal Ackermann function[9] Ack or the ones built in the previous section), that dominate every c-fundamental functions, which we have for certain sub-computabilities c, there is a simple way to build for every c a total recursive function which is not c-fundamental. The function $x \mapsto \mathbf{U}_{\mathsf{c}}^{\mathbf{\Phi}}(x,x)$ is a total recursive function that is not c-fundamental, but is somewhat p-computable. Also, apart from the diagonal set adapted to c-computability, $\mathbf{K}_{\mathsf{c}}$, we can define c-versions of the *busy beaver* functions of Tibor Radó: $\mathbf{BB}_{\mathsf{c}}(x) = \max\{\varphi_i^{\mathsf{c}}(0) : i \leqslant x\}$ and $\mathbf{BB}_{\mathsf{c}}^{\mathbf{\Phi}}(x) = \max\{\mathbf{\Phi}_i^{\mathsf{c}}(0) : i \leqslant x\} + x$. The classical results and proofs carry through: $\mathbf{BB}_{\mathsf{c}}$ is not somewhat c-computable. Moreover, $\overline{\text{graph}(\mathbf{BB}_{\mathsf{c}})}$ is not c-enumerable. $\mathbf{BB}_{\mathsf{c}}^{\mathbf{\Phi}}$ is recursive, but not somewhat c-computable.

There is a natural superset of the foundation c of a sub-computability c: Denote by $\copyright$ the sub-computability of foundation $\mathsf{c}[\mathbf{BB}_{\mathsf{c}}^{\mathbf{\Phi}}]$. Notice that for any sub-computability c, $f \in \mathsf{c}$ does not necessarily[10] imply $f^{-1} \in \mathsf{c}$. Notice also that, for any sub-computability c, all the functions in $\copyright$ stay recursive. This operation can be seen as some kind of jump since it provides a new sub-computability $\copyright$ in which there is a fundamental universal function for c-fundamental functions: There is a $\copyright$-fundamental universal $\mathbf{U}_{\mathsf{c}}^{\mathbf{\Phi}}$ function for c-fundamental functions. As a corollary, we have that $\chi_{\mathbf{K}_{\mathsf{c}}^{\mathbf{\Phi}}} \in \mathsf{c}[\mathbf{BB}_{\mathsf{c}}^{\mathbf{\Phi}}]$ but $\mathbf{BB}_{\mathsf{c}}^{\mathbf{\Phi}} \notin \mathsf{c}[\chi_{\mathbf{K}_{\mathsf{c}}^{\mathbf{\Phi}}}]$.

It is easy to see that the fundamental functions of $\textcircled{\mathsf{p}}$ are exactly the functions of $\mathsf{p}[\text{Ack}]$. First, notice that Ack is in $\textcircled{\mathsf{p}}$ since the p-fundamental index of $m \mapsto \text{Ack}_2(n,m)$ is computable from n and can be simulated with the universal function for p-fundamental functions. The converse is true since for any *acceptable* enumeration $\mathbf{\Phi}_{\cdot}^{\mathsf{c}}$ of p-fundamental functions, there exists a primitive recursive function f such that $\forall e, \forall n, \text{Ack}_2(f(e),n) \leqslant \mathbf{\Phi}_e^{\mathsf{c}}(n)$[11]. Hence, the Ackermann function permits us to compute the p-fundamental busy beaver.

Following our s-m-n theorem for c-fundamental functions, we obtain generalizations of Kleene's second recursion theorem to sub-computabilities.

Theorem 3 (Recursion theorem *à la* Kleene). *Let f be a c-fundamental function.*

1. *(*Unbalanced fundamental recursion*) There is an $n \in \mathbb{N}$ such that $\mathbf{\Phi}_n^{\copyright} \cong \mathbf{\Phi}_{f(n)}^{\mathsf{c}}$.*

[9] Ack_2 denotes the classical binary function and Ack the diagonal unary version.

[10] For example, take f a total computable function not in c. Transform it into f' such that $\forall n, f'(n) = \langle f(n),\ \text{steps}(f,n)\rangle$ (where $\text{steps}(f,n)$ is the number of steps needed by f on the input n to halt). f' still does not belong to c, but f'^{-1} does.

[11] This function works by recursion on the code of $\mathbf{\Phi}_e^{\mathsf{c}}$, see [9], thm VII.8.10 p299-300.

2. *(*Unbalanced recursion*) There is an $n \in \mathbb{N}$ such that $\varphi^{\complement}_n \cong \varphi^{\complement}_{f(n)}$.*
3. *(*Half recursion*) Let h be a somewhat $\complement$-computable function defined on an infinite $\complement$-enumerable and co-r.e. set A. There is an $n \in \mathbb{N}$ such that $\varphi^{\complement}_n|_{\overline{A}} \cong \varphi^{\complement}_{f(n)}|_{\overline{A}}$ and $\varphi^{\complement}_n|_A \cong h$.*

Proof. The function $u \mapsto g_x(u) = \begin{cases} \varphi^{\complement}_a(u) & \text{if } u \in A, \\ \varphi^{\complement}_{\mathbf{\Phi}^{\complement}_x(x)}(u) & \text{otherwise,} \end{cases}$ (where a is a $\complement$-index for h) is somewhat $\complement$-computable and of $\complement$-index computable from x by $d_a \in \complement$. Let e_a be a $\complement$-fundamental index for $f \circ d_a$. Choose $n = d_a(e_a)$, since $\varphi^{\complement}_{d_a(e_a)}(u) \cong \varphi^{\complement}_a(u)$ if $u \in A$, and $\varphi^{\complement}_{d_a(e_a)}(u) \cong \varphi^{\complement}_{\mathbf{\Phi}^{\complement}_{e_a}(e_a)}(u) \cong \varphi^{\complement}_{f \circ d_a(e_a)}(u)$ otherwise.

We also have the usual corollary of a $\complement$-recursion theorem *with parameters*. As it is visible in the various proofs of Theorem 3, the function providing the *fixed point* for each parameter is a $\complement$-fundamental function.

As we have already remarked, we do not have a general s-m-n theorem for somewhat computable functions, only one for fundamental functions[12]. The principal reason is the fact that somewhat computable functions are not closed by composition[13]. And at the same time, there is no[14] (balanced) full fixed-point

[12] There is a duality at play between the fundamental functions and the somewhat $\complement$-computable functions. Duality both in their use in theorems, usually generalizations (or really restrictions) of classical computability theorems, and in the theorems they verify. The former is used to construct the latter and verifies padding and s-m-n (or composition) but not fixed-point, nor universality, while the latter verifies padding, fixed-point and universality, but not s-m-n (nor composition). An example of the use in theorems is the fixed point theorem: the fixed point is of a fundamental function over the enumeration of the somewhat computable functions. These facts are to be put in perspective with the classical results that enumeration alone or even enumeration and fixed point are not enough to ensure having an acceptable computation system, *i.e.* a computation system isomorphic to the canonical enumeration of all r.e. sets, while enumeration and composition are sufficient: Enumeration is not enough since a Friedberg enumeration, *i.e.*, without repetition, of r.e. sets will obviously not verify the padding lemma. Enumeration and fixed point is not enough by the following counter-example: if $e \mapsto \varphi_{h(e)}$ is a Friedberg enumeration without repetition of the partial recursive functions, then consider the enumeration $e \mapsto \psi_e = \varphi_{h(\varphi_e(0))}$. It is not hard to show that $\psi_\cdot$ verifies the fixed-point theorem ($\exists$ recursive f, $\forall$ total ψ_i, $\psi_{\psi_i(f(i))} \cong \psi_{f(i)}$) but each partial recursive function is equal to a $\varphi_{h(i)}$ for a unique i and thus equal to a ψ_e only if $\varphi_e(0) = i$. Thus, the set of ψ-indices of each partial recursive function is r.e., which is in opposition to the fact that the nowhere-defined function ($x \mapsto \uparrow$) cannot have an r.e. set of indices in an acceptable enumeration.

[13] For well-behaving functions, composition is still possible and in a primitively effective way. For example, composition $f \circ g$ is possible for somewhat $\complement$-computable functions f and g such that g^{-1} is $\complement$-fundamental and there is a $\complement$-enumerable subset of the intersection of the range of g and the graph of f. If you take the function $\mathbf{J}_{\complement} : x \mapsto \varphi^{\complement}_x(x)$ as f, then f being somewhat p-computable and having a graph containing many p-enumerable subsets, many functions g will make $f \circ g$ somewhat $\complement$-computable.

[14] Both fundamental and somewhat (balanced) full fixed-point theorems for p would provide fixed points which would be indices of the Ackermann function. The function

theorem on fundamental or somewhat computable functions (staying in a particular sub-computability and not using higher sub-computabilities in an unbalanced way like in Theorem 3).

Even though, several corollaries of s-m-n still carry through: the effective union given by H_{c} and the following corollary of the existence of a c-universal function in $\copyright$: if f is a c-fundamental function, then there exists a $\copyright$-fundamental function g such that for all x, $\mathbf{W}^{\mathsf{c}}_{g(x)} = f\text{“}\mathbf{W}^{\mathsf{c}}_{x}$.

It is worthwhile to remark that a universal function (for fundamental functions) and $x \mapsto \mathbf{\Phi}^{\mathsf{c}}_{x}(x)$ provide examples of functions which are not c-fundamental but which are total and somewhat c-computable. Notice also that fundamentals go beyond total somewhat c-computable functions: $\mathbf{BB}^{\mathbf{\Phi}}_{\mathsf{c}}$ is not somewhat c-computable but is $\copyright$-fundamental.

We should also notice that as in classical computability, not all partial somewhat c-computable functions are extendible to total somewhat c-computable functions. A counterexample is the function $x \mapsto \varphi^{\mathsf{c}}_{x}(x)+1$, since it is somewhat c-computable and differs with all somewhat c-computable functions and thus also with all total ones.

We generalize Rice's theorem to sub-computabilities. To this end, we define the following reducibility and notion of set of indices: a set A is somewhat c-reducible to a set B (designated by $A \preccurlyeq^{\mathsf{c}}_{\sim} B$) if there is a somewhat c-computable function f such that $x \in A$ if and only if $f(x) \in B$. A set $A \subseteq \mathbb{N}$ is a *set of c-somewhat indices* (resp. *a set of c-fundamental indices*) if for all x, y, ($x \in A$ and $\varphi^{\mathsf{c}}_{x} \cong \varphi^{\mathsf{c}}_{y}$) $\Longrightarrow y \in A$, (resp. ($x \in A$ and $\mathbf{\Phi}^{\mathsf{c}}_{x} \cong \mathbf{\Phi}^{\mathsf{c}}_{y}$) $\Longrightarrow y \in A$).

Theorem 4 (Rice for sub-computabilities). *If A is a non-trivial ($A \neq \varnothing, \mathbb{N}$) set of c-indices (resp. c-fundamental indices), then we have either $\mathbf{K}_{\mathsf{c}} \preccurlyeq^{\mathsf{c}}_{\sim} A$ or $\mathbf{K}_{\mathsf{c}} \preccurlyeq^{\mathsf{c}}_{\sim} \overline{A}$ (resp. $\mathbf{K}^{\mathbf{\Phi}}_{\mathsf{c}} \preccurlyeq^{\copyright}_{1} A$ or $\mathbf{K}^{\mathbf{\Phi}}_{\mathsf{c}} \preccurlyeq^{\copyright}_{1} \overline{A}$).*

Productivity and creativity can also be non-trivially ported to our setting: A set X is c-*productive* if there exists a one-one c-fundamental function ψ, called the *productive function* for X, such that $\forall x, \mathbf{W}^{\mathsf{c}}_{x} \subseteq X \Longrightarrow \psi(x) \in X \setminus \mathbf{W}^{\mathsf{c}}_{x}$. A c-enumerable set X is c-*creative* if $\overline{X}$ is c-productive.

It is a non-trivial generalization of the classical notion, since Ack “$\overline{\mathbf{K}}$ is productive, but not p-productive. As expected, $\mathbf{K}_{\mathsf{c}}$ is still c-creative. And variants of the properties carry through: If X is c-productive, then X is neither c-enumerable nor weakly c-computable. (Hence $\mathbf{K}^{\mathbf{\Phi}}_{\mathsf{c}}$ is not c-creative.) If X is c-productive, then X contains a c-enumerable infinite subset. And, if X is c-productive and $X \preccurlyeq^{\mathsf{c}}_{1} A$ by a function f such that f^{-1} is also one-one c-fundamental, then A is $\copyright$-productive[15].

$n \mapsto$ fundamental c-index of $y \mapsto \sum_{x \leqslant y} \mathbf{\Phi}^{\mathsf{c}}_{n}(x) + 1$ is also designed not to have any of those full fixed-points.

[15] $\mathbf{K}^{\mathbf{\Phi}}_{\mathsf{c}}$ is not c-creative, but it is $\mathbf{\Phi}_{\mathsf{c}}$-creative where $\left(\mathbf{W}^{\mathsf{c}}_{n}\right)_{n\in\mathbb{N}}$ is replaced by $\left({}^{\mathbf{\Phi}}\mathbf{W}^{\mathsf{c}}_{n} = \{y : \mathbf{\Phi}^{\mathsf{c}}_{n}(y) > 0\}\right)_{n\in\mathbb{N}}$ in the definition of productivity.

We would like to carry through the characterization of creativity by $\mathbf{K}$ (creativity is equivalent[16] to m-completeness) to sub-computabilities but we need for that to have the right notion of m-reducibility. The third property above hints that it could be challenging.

We now generalize Rogers' isomorphism theorem to sub-computabilities. To this end, we first show that the constructive version, due to Myhill, of the Cantor-Schröder-Bernstein Theorem can be adapted to $\mathtt{C}$-computabilities: (Myhill for sub-computabilities)[17] $A \equiv_1^{\mathtt{C}} B \iff A \equiv^{\mathtt{C}} B$.

We can generalize the notion of "acceptable system of indices" to sub-computabilities:

Definition 5. An *acceptable system of* $\mathtt{C}$*-computability* is a pair $(\mathbf{\Psi}^{\mathtt{C}}_{\cdot}, \psi^{\mathtt{C}}_{\cdot})$ providing maps from $\mathbb{N}$ onto the set of $\mathtt{C}$-fundamental functions and the set of somewhat $\mathtt{C}$-computable functions such that there are $\mathtt{C}$-fundamental functions f, g, $\mathbf{f}$ and $\mathbf{g}$ such that for all e $\psi^{\mathtt{C}}_e \cong \varphi^{\mathtt{C}}_{f(e)}$ and $\varphi^{\mathtt{C}}_e \cong \psi^{\mathtt{C}}_{g(e)}$, $\mathbf{\Psi}^{\mathtt{C}}_e \cong \mathbf{\Phi}^{\mathtt{C}}_{\mathbf{f}(e)}$ and $\mathbf{\Phi}^{\mathtt{C}}_e \cong \mathbf{\Psi}^{\mathtt{C}}_{\mathbf{g}(e)}$ and g is one-one on indices of finite domain functions.

An acceptable system of sub-computability inherits many properties of the respective canonical sub-computability. In particular, the Half recursion[18] for sub-computabilities (Theorem 3), parametrization[19] for fundamentals and the padding lemma[20] for somewhat $\mathtt{C}$-computable functions holds in any acceptable system of sub-computability.

Theorem 5 (Isomorphism theorem *à la* Rogers). *Let* $\mathtt{C}$ *be a sub-computability. For any acceptable system of* $\mathtt{C}$*-computability,* $(\mathbf{\Psi}^{\mathtt{C}}_{\cdot}, \psi^{\mathtt{C}}_{\cdot})$*, there is a* $\mathtt{C}$*-fundamental permutation* h *of* ω *such that for all* e*,* $\varphi^{\mathtt{C}}_e \cong \psi^{\mathtt{C}}_{h(e)}$.

We thus notice that acceptable systems of indices provide the same structure theory for a sub-computability as the *standard* one we have defined and been using. Things happen not because of the coding used: a foundation induces a unique sub-computability.

---

[16] If X is productive and $X \preccurlyeq_1 A$, then A is productive. And if X is productive then $\overline{\mathbf{K}} \preccurlyeq_1 X$.

[17] A set A is $\mathtt{C}$-reducible to a set B (designated by $A \preccurlyeq_1^{\mathtt{C}} B$) if there is a one-one $\mathtt{C}$-fundamental function f such that $x \in A$ if and only if $f(x) \in B$. A set A is $\mathtt{C}$-equivalent to a set B (designated by $A \equiv_1^{\mathtt{C}} B$) if $A \preccurlyeq_1^{\mathtt{C}} B$ and $B \preccurlyeq_1^{\mathtt{C}} A$. A set A is $\mathtt{C}$-isomorphic to B (designated by $A \equiv^{\mathtt{C}}$) if there exists a $\mathtt{C}$-fundamental permutation p such that $p''(A) = B$.

[18] There is a $\mathbf{\Psi}^{\mathtt{C}}_{\cdot}$-fundamental index Kleene_ψ such that for all i, $\mathbf{j}$, if $\mathrm{domain}(\psi^{\mathtt{C}}_i)$ is infinite and co-r.e., then $\psi^{\mathtt{C}}_{\mathbf{\Psi}^{\mathtt{C}}_{\mathrm{Kleene}_\psi}(i,\mathbf{j})}|_{\overline{\mathrm{domain}(\psi^{\mathtt{C}}_i)}} \cong \psi^{\mathtt{C}}_{\mathbf{\Psi}^{\mathtt{C}}_{\mathbf{j}}(\mathbf{\Psi}^{\mathtt{C}}_{\mathrm{Kleene}_\psi}(i,\mathbf{j}))}|_{\overline{\mathrm{domain}(\psi^{\mathtt{C}}_i)}}$ and $\psi^{\mathtt{C}}_{\mathbf{\Psi}^{\mathtt{C}}_{\mathrm{Kleene}_\psi}(i,\mathbf{j})} \cong \psi^{\mathtt{C}}_i$.

[19] There is a $\mathbf{\Psi}^{\mathtt{C}}_{\cdot}$-fundamental index param_ψ such that for all i_0, i_1, $\mathbf{j}$, the function $\psi^{\mathtt{C}}_{\mathbf{\Psi}^{\mathtt{C}}_{\mathrm{param}_\psi(\langle i_0,i_1\rangle,\mathbf{j})}(e)}$ is equal (when defined) to $\begin{cases} \psi^{\mathtt{C}}_{i_0} \text{ if } \mathbf{\Psi}^{\mathtt{C}}_{\mathbf{j}}(e) = 0, \\ \psi^{\mathtt{C}}_{i_1} \text{ otherwise.} \end{cases}$

[20] In any acceptable system of $\mathtt{C}$-computability, given one index of a somewhat $\mathtt{C}$-computable function with infinite domain, we can $\mathtt{C}$-fundamentally generate infinitely many indices of the same function.

3 Sub-reducibilities

Our interest in sub-reducibilities in the context of sub-computabilities is twofold.

As we have already seen with the previous theorems, the same kind of results than the classical ones appear in most sub-computabilities. For example, we have recursion theorems *à la* Kleene and an isomorphism theorem *à la* Rogers. It is even more striking with the degree structure as will be outlined in this section. We will see in particular that each Turing degree is divided into infinitely many different degrees, our sub-reducibilities being a recursive refinement of the classical ones. Some computable sets are shown to play a role similar to the classical solutions of Post's problem, from our sub-reducibilities point of view. As these results are quite natural and easy to prove, it would be interesting, and it is one of our goal for the future, to prove some kind of degree-structure homogeneity theorem to be able to somehow carry the sub-computability proofs back to the classical computability setting.

Another goal in the study of sub-computabilities is to find real classical computability sets, *e.g.*, solutions to Post's problem, that are already in a sub-computability but not in *weaker* sub-computabilities, *i.e.*, a r.e. set $\varnothing <_T X <_T \varnothing'$ that first appears in the c-r.e. sets.

Since $A \preccurlyeq_T B$ means that A is computable in B (really, computable in χ_B), and in sub-computabilities, we have several notions of recursiveness (for a set), we now introduce several associated Turing (or stronger) reducibilities:

Definition 6 (Sub-reducibilities). A function f is χ-c-*Turing* reducible to B ($f \preccurlyeq^{\chi}_{\mathsf{c}\text{-}T} B$) if f is in $\mathsf{c}[\chi_B]$.[21] A set A is χ-c-*Turing* reducible to B if its characteristic function is.

A function f is somewhat c-*Turing* reducible to B ($f \preccurlyeq^{\sim}_{\mathsf{c}\text{-}T} B$) if f is somewhat computable over $\mathsf{c}[\chi_B]$. A set A is somewhat c-*Turing* reducible to B if its characteristic function is.

A set A is χ-c-r.e. in B if A is one-one enumerable by a function χ-c-*Turing* reducible to B.

A set A is weakly c-*Turing* reducible to B ($A \preccurlyeq^{\mathrm{w}}_{\mathsf{c}\text{-}T} B$) if A and $\overline{A}$ are one-one enumerable by functions in $\mathsf{c}[\mathfrak{e}_B, \mathfrak{e}_{\overline{B}}]$, for any function $\mathfrak{e}_X$ which one-one enumerates X.

A set A is strongly-c-*Turing* reducible to B ($A \preccurlyeq^{\mathrm{s}}_{\mathsf{c}\text{-}T} B$) if A is enumerable by an increasing function in $\mathsf{c}[p_B, p_{\overline{B}}]$, where p_X is the increasing enumeration of X.

A function is reducible to B if its graph is reducible to B.

When the type of c-reducibility is not specified, it usually means that it is true for each of these reducibilities.

$\mathbf{K}_{\mathsf{c}}$ is (weak/strong/χ) c-Turing complete: $\mathbf{K} \equiv_{\mathsf{c}\text{-}T} \mathbf{K}_{\mathsf{c}} \equiv_{\mathsf{c}\text{-}T} \varnothing'$. There are also new incomparable sets: for some A and some c, $A \not\preccurlyeq^{\chi}_{\mathsf{c}\text{-}T} A'$, for example, for $A = \mathrm{range}(\mathrm{Ack})$ and $\mathsf{c} = \mathsf{p}$.

[21] $\mathsf{c}[g]$ is the primitive closure of $\mathsf{c} \cup \{g\}$ (See Definition 1). We could take a stronger closure since c could be closed by more than composition and primitive recursion but for our purposes, taking the primitive closure seems to be enough.

$\preccurlyeq^{\chi}_{\mathtt{C}\text{-}T}$ is strictly stronger than the truth table reducibility $\preccurlyeq_{tt}$ (and thus also than the weak truth table reducibility $\preccurlyeq_{wtt}$ and the Turing reducibility $\preccurlyeq_T$). $\preccurlyeq^{\mathrm{w}}_{\mathtt{C}\text{-}T}$ and $\preccurlyeq^{\mathrm{s}}_{\mathtt{C}\text{-}T}$ are strictly stronger than the Turing reducibility (the inclusion is trivial and range(Ack) $\preccurlyeq_T \varnothing$ but range(Ack) $\not\preccurlyeq^{\mathrm{w,\ s}}_{\mathtt{C}\text{-}T} \varnothing$). $\preccurlyeq^{\mathrm{w}}_{\mathtt{C}\text{-}T}$ is incomparable[22] with $\preccurlyeq_{tt}$ and $\preccurlyeq_{wtt}$. Ack $\preccurlyeq^{\chi}_{\mathtt{C}\text{-}T} \varnothing$ but Ack $\not\preccurlyeq^{\mathrm{s}}_{\mathtt{C}\text{-}T} \varnothing$, thus $\preccurlyeq^{\chi}_{\mathtt{C}\text{-}T}$ does not imply $\preccurlyeq^{\mathrm{s}}_{\mathtt{C}\text{-}T}$. $\preccurlyeq^{\chi}_{\mathtt{C}\text{-}T}$ implies $\preccurlyeq^{\sim}_{\mathtt{C}\text{-}T}$, which in turn implies $\preccurlyeq_{wtt}$.

A generalization of *Martin-Arslanov-Lachlan's completeness criterion* holds: Given a C-enumerable set A, $\varnothing' \preccurlyeq^{\sim}_{\mathtt{C}\text{-}T} A$ is equivalent to the existence of a weakly C-computable set B and function $f \preccurlyeq^{\sim}_{\mathtt{C}\text{-}T} A$, such that f is FPF$[B]$[23].

One of the interests of considering these reducibilities is to see what sub-computabilities have to say about the computability of sets of integers that are not necessarily recursively enumerable. A lot of results of classical computability, especially concerning Δ^0_2 sets, can be extended (or really restricted) to sub-computabilities but we will only consider here ways of providing solutions to Post's problem of finding an incomputable yet incomplete recursively enumerable set.

When considering Post's problem with regard to these strong reducibilities $\preccurlyeq_{\mathtt{C}\text{-}T}$, it becomes fast obvious that finding solutions is very simple.

Theorem 6 (Solutions to Post's sub-problem). *There exists an r.e. set X such that $\varnothing \prec_{\mathtt{C}\text{-}T} X \prec_{\mathtt{C}\text{-}T} \varnothing'$ for $\prec_{\mathtt{C}\text{-}T}$=$\prec^{\chi}_{\mathtt{C}\text{-}T}$ or $\prec^{s}_{\mathtt{C}\text{-}T}$.*

A proof of this theorem may use the fact that the χ-C-jump[24] of $\mathbf{K}^{\mathbf{\Phi}}_{\mathtt{C}}$ is χ-C-reducible to $\mathbf{K} \equiv^{\chi}_{\mathtt{C}\text{-}T} \mathbf{K}_{\mathtt{C}} \equiv^{\chi}_{\mathtt{C}\text{-}T} \varnothing'$. $\mathbf{K}^{\mathbf{\Phi}}_{\mathtt{C}}$ is thus a C-low C-r.e. set; it represents a low r.e. set in the C sub-computability. The set graph($\mathbf{BB}^{\mathbf{\Phi}}_{\mathtt{C}}$) is another example of a C-low set, albeit not C-r.e.

There are obvious solutions to Post's (real) problem in our sub-computabilities, *i.e.*, a C-r.e. set X such that $\varnothing <_T X <_T \varnothing'$. But the nice thing is that we can manage to make them appear only starting from a given C-computability. And the same goes for C-r.e. low degrees.

These results put together hint to the following lemma: in each 1-degree, one can find a set as high in the sub-computability hierarchy as desired. More generally, for any sub-computability C, each r.e. 1-degree has a non-C-enumerable r.e. member and a C-enumerable member.

A corollary that completes the previous result is that, there exists a non-C-enumerable r.e. set X, such that $\varnothing <_T X <_T \varnothing'$. A consequence is that there exists a low r.e. set W which is not C-enumerable.

---

[22] Let A be a set having the property of being Turing complete but not truth table complete, *e.g.*, an effectively hypersimple set. Then, $B = \{3a : a \in A\} \cup \{3n+1 : n \in \mathbb{N}\}$ is weakly C-Turing complete and not truth table complete.

[23] f is fixed point free relatively to B (FPF$[B]$) if $\forall x, \exists n \in B, \varphi^{\mathtt{C}}_x(n) \neq \varphi^{\mathtt{C}}_{f(x)}(n)$.

[24] Since this section is, for now, only an introduction to sub-reducibilities in sub-computabilities, we will only introduce the χ-C-jump in this footnote. The (degree of the) χ-C-jump of A, denoted by $A'_{\mathtt{C}}$, is the $\preccurlyeq^{\chi}_{\mathtt{C}\text{-}T}$ maximal equivalence class containing a set which is χ-C-r.e. in A.

Using a priority argument, we can even ensure the low promptly simple set X to be $\mathtt{C}$-low ($X'_{\mathtt{C}} \preccurlyeq_{\mathtt{C}\text{-}T} \varnothing'$) but not $\mathtt{C}^-$-low for sub-computabilities $\mathtt{C}^-$ weaker[25] than $\mathtt{C}$, in the same way we can build a low simple set that is not superlow (see [7, Ex. 1.6.7, p. 386]).

As we saw multiple times in this paper, sub-computabilities are strongly linked with fast growing functions, which gives evidence of their *recursive power*. As this kind of reasoning also appears in honest elementary degrees theory we hope to soon be able to develop the links between these degree theories.

Other immediate goals are to port some essential computability results to sub-computabilities, especially various characterization of r.e. sets and reducibilities.

Acknowledgements

We would like to thank Bruno Durand for many challenging discussions on the ideas of this paper. Many thanks also to the anonymous referee whose comments and remarks have helped us to clarify its presentation.

References

1. Friedman, H., Sheard, M.: Elementary descent recursion and proof theory. Annals of Pure and Applied Logic 71(1), 1–45 (1995)
2. Koz'minyh, V.V.: On a presentation of partial recursive functions by compositions. Algebra i Logika 11(3), 270–294 (1972), in Russian
3. Kristiansen, L.: Papers on subrecursion theory. Ph.D. thesis, Department of Informatics, University of Oslo (1996)
4. Kristiansen, L.: A jump operator on honest subrecursive degrees. Archive for Mathematical Logic 37(2), 105–125 (1998)
5. Kristiansen, L.: Low_n, high_n, and intermediate subrecursive degrees. In: Calude, Dinneen (eds.) Combinatorics, computation and logic, pp. 286–300. Springer, Singapore (1999)
6. Kristiansen, L., Schlage-Puchta, J.C., Weiermann, A.: Streamlined subrecursive degree theory. Annals of Pure and Applied Logic (2011), to appear
7. Nies, A.: Computability and Randomness. Oxford University Press (2009)
8. Odifreddi, P.: Classical Recursion Theory. North Holland - Elsevier (1988)
9. Odifreddi, P.: Classical Recursion Theory. Volume II. North Holland - Elsevier (1999)
10. Rathjen, M.: The realm of ordinal analysis. In: Cooper, S.B., Truss, J. (eds.) Sets and Proofs, pp. 219–279. Cambridge University Press (1999)
11. Rose, H.E.: Subrecursion: Functions and Hierarchies, Oxford Logic Guides, vol. 9. Oxford University Press, USA, New York, NY (1984)
12. Veblen, O.: Continuous increasing functions of finite and transfinite ordinals. Trans. Amer. Math. Soc. 9, 280—292 (1908)

[25] In the sense that $\mathtt{C}^-$ is strictly included in $\mathtt{C}$.