A General Framework for Computability[*]

Fabien Givors, Grégory 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 the structure of the r.e. sets that are 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. The similarity between sub-computabilities and (complete) computability is surprising, since there are so many missing r.e. sets in sub-computabilities. This similarity hints at a general framework for computability, in which other computabilities, especially hyper-computabilities, can be embedded.

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 main focus of this article. Higher computability can also be cast in this setting. 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 an enumeration of r.e. sets (called c-*enumerable*), which are one-one enumerated[1] by functions (called c-*fundamental*) in a sufficiently closed collection $\mathfrak{F}_{\mathsf{c}}$ (called the foundation of c) of total recursive functions. 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.

Sub-computabilities are like classical computability, the collection of all r.e. sets, but with holes punched in. Building on an enumeration $\mathbf{\Phi}_{\cdot}^{\mathsf{c}}$ of $\mathfrak{F}_{\mathsf{c}}$, we can respectively build canonical enumerations $\mathbf{W}_{\cdot}^{\mathsf{c}}$ (resp. $\varphi_{\cdot}^{\mathsf{c}}$) of c-enumerable sets (resp. somewhat c-computable functions).

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 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) effectiveness 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 *sub-recursion*. 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.

[*]The abstract 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).

[1]The complete definition also incorporates finite sets and $\varnothing$.

Studying sub-computabilities amounts to classifying r.e. sets by measuring the difficulty of enumerating them by way of sub-recursion. 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 approximation 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 recursvive". Another example is Kleene's second recursion theorem, which only partially holds in various ways in our sub-computabilities.

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. One of the interesting aspects of this study is the simplicity of some of the proofs of the degree structure. 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. Another application of sub-computabilities is using the strong links between proof theory and sub-recursion to tailor some reverse mathematics of computability and to partition the r.e. degrees according to the strength needed to prove their properties.

Our study is very different from other *sub-recursive 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 sub-recursive degrees* (see [3, 4]), to be able to use their techniques and have their applications.

This abstract is a summary of [1], covering sub-computabilities, sub-reducibilities and related results concerning completeness and Post's problem. Our work uses elements of proof theory, in particular sub-recursion (see [6, 7]). A reference on higher computability is [8].

References

[1] F. Givors, G. Lafitte. *Holes punched computabilities.* 25 pages, soon at `http://www.lif.univ-mrs.fr/~lafitte/hpc.pdf` (February 2011)

[2] V.V. Koz'minyh. *On a presentation of partial recursive functions by compositions.* Algebra i Logika **11**(3), pp.270—294. In Russian. 1972.

[3] L. Kristiansen. *Papers on subrecursion theory.* PhD thesis, University of Oslo, 1996.

[4] L. Kristiansen. *A jump operator on honest subrecursive degrees.* Archive for Mathematical Logic **37**(2), pp.105—125, 1998.

[5] P. Odifreddi. *Classical Recursion Theory.* Volume I and II. North Holland - Elsevier, 1988 and 1999.

[6] W. Pohlers. *Proof Theory: The First Step into Impredicativity.* Springer, 2008.

[7] H.E. Rose. *Subrecursion: Functions and Hierarchies.* Vol. 9 of Oxford Logic Guides. Oxford University Press, USA, New York, 1984.

[8] G.E. Sacks. *Higher Recursion Theory.* Springer-Verlag, 1990.