EXPLORING AND UNDERSTANDING THE ABSTRACT BY DIRECT MANIPULATION OF THE CONCRETE

Oksana Arnold[1], Jun Fujima[2], Klaus P. Jantke[2] and Yuzuru Tanaka[3]
[1]*Erfurt University of Applied Sciences, Department of Applied Computer Science, Altonaer Str. 25, 99085 Erfurt, Germany*
[2]*Fraunhofer IDMT, Children's Media Department, KinderMedienZentrum, Erich-Kästner-Str. 1a, 99094 Erfurt, Germany*
[3]*Hokkaido University Sapporo, Meme Media Laboratory, Kita-13, Nishi-8, Kita-ku, Sapporo, 060-8628 Japan*

Keywords: Computability Theory, Webble Technology, Direct Manipulation, Exploratory Learning, Playful Learning.

Abstract: In schools and in universities, there will always remain a certain very abstract, but highly relevant content. Remarkable percentages of learners, therefore, are facing severe difficulties and need some substantial support. The paper demonstrates an involved case of learning some of the most abstract contents in higher education by means of exploratory direct manipulation. Direct manipulation technologies allow for touching media objects, for moving, modifying and combining them, and for–even playfully–exploring the behavior of mechanisms. Webble technology has been used to represent content and to implement direct manipulation functionalities. The implementation reported is in practical use within some moodle environment.

1 INTRODUCTION

Recursion theory also known as the theory of effective computability (Rogers jr., 1967) is among the most abstract areas of computer science. The basic results of recursion theory are explaining the principle reach and limitations of computers. They are underlying the fundamental insights of logic–from the classics to the present–into decidability and undecidability. Last but not least, recursion theory is determining the reach of the automation of human reasoning (Siekmann and Wrightson, 1983a; Siekmann and Wrightson, 1983b).

A correct and comprehensive treatment requires enormous efforts of understanding abstract concepts and brings with it notations of a high complexity like

$$\varphi_x^{(m+n)}(y_1,\ldots,y_m,z_1,\ldots,z_n) = \varphi^{(n)}_{s_n^m(x,y_1,\ldots,y_m)}(z_1,\ldots,z_n)$$

This equality holds for any GÖDEL numbering φ, where for any two natural numbers m and n there does exist a certain general recursive encoding function s_n^m accordingly ((Rogers jr., 1967), §1.8, p. 23).

Despite awareness of the discipline's relevance, many students are reluctant to memorizing, understanding, and actively using formalisms like this.

What are the potentials of media technologies and media didactics to alleviate the bottleneck of calculi of effective computability? In particular, what are the potentials of direct manipulation and of intuitive interfaces? If technologies may be of any help, how? How to learn the abstract by concrete manipulation?

2 TOWARD MEDIA DIDACTICS FOR TEACHING ABSTRACT CONCEPTS BY MEANS OF CONCRETE MANIPULATION

There is some scientific evidence that teaching resp. learning abstract concepts may be substantially supported by concrete activities (Barsalou and Wiemer-Hastings, 2005). However, concepts considered in the present approach are by far more formal than those studied by Barsalou and Wiemer-Hastings (2005).

One of the problems with the abstract concepts and relations of computability theory is that starring at a formalisms like the s-m-n-theorem on the left does not help at all. Those formalisms have some operational meaning, but they are not "running" in any way.

In particular, it helps to learn and, finally, master the formalisms by studying the formulas' meaning for a larger number of specific input data. Unfortunately, given any particular data, such an exploration of some formula's meaning is tedious and time-consuming. Learners have to perform by hand lengthy formal and mostly trivial, but error-prone transformations.

Arnold O., Fujima J., P. Jantke K. and Tanaka Y..
EXPLORING AND UNDERSTANDING THE ABSTRACT BY DIRECT MANIPULATION OF THE CONCRETE.
DOI: 10.5220/0003916001000107
In *Proceedings of the 4th International Conference on Computer Supported Education* (CSEDU-2012), pages 100-107
ISBN: 978-989-8565-07-5

Roughly speaking, there is the question for media environments in which the abstract has some concrete representations that allow for a playful manipulation by hand. Operational meanings "just take place" and are observed to provoke thought and interpretation.

3 COMPUTABILITY THEORY

This section is intended to present the domain in some detail to provide a firm foundation of the subsequent investigation.

About hundred years ago, a large community of scientists did strongly believe in the solvability of any problem supposed the problem has been expressed sufficiently clear and formal. This assumption was underlying David Hilbert's famous and remarkably influential speech to the 2^{nd} International Congress of Mathematicians, Paris, August 1900 (Gray, 2000).

However, a large variety of problems such as the difficulty to find a sound foundation of set theory lead to severe doubts. Could it, perhaps, be the case that there were some generally unsolvable problems?

For strictly proving the unsolvability of any particular problem, i.e. to demonstrate that no algorithm ever will be able to provide a solution, one needs to have a sufficiently clear understanding of definitely all algorithms. This insight lead to the emergence of a general theory of algorithms or, in other terms, of effective computability (Rogers jr., 1967).

There are largely varying approaches primarily by authors such as Alonzo Church (Church, 1936), Stephen C. Kleene (Kleene, 1936), Emil Post (Post, 1936), and Alan M. Turing (Turing, 1936). All these attempts to specificy computability, in general, turn out to be provably equivalent (see (Machtey and Young, 1978), e.g.)–an evidence for the so-called CHURCH's Thesis that each formalization correctly reflects the true nature of computability (see (Church, 1936), footnote 2, page 346).

Relying on CHURCH's Thesis, it is sufficient to study one of these approaches in detail. For the purpose of the research and the applications in higher education reported in this paper, the authors have selected the concept of partial recursive functions originating from (Gödel, 1931) and (Kleene, 1936) (see also (Péter, 1981)).

The underlying key idea of this approach is to begin with a few obviously computable functions and to define the class of all computable functions to be the closure under a few obviously computable operators.

- Every constant function is computable.
- The successor function is computable.
- The projection on an argument is computable
- Substitution inherits computability.
- Limited recursion inherits computability.
- The minimum operator inherits computability.

The first three items above specify the basic functions, whereas the following three items name the operators.

Illustrative applications will follow in section 6.

4 DIRECT MANIPULATION

Conventionally, studies of recursive function theory involve much writing of terms and term equations describing certain functions. Suppose s denotes the successor function and p_n^m (for any m and n with $n \leq m$) denotes the m-ary projection function which selects the nth element of any m-tupel.

$$add(x,0) = p_1^1(x) = 1 \tag{1}$$

$$add(x,s(y)) = s(p_3^3(x,y,a(x,y))) \tag{2}$$

is a definition of addition add by means of limited recursion which is the abstract form of inductive definition (the terminus technicus is "primitive recursion").

Imagine that, instead of writing formulas as shown above, functions appear as media objects. Imagine the same applies to operators for the construction of new functions. Instead of writing equations, you may grasp an operator media object (very much in the sense of grasping in (MacKenzie and Iberall, 1994)) and, in addition, you may grasp some function objects to plug them into the input slots of the operator object. In the end, you get a new media object describing a certain new function.

The enormous advantage of directly manipulating objects is that the newly constructed one at the learner's fingertips is operational.

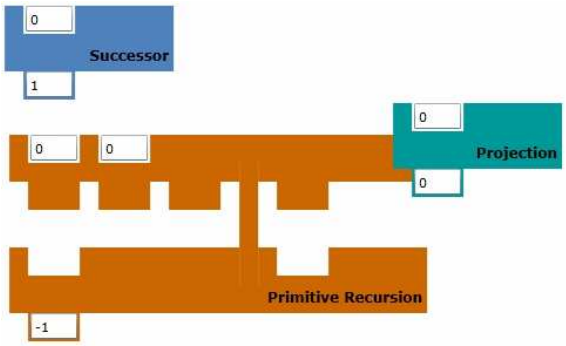

Figure 1: Two function objects and an operator object in use; the learner plugging the Projection object into the Primitive Recursion object by dragging it with her fingers into the slot of the operator object (at the figure's bottom).

Pondering the correctness of your construction, there is no need to stare at a formula. Instead, you simply play with your construct putting some values in and looking for the outcome.

In particular, error-prone symbol manipulations are avoided. Instead, learners can concentrate on the key phenomena of setting up some new definition. In particular, variants can be explored easily. Last but not least, it may be more or less fun to manipulate objects physically on a touch screen or on an interactive whiteboard. In contrast, it has been rarely reported that writing lengthy formulas is considered any fun.

5 MEME MEDIA TECHNOLOGY

Independent of computer science and engineering and of media technologies, Richard Dawkins (1976) has published an influential book on the evolution of thought. Excited by these ideas, Susan Blackmore (Blackmore, 1999) is discussing the philosophical reach of Dawkins' perspective.

Dawkins' key concept is named Meme used to denote the building blocks of human thought somehow similar to the genes building the characteristics of the whole human being. Yuzuru Tanaka has accepted the challenge, so to speak, from a computer science and media technology perspective (Tanaka, 2003). His pioneering work aims at concepts and implementations that result in environmental conditions for the computer-supported evolution of human knowledge–memes à la Dawkins being digitalized and prepared for digital evolution. Accordingly, Tanaka speaks about Meme Media.

From a somehow technocratic point of view, particular meme media technologies as described in (Tanaka, 2003) establish some middleware which is particularly suitable for the development of systems in which knowledge evolution plays a prominent role. Pads are the meme media object encapsulating knowledge. Knowledge units can be subject to mutation or cross-over. Though these manipulations are executed by humans, the overall outcome of such a knowledge evolution process is usually unforeseeable.

Early versions of meme media middleware have been named IntelligentPad stressing that meme media objects appear like pads on a screen which can be directly manipulated by dragging and dropping and, thus, combining them to form new composite pads.

The most recent variant of meme media middleware is named Webble to abbreviate "Web-based lifelike entities". Webbles, so to speak, live in browser windows as illustrated by means of figure 2. The roots of the technology are due to Tanaka and Kuwahara

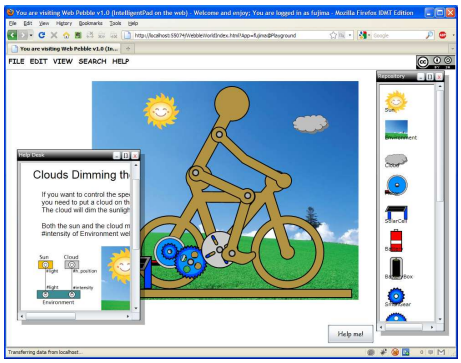

Figure 2: A virtual laboratory based on Webble technology.

(Kuwahara and Tanaka, 2009).

6 COMPUTABILITY WEBBLES

The authors have set up some environment to host a variety of media objects for computable functions and for operators which allow for the composition of any further computable functions. As will be shown below, the environment contains all basic functions and all operators necessary to define any computable function (Rogers jr., 1967), (Machtey and Young, 1978), (Péter, 1981).

Hence, this little environment is capable of representing anything computable. It implements the fundamentals of a full recursion theory accessible by means of direct media manipulation–computability theory at your fingertips.

6.1 Webbles for Basic Functions

The functions needed initially are

- the unary successor function s,
- all constant functions c_k^n of any arity n,
- all projection functions p_m^n selecting the mth element out of n given arguments (for any $m \leq n$).

The repository originally contains a few basic functions as on display in figure 3.

In case other initial functions are needed, they are easily generated by duplicating one of the original functions and, successively, setting parameters. For illustration, the lowest Webble shown represents p_1^2. It may be easily edited to select the second instead of the first input, that way becoming p_2^2, or to work for one more argument, that way becoming p_1^3.

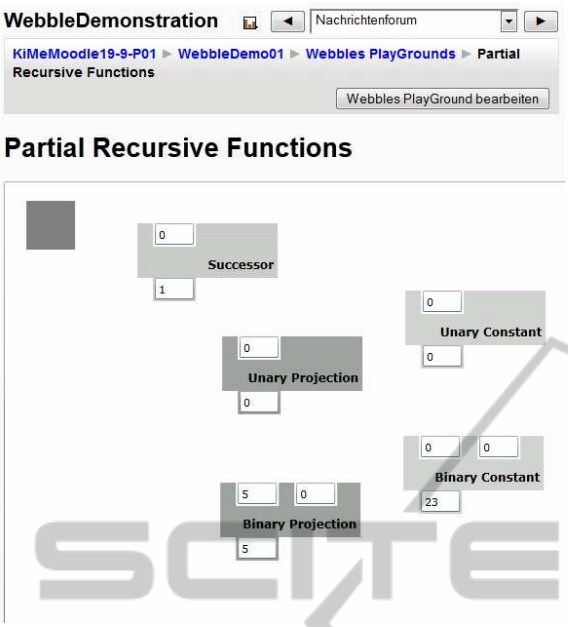

Figure 3: The initial functions of Primitive Recursion.

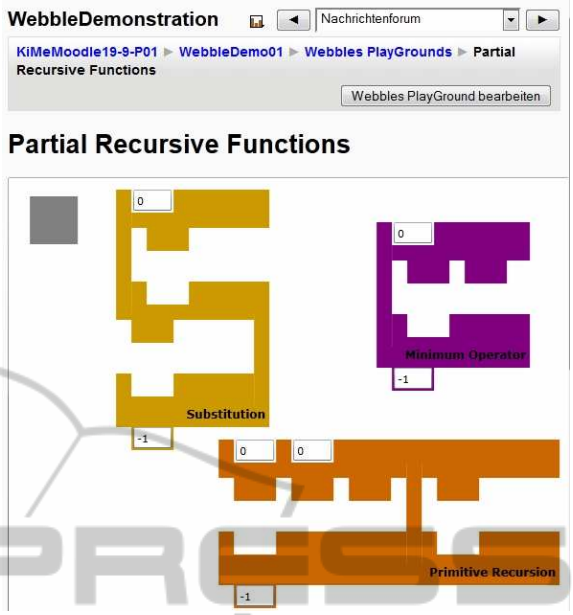

Figure 4: Operators necessary and sufficient for generating the complete collection of all Partial Recursive Functions.

6.2 Webbles for Operations

The set of all computable functions, usually named the partial recursive functions (Rogers jr., 1967), may be generated, if the three operators

- substitution of functions,
- limited, i.e. so-called primitive, recursion,
- minimum operator

are available.

Note that in figure 4 the values “-1” in the output slots indicate some failure, because the operators are not in use.

Substitution as shown above is prepared for the substitution of one unary function into another one. One may easily edit the scheme, for instance, to allow for the substitution of one binary function into one unary function or, alternatively, to allow for the substitution of two unary functions into some binary one. These two examples are on display in figure 5.

The other operators may be easily edited similarly.

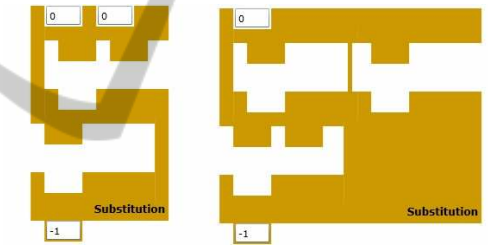

Figure 5: Two more variants of the substitution operator.

6.3 Direct Manipulation of Definitions

Direct manipulation, loosely saying, means that you are able to take a media object directly, move it to a place where you would like to have it, and perform what you want to do–turning around, rescaling, setting parameters, and the like–immediately. Using modern interfaces, you can do it with your own hands.

When students shall learn that for the purpose of defining a particular target function, one has to substitute some given original into another function given as well, they should be allowed to do it by hand, so to speak. The physical activity is assumed to support cognition (Barsalou and Wiemer-Hastings, 2005).

When using Webbles for direct manipulation, a standard activity is to drag and drop the one media object over the other to plug it in. See the figure 6 below for a simple illustration.

The figure displays a real screenshot of the authors’ learning environment within moodle in which three subsequent phases are overlayed in a single window, for the sake of comparison.

In phase one, the learner takes some media object (the one with the big dot at its left upper corner) to drag and drop it over another composite object into its apparently empty slot. Then, the object is latching automatically. The result is shown as a composite media object on the right side of figure 6–phase two

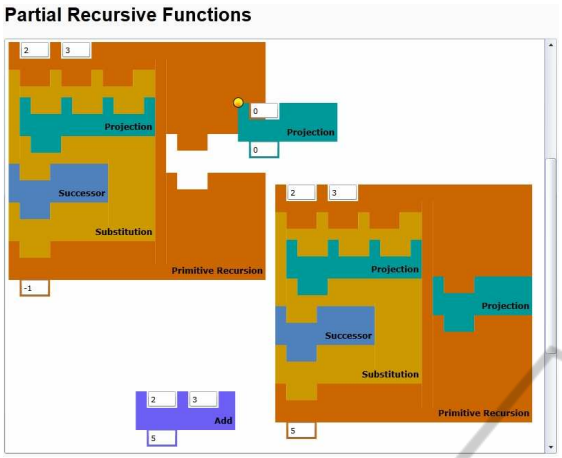

Figure 6: Phases of a function definition by drag & drop.

of the construction. With some click to the object, the learner may open a menu offering further opportunities. There is the action of grouping or, so to speak, hiding details. After performing this activity as phase three of the overall assembly process, the composite object appears as displayed at the bottom.

Learners may define any computable function in this way simply dragging and dropping elementary objects or those constructed before as intermediate stages of a more complex construction process.

To automatically lock objects in place is a very convenient support to the human learner, but several alternatives exist (Fujima, 2010).

According to the authors' experience (varying up to 25 years) in teaching computer science courses, a remarkable percentage of students have difficulties in understanding partiality of recursive functions theory. It is always a sticking point to define one's first partial recursive function that is not total. One has to think of processes that terminate sometimes, but not always.

It is crucial to employ the minimum operator–usually denoted by μ–appropriately. In the case of defining some unary function f, the terminology

$$f(x) \;=\; \mu y[g(x,y) = 0] \qquad (3)$$

means the value of f on input x is defined to be the smallest y such that $g(x,y)$ equals 0. Obviously, in case $g(x,y)$ is always greater than zero, the value $f(x)$ remains undefined. The minimum operator represents the most abstract concept of unbounded search.

In several exercises, the function *exact* is defined as follows

$$exact(x,y) \;=\; (sq(y) \dot{-} x) \;+\; (x \dot{-} sq(y)) \qquad (4)$$

where $sq(y)$ is the square of y and the subtraction in use yields zero if the second argument is not smaller than the first one. Students may be encouraged to play

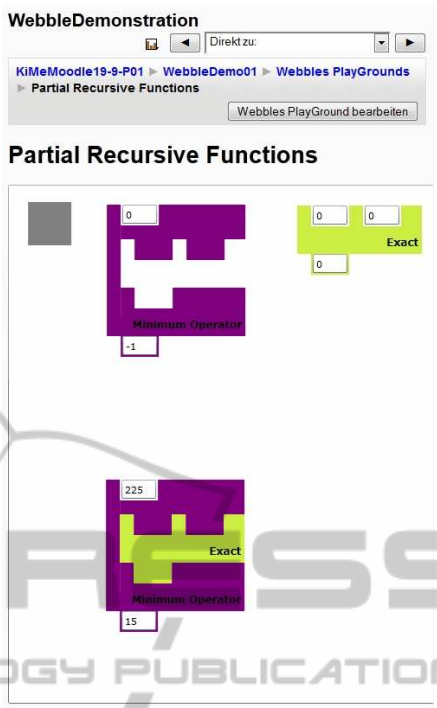

Figure 7: Exploring how the minimum operator works.

with some functions like this and with the minimum operator (see figure 7 above).

The advantage of direct manipulation becomes obvious. Learners can playfully explore the theory.

7 SCENARIOS OF PLAYFUL EXPLORATORY LEARNING

From some methodological point of view, the authors' approach towards learning of abstract concepts and of their properties and mutual relations may be seen as an experiment to support Barsalou's and Wiemer-Hasting's fourth hypothesis that "the content of abstract concepts could, in principle, be simulated" (see (Barsalou and Wiemer-Hastings, 2005), pp. 136/137). From a technological point of view, the approach may be understood as just another case of using Webble technology for playful learning (Fujima et al., 2010).

Direct manipulation may be fun (Xie et al., 2008) benefical to learning–even extremely abstract content.

7.1 Exploratory Learning

Let us take the illustration from figure 7 in the column on the left as an easy case study. The function object (top right) which is plugged into the minimum operator object has been constructed beforehand. Taking

the requirements of partial recursive function theory definitions (Péter, 1981) seriously, a formally complete specification of the mentioned function object requires an equational specification as shown in the Appendix.

Conventionally, the application of the minimum operator to the function *exact* would be written as

$$f(x) = \mu y[exact(x,y) = 0] \quad (5)$$

Would you as a reader, personally, like to compute the value $f(225)$, e.g., according to this definition taking the ten equations from the appendix as a basis? How easy is it to derive characteristics of the function f from its equational specification?

The authors are clearly in favor of exploratory learning which is supported by the operationality of composite media objects such as the one next to the bottom of figure 7.

Experimenting with media objects at your fingertips, you can easily find out that on input 225 you get the output 15 as shown in figure 7. Some additional explorations yield further insights such as, e.g., that there are outputs in response to the inputs 1 and 4, but no response to 2 and 3.

Interested learners might now go further and explore why there are no outputs in these cases. Some might inspect the defining equations in detail. Others might, instead, decompose the media object and inspect the data flow and transformation in its components–direct manipulation and exploration.

7.2 Game-based Learning

Webbles implementing partial recursive functions offer a large number of opportunities for competitive exercises and playful exploration, although the domain is extremely abstract and formulas are frequently cumbersome.

It is somehow exciting that most–if not all–of the deeper insights into computable functions theory lead to numerous specific exercises.

The fundamental KLEENE normal form theorem (see (Rogers jr., 1967), §1.10, p. 30) may serve as a convenient introductory example. Roughly, although applications of the minimum operator may be iterated several times, this basic theorem says that for any computable function there is no need for more than just one call of the minimum operator. The theorem may be summarized by the following simple equality for any computable function f of n arguments.

$$f(x_1,\ldots,x_n) = \alpha(\ \mu y[\ \beta(i,x_1,\ldots,x_n,y) = 0\ ]\ ) \quad (6)$$

Here, α and β are two universal functions definable without any application of the minimum operator and i is some index specific for f. By the way, α and β are just primitive recursive decodings and encodings.

It follows a certain generic exercise based on the KLEENE normal form theorem:

Provide any function definition using two or more calls of the minimum operator. Ask students for the simplest alternative Webble media object describing the same particular function, but using the minimum operator Webble (as shown in figure 7) at most once.

Competitions of this type are complicated by the problem to demonstrate that a certain proposed alternative is correct. In general, the equivalence problem for definitions of partial recursive functions is known to be undecidable (Rogers jr., 1967).

To sum up, competitive exercises as sketched here may lead to a variety of deeper problems such as, e.g., program equivalence which may be experienced by the learners in a quite playful manner. In this way, we return from competition to exploration.

In another game, one might proceed as follows:

- Peel off an inner media object.
- An opponent has to find another object for plug-in leading to an inequivalent definition.
- If the opponent fails, he looses. Otherwise, the players' roles are exchanged and it becomes the opponent's turn to peel off some object.
- The game ends when constructions are repeated.

To complete the present section, readers should recall that the current practice of teaching recursive function theory is far from being fun on both sides–the teachers and the students.

8 DIRECT MANIPULATION FOR LEARNING THE ABSTRACT IN RECENT HIGHER EDUCATION

Direct manipulation, in its early years, has been seen as a paradigm for alleviating severe bottlenecks of human-computer interaction (Shneiderman, 1982), (Shneiderman, 1983), (Hutchins et al., 1985). More recently, it has developed into a paradigm supportive to human understanding in highly involved studies. (Thalheim, 2008) provides some valuable approach to teaching SQL. Thalheim's approach, however, is much less operational than the present one, because his Visual SQL queries are not functional. They still need to be translated. In contrast, the authors' partial recursive functions Webbles are ready to run and they respond to any data input by processing the data.

The implementation of computability theory by means of Webble technology described in the paper's

preceding sections has been embedded into some Moodle learning environment for seamless curricular integration. It is currently in use in university level courses on Theoretical Computer Science.

In this environment, the students find a certain repository of Webbles sufficient to construct arbitrary computable functions. This provides, so to speak, computability theory at the students' fingertips.

There are exercises that enforce students' collaboration. In particular, students are encouraged to work together on a project exchanging composite Webbles.

In some sense, this may be seen as a Web 2.0 approach to the study of computability theory, because learners contribute their own constructs to some repository which may be used within the community.

In particular, there are tasks that establish contests and encourage students to compete for optimal solutions. That way some of the exercises are getting close to games and studying computability theory may become a bit playful.

Foremost, the implementation provides a practical case study in which learning of rather involved and far reaching abstract content is substantially supported by active learning based on direct media manipulation, exploration, and play.

Within the oral presentation of the present paper as well as in the related discussion, the authors will be able to report their recent experiences from teaching during the Summer term 2012.

However, it is the present authors' strong believe that experimentation and systematic evaluation, in particular, is a field of scientific work in its own right. This paper is a contribution to e-learning technologies introducing Webbles as a middleware for exploratory and playfully learning studying an abstract domain.

9 CONCLUSIONS & OUTLOOK

The present contribution is intended to set the stage for direct manipulation approaches to studies of highly abstract domains such as computability theory. The basics have been prepared and some complete implementation–including its integration into some Moodle environment–has been provided. Furthermore, a couple of scenarios of exploration and playful competition have been developed.

This way, the authors are facing now systematic experimentations and evaluations. The results should be published in some subsequent paper, but go beyond the limits of the present short paper.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the cooperation with a larger number of scientists and engineers who contributed to the development of IntelligenPad and Webble technologies providing cases of application. Micke Kuwahara deserves particular thanks.

Furthermore, they gratefully acknowledge the support by colleagues to introduce the development reported in the present paper into higher education practice including some students providing feedback. André Schulz is the one who has always kept our computability Webbles running in Moodle.

REFERENCES

Barsalou, L. W. and Wiemer-Hastings, K. (2005). Situating abstract concepts. In Pecher, D. and Zwaan, R. A., editors, *Grounding Cognition: The Role of Perception and Action in Memory, Language, and Thought*, pages 129–163. New York: Cambridge University Press.

Blackmore, S. (1999). *The Meme Machine*. Oxford University Press.

Church, A. (1936). An unsolvable problem of elementary number theory. *The American Journal of Mathematics*, 58:345–363.

Dawkins, R. (1976). *The Selfish Gene*. Oxford University Press.

Fujima, J. (2010). Auto-connection mechanisms for educational virtual laboratory. In *The IET International Conference on Frontier Computing, Proceedings (DVD), Taichung, Taiwan, August 4-6, 2010*, pages 396–401.

Fujima, J., Hawlitschek, A., and Hoppe, I. (2010). Living machinery – Advantages of Webble technologies for teaching and learning. In *Proc. 2nd International Conference on Computer Supported Education, CSEDU 2010, Valencia, Spain, April 7-10, 2010*.

Gödel, K. (1931). Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. *Monatshefte für Mathematik und Physik*, 38:173–198.

Gray, J. (2000). The Hilbert problems 1900-2000. *Newsletter of the European Mathematical Society*, 36:10–13.

Hutchins, E. L., Hollan, J. D., and Norman, D. A. (1985). Direct manipulation interfaces. *Human-Computer Interaction*, 1:311–338.

Kleene, S. C. (1936). General recursive functions of natural numbers. *Mathematische Annalen*, 112(5):727–742.

Kuwahara, M. and Tanaka, Y. (2009). Webble portal. http://www.meme.hokudai.ac.jp/WebbleWorldPortal/.

Machtey, M. and Young, P. (1978). *An Introduction to the General Theory of Algorithms*. New York, Oxford, Shannon: Elesevier North-Holland.

MacKenzie, C. L. and Iberall, T. (1994). *The Grasping Hand*. Amsterdam: North-Holland.

Péter, R. (1981). *Recursive Functions in Computer Science*. Akadémia Kiadó, Budapest.

Post, E. L. (1936). Finite combinatory processes. formulation I. *The Journal of Symbolic Logic*, 1:103–105.

Rogers jr., H. (1967). *Theory of Recursive Functions and Effective Computability*. McGraw-Hill.

Shneiderman, B. (1982). The future of interactive systems and the emergence of direct manipulation. *Behavior and Information Technology*, 1:237–256.

Shneiderman, B. (1983). Direct manipulation: A step beyond programming languages. *IEEE Computer*, 16:57–69.

Siekmann, J. and Wrightson, G., editors (1983a). *Automation of Reasoning 1. Classical Papers on Computational Logic 1957–1966*. Berlin, Heidelberg, New York: Springer-Verlag.

Siekmann, J. and Wrightson, G., editors (1983b). *Automation of Reasoning 2. Classical Papers on Computational Logic 1967–1970*. Berlin, Heidelberg, New York: Springer-Verlag.

Tanaka, Y. (2003). *Meme Media and Meme Market Architectures: Knowledge Media for Editing, Distributing, and Managing Intellectual Resources*. IEEE Press & Wiley-Interscience.

Thalheim, B. (2008). Visual SQL: Towards ER-based object-relational database querying. In Li, Q., Spaccapietra, S., Yu, E., and Olivé, A., editors, *Proceedings ER 2008, 27th International Conference on Conceptual Modeling, Barcelona, Spain, October 20-24, 2008*, number 5231 in LNCS. Springer Verlag.

Turing, A. M. (1936). On computable numbers with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42(7):230–265.

Xie, L., Antle, A. N., and Motamedi, N. (2008). Are tangibles more fun? Comparing children's enjoyment and engagement using physical, graphical and tangible user interfaces. In *Proceedings of the Second International Conference on Tangible and Embedded Interaction (TEI'08), Feb 18-20, 2008, Bonn, Germany*, pages 191–198. New York, NY, USA: ACM.

APPENDIX

$$add(x,0) = p_1^1(x) \tag{7}$$

$$add(x,s(y)) = s(p_3^3(x,y,add(x,y))) \tag{8}$$

$$mult(x,0) = c_0^1(x) \tag{9}$$

$$mult(x,s(y)) = add(p_1^3(x,y,mult(x,y)),p_3^3(x,y,mult(x,y))) \tag{10}$$

$$sq(x) = mult(x,x) \tag{11}$$

$$pred(0) = 0 \tag{12}$$

$$pred(s(y)) = p_1^1(y) \tag{13}$$

$$subtr(x,0) = p_1^1(x) \tag{14}$$

$$subtr(x,s(y)) = pred(p_3^3(x,y,subtr(x,y)) \tag{15}$$

$$exact(x,y) = add(subtr(p_1^2(x,y),sq(p_2^2(x,y))),subtr(sq(p_2^2(x,y)),p_1^2(x,y))) \tag{16}$$