Discrete mathematics at university level. Interfacing mathematics, computer science and arithmetic

Cécile Ouvrier-Buffet, Antoine Meyer, Simon Modeste

Discrete mathematics at university level
Interfacing mathematics, computer science and arithmetic

Cécile Ouvrier-Buffet[1], Antoine Meyer[2], and Simon Modeste[3]

[1]URCA, CEREP, Reims, France, cecile.ouvrier-buffet@univ-reims.fr;

[2]Université Paris Est - Marne-la-Vallée, LIGM, France, antoine.meyer@u-pem.fr;

[3]IMAG, UMR CNRS. Université de Montpellier, simon.modeste@umontpellier.fr

Discrete mathematics is a recent field linked with Computer Science. We discuss its place in university mathematics curricula and in the particular case of France, where it has difficulties to find its place. We make explicit the didactical challenges posed by discrete mathematics at university level, and present DEMIPS network and its plans to tackle them. Through two detailed examples we discuss the reasons for teaching Discrete Mathematics at university level, and illustrate our conclusions.

Keywords: teaching and learning of number theory and discrete mathematics, teaching and learning of mathematics in other fields, proof, algorithms.

INTRODUCTION

This paper points out the current need for the construction of resources and debates regarding discrete mathematics at university level. We wish to emphasize the features of the French context, both from an educator's and researcher's point of view, at the intersection of didactics, mathematics and computer science. Indeed, teaching and learning discrete mathematics involves mathematical skills and heuristics (e.g. different kinds of proofs and reasoning, several ways of modelling etc.)[1] and also develops objects, concepts, methods and tools that are necessary for computer science. This link with computer science brings new types of questions to mathematics (for instance, regarding algorithmic complexity). Then, our aims are to design original situations for schools and at university level, and to construct appropriate introductory situations for computer science and maths majors.

We propose an overview of discrete mathematics in mathematics education and make a focus on the interface between discrete mathematics and computer science. Then, after presenting our research group in France, we analyse two kinds of situations.

DISCRETE MATHEMATICS IN AND FOR MATHEMATICS EDUCATION

How to define discrete mathematics?

Several mathematical topics are often gathered under the blanket term *discrete mathematics*. A first step in contributing to a thorough didactical study of discrete mathematics is to provide a satisfactory definition, or at least delimitation, of what it refers to. Several definitions exist, which either attempt to provide a general common trait to the covered topics, such as "the mathematics of discrete sets", or resort to an enumeration of objects, concepts or techniques most often associated with discrete mathematics. Most of these definitions include or are followed by a discussion on some typical difficulties, such as the distinction between finite, discrete and continuous mathematics (e.g. Maurer, 1997). To clarify the distinction between finite and discrete mathematics, the MAA (1992) places finite mathematics in the pre-calculus category and discrete mathematics in the same category as calculus.

We advocate that an interesting way to define discrete mathematics both for research and didactical perspectives (for the design of courses and of didactical engineering at university level) is to emphasize the features of the modes of reasoning that are common (or specific) to the various topics usually recognized as discrete mathematics, and the discrete nature of the structures they involve. Moreover, a classification of problems is required in order to structure a didactical analysis of the field of discrete mathematics. Furthermore, as the development of discrete mathematics has been strongly directed by the needs for computer science, the links with computer science must be explicitly explored.

In 1974, Knuth, a pioneer in computer science and its teaching made a similar analysis (Knuth, 1974, p. 329) :

> "The most surprising thing to me, in my own experiences with applications of mathematics to computer science, has been the fact so much of the mathematics has been of a particular discrete type [...]. Such mathematics was almost entirely absent from my own training, although I had a reasonably good undergraduate and graduate education in mathematics. [...] I have naturally been wondering whether or not the traditional curriculum – the calculus courses, etc. – should be revised to include more of these discrete mathematical manipulations, or whether computer science is exceptional in its frequent application of them."

We consider that these questions are still topical, even at university level, and deserve a careful didactical analysis.

Where is discrete mathematics? What questions are relevant at university level?

It is often stated that discrete mathematics can be a tool for improving reasoning and problem-solving skills (see for instance Rosenstein, Franzblau & Roberts (1997) who advocated an introduction of discrete mathematics in curricula, asked didactical questions, and made propositions that were taken into account for *Principles and Standards for School Mathematics* NCTM, 2000 for instance). Moreover, discrete mathematics is an active modern branch of contemporary mathematics with a wide range of applications in society, which is a very legitimate reason to teach it at school, high school and college. In fact, discrete mathematics courses are relevant to a wide variety of majors at university level, including mathematics, number theory, computer science, and engineering: from an epistemological point of view, discrete mathematics has an interdisciplinary nature and can provide a mathematical foundation (with specific ways of reasoning and proving, and mathematical concepts) for computer science and engineering courses. By 1989, an MAA report (Ralston, 1989) from an *ad-hoc* committee consisting of mathematicians and computer scientists recommended that "discrete mathematics should be part of the first two years of the standard mathematics curriculum at all colleges and universities”. This report also emphasizes the notions of proof, recursion, induction, modelling, and algorithmic thinking, as well as the benefits of using discrete mathematics in the secondary level to improve problem-solving skills with the transition to university level in mind (Ralston, 1989). Moreover, Epp (2016) points out the strong necessity of abstract thinking for the course and its applications in computer science. She underscores that it is done in the frame of the current curricular recommendations, prepared by The Joint Task Force on Computing Curricula (2013) of the ACM and the IEEE Computer Society, which gives discrete mathematics as one of the two largest components in the “core body of knowledge” recommended for all computer science students. Besides, discrete mathematics is in close relationship with other mathematical areas: other fields of mathematics use its methods and results, and, are useful for solving some discrete mathematics problems.

What is currently the place of discrete mathematics and its links with other scientific fields at university level? In several countries (Hungary, USA, Germany for instance), its significance in university programs is well-established and acknowledged. That is not always the case in France where the status of discrete mathematics in the first three university years is unclear, at least in mathematical curricula. However, discrete mathematics appears sporadically in few parts of mathematics curricula as probability theory (in particular combinatorics for discrete probability theory) or arithmetic. It sometimes appears in courses dedicated to the learning of proving, mathematical reasoning and problem solving, but we question whether its specificity is emphasized. One is more likely to find courses where discrete mathematics is taught for itself in computer science or mathematics and computer science curricula, where there exists a kind of common basis shared between teachers and including classical contents of discrete mathematics as can be seen abroad. These reports and recommendations coming from academic societies and the above remarks underscore two key questions for mathematics education at university level, and more specifically in France:

- What are the place and role of discrete mathematics at university level? How to design curricula and didactical engineering for the first university years ?
- What links are there between discrete mathematics and other areas (mainly of mathematics and computer science) and how are they (or should they be) practised / worked in the first university years?

These questions are particularly crucial for countries where discrete mathematics does not have a well-established status is the first university years.

What do we know from a didactical point of view?

In mathematics education, various research regarding the teaching and learning of discrete mathematics exist, focusing mainly on the primary and the secondary levels (ZDM (2004), Hart & Sandefur (in press) propose overviews). This research meets general approval, and points out epistemological features of discrete mathematics such as: discrete problems bring out different ways of proving (Grenier & Payan, 1998); discrete structures enable work on the construction of mathematical models, optimization, operational research and experimental mathematics (e.g. Grenier & Payan, 1998; Maurer, 1997); discrete concepts are accessible and problems are easy to understand (Grenier & Payan, 1998; De Bellis & Rosenstein, 2004); discrete concepts have different kinds of definitions and representations (Ouvrier-Buffet, 2006, 2011); some discrete problems are real world problems developing and using techniques from mathematics and computer science (Schuster, 2004), etc. Discrete mathematics problems are also a frame for developing and teaching algorithms; conversely, the study of algorithms requires a lot of discrete mathematics, and studying algorithms and programming can be a good way to justify the introduction of discrete mathematics contents (e.g. Modeste, 2012 & 2016). In all this research, discrete mathematics seems to be a powerful source of problems for teaching and learning mathematical proofs and processes and engaging students in developing new ways of thinking (such as recursive thinking), heuristics and problem-solving skills from primary school to university. Besides, some researchers point out that its teaching provides opportunities to bypass some of the sources of commonly-occurring negative affect in students (e.g. Goldin, 2016).

It appears that the features of discrete mathematics clearly represent challenges for university mathematics, in particular in France.

THE "DEMIPS" NETWORK – A WAY TO FEDERATE DISCRETE MATHEMATICS EDUCATION

Presentation of the DEMIPS network

In the French framework of mathematics education, there is a need to federate (isolated) research in university mathematics. Following the INDRUM momentum, the national network DEMIPS[2] supports the development of new research programs. DEMIPS's research involves around 40 researchers in mathematics, mathematics education, physics education, computer science, and epistemology and history of mathematics and sciences, and is concerned with five main topics: three topics dealing with mathematical contents (analysis; linear algebra and abstract algebra; arithmetic, discrete mathematics and algorithmics) at the secondary – post secondary transition and at university level (the links with physics and computer science are questioned); a transversal topic (logic, language, reasoning, proofs - from both a mathematics and computer science point of view); and a specific topic dealing with the practices of teachers and teachers-researchers at university level (in mathematics, computer science and physics).

We (the authors of this paper) organize federative research in the fields of arithmetic, discrete mathematics and algorithms. The members of our group are mathematics educators (didacticians) with specific skills in teaching and learning at university level, mathematicians, and researchers in computer science. We choose to study the parts of mathematics which lie at the intersection of "classical" mathematics and theoretical computer science (for instance discrete mathematics, arithmetic, and algorithms), which interact and complement each other. As theoretical background we will follow Brousseau's theory of didactical situations (Brousseau, 1998) for its notion of didactical engineering, and the notion of scheme (Vergnaud, 1990) in order to structure our analysis of mathematical concepts. We organize our questions around key axes regarding the French university level:

- What are the epistemological features of concepts and reasoning in arithmetic, discrete mathematics and algorithms? How do they interact? (And then, how can these interactions be used to enrich the way these concepts are taught?)
- What kind of situations can one design in these mathematical areas for the university level and for pre-service and in-service teacher training? What for?
- What kind of curricula are there for this kind of mathematics at university level? What can be said about the design of these curricula?

Our research questions try to break down the barriers between scientific disciplines involving discrete mathematics. They also underline typical situations and questions common to mathematics and computer science, and try to put to use didactical analysis techniques to cast a new light on the way these questions are, or could be, tackled at university level. We develop below two examples to illustrate our work, and elaborateon the place and role of discrete mathematics at university level.

SITUATIONS AND IMPLEMENTATIONS AT UNIVERSITY LEVEL - EXAMPLES FROM DEMIPS' WORKSHOPS

We develop here two examples to illustrate the potentialities of discrete mathematics to engage students in learning modelling, proving, and mathematical reasoning and also to underscore the validity and the interest of keeping in mind the algorithmic point of view and the connections with computer science. These examples emphasize new perspectives for the teaching and learning of mathematics. The first example explore the links between mathematics and computer science in a problem-solving context and the second deals with a classical "divide and conquer"-type algorithm.

Example 1 – Discrete lines

The mathematical object concerned here is the discrete straight line (colouring squares, or "pixels", on a regular rectangular grid, in order to give the best possible visual impression of a straight line). The (real) straight line can act as a referent. Discrete straight lines are accessible through their representations (e.g. perceptive and analytical aspects of geometry) and their definitions and properties are non-institutionalized (a concept is institutionalized if it has a place in a classically taught content). Computer programmers are familiar with this concept. Professional researchers in discrete geometry (both mathematicians and computer scientists) use several definitions, but the proof of the equivalence of these definitions remains worth considering. The complexity of the underlying axiomatization of discrete geometrical concepts is actually an open and interesting problem.

Ouvrier-Buffet (2006) has analysed the evolution potential of zero-definitions (in Lakatos' sense, zero-definitions act as working definitions) of the concept "discrete straight line" in a defining situation implemented with freshmen. She underscores several approaches dealing with this concept, namely "real straight line" (What is the "nearest" pixel to a real line? What kind of modelling should be used?), "regularity" (What are the properties of the sequence of stages (called *chaincode string*)?), and "axiomatization" (What about the existence of the intersection of two discrete straight lines? Is a discrete straight line unique?). Each point of view brings about several zero-definitions. To engage into an axiomatic perspective carries great difficulty. This approach deals with both the perceptive aspect of a straight line and the axiomatic perspective. We are here confronted with two markedly different defining styles: a local one and a global-theoretical one, the latter mobilizing some implicit skills and knowledge in students (e.g. building a theory and choosing among competing definitions). The main results of this experiment underscore the ability of students to engage in a defining activity with a "neutral" but complex concept. Students do not assume an axiomatic perspective but mobilize reasoning involving approximate methods close to those used for real straight lines (and then arithmetic tools) and also the characterization of the sequence of stages of pixels (how can we modify a sequence to obtain a better regularity?) that involves recursive arguments.

From a didactical point of view, this research requires the development of a new theoretical background in order to model the defining process. From a mathematical point of view, the discrete geometrical objects, and more specifically the discrete straight lines can be approached in several ways: differential discrete analysis, the Bresenham algorithm, algorithms involving combinatorial analysis, several discretizations using algorithms which generate and study errors (Greene & Yao, Freeman & Pham, Rosenfeld), and the introduction by Reveillès of the arithmetical definition of a discrete straight line (1991). For instance, the approach to the discretization of a real straight line by checking linearity conditions is directly related to number theoretical issues in the approximation of real numbers by rational numbers. These linearity conditions can be checked incrementally, leading to a decomposition of arbitrary strings into straight substrings (Wu, 1982). The ongoing mathematical problems in discrete geometry are intimately related to questions in other fields of mathematics and computer science. The construction and the manipulation of algorithms are important for this purpose.

Example 2 – Exponentiation by squaring

A classic algorithmic problem is that of computing for some natural n the n-th power an of real number a. A naïve solution is, starting with value 1, to multiply n times this value by a. The final value one obtains is indeed the expected result, which is not very difficult to establish. The fact that this algorithm terminates is also trivially true since it contains a single bounded repetition. Finally, the complexity of this computation is clearly in Θ(n) (i.e. asymptotically bounded above and below by n), counting for instance the number of multiplications performed, and assuming that multiplication by a is an elementary operation.

This algorithm is not very efficient, considering that its running time is actually *exponential* in the representation size of n (which is of the order of log(n)). A more efficient technique relies on the observation that $a^n=(a^2)^{n/2}$ if n is even, otherwise $a^n=a.(a^2)^{(n-1)/2}$. Written as a recursive Python function, this algorithm reads as follows[3]:

```
def power(a, n):
  if n == 0:
    return 1
  elif n % 2 == 0:
    return power(a * a, n // 2)
  else:
    return a * power(a * a, n // 2)
```

We will now study a few common questions asked about algorithms, which will allow us to illustrate examples of mathematical techniques relevant to the analysis of, and discussion about, algorithms. In the following, typewriter face (as in n) will be used for formal parameters, and italic (as in n) to denote actual values.

Termination. A first question when it comes to analysing an algorithm is to determine whether or not it terminates, i.e. whether its execution on any instance of the problem (i.e. any pair (a, n)) yields a result after a finite number of execution steps or *elementary operations*. A standard technique used to prove this kind of result in non-trivial cases is the following. Assume here that there exist a_0 and n_0 such that power(a_0, n_0) performs infinitely many recursive calls. Call n_i the value of parameter n on the i-th recursive call. The sequence of naturals $(n_i)_{i\geq 0}$ is strictly decreasing, because whenever $n_i>0$, n_{i+1} is the quotient of n_i by 2, rounded down. This contradicts the fact that infinitely many calls are made, which means that the value of n eventually has to reach 0 and the function must terminate for all values of a and n.

Correctness. It remains to prove that the result is indeed correct for any instance of the problem. This is often done using some form of induction due to the intrinsically discrete and recursive or iterative nature of algorithms. In our case, we will establish that the value returned by a call to power(a, n) is indeed a^n, via a simple recurrence on the *call depth*, which is the maximal number, say k, of generated nested calls. The base case (k=0) is obvious: since there is no recursive call it must mean that n=0 and the returned is indeed $1 = a^n$. In the inductive case, assume the property holds for call depth k and consider a call of maximal depth k+1. Necessarily n must be greater than 0. If n is even, n//2 evaluates to n/2, a single nested call power(a*a, n//2) is performed and the obtained value is returned directly. This call itself has call depth exactly k, therefore by induction hypothesis its return value is $(a^2)^{n/2}=a^n$. Similarly if n is odd, n//2 evaluates to (n-1)/2, the value returned by power(a*a, n//2) is, by induction hypothesis, $(a^2)^{(n-1)/2}=a^{n-1}$, and the value returned by the main call is a*power(a*a, n//2), which evaluates to a^n. Therefore by the recurrence principle, the function returns the correct value whatever the initial value of its parameters.

Complexity. In the study of termination, we observed that in a call power(a, n), the value of n for the next call (if there is one) is divided by two (rounded down). One may observe the successive values of n more easily when it is written down in binary. Indeed, the operation of dividing a number by two and rounding down corresponds, in binary representation, to erasing its rightmost digit. The algorithm stops when n is 0, and performs one recursive call otherwise, modifying its value as we just saw. The number of nested calls for some initial value of n is therefore equal to the length, say k, of its binary representation, in other words its number of digits. Moreover, when n is even, exactly one multiplication is performed in the current call, two when it is odd. Therefore, denoting by m the number of digits equal to 1 in the binary representation of n, the total number of multiplications performed by the power(a, n) is exactly k+m, which is asymptotically bounded above by $\log_2(n)$.

Summary. We chose this example to illustrate, on a simple problem, the type of questions which can be asked about algorithms and the methods which are likely to be used to answer them. Note that in this simple case, all three properties could have been proven simultaneously using a complete recurrence on n. For our purpose, we chose a more basic and detailed approach. It would have been interesting to show how these proofs could be rephrased in the context of an iterative function. This example also tries to advocate the necessity for students in mathematics, computer science and related topics to have at least a basic understanding of various flavours of recursion and induction (including basic properties of orderings), to be able to present rigorous proof arguments (at least informally), and to possess minimal fluency in arithmetic, in order to be able to envision algorithms as objects worth studying in their own right. It is moreover often the case that the study of algorithms provides insight on related mathematical objects (here, the relationship between the value of a number and the length of its binary representation). Finally, this example illustrates a typical preoccupation of algorithmics, which is to provide more efficient, sometimes even optimal, algorithmic solutions to problems.

DISCUSSION AND CONCLUSIONS

Discrete mathematics is now considered as an entire field of mathematics, with many links to computer science. While it has entered university curricula in many countries, its status and contour are not always clear, and there are countries (such as France) where it has difficulties finding a legitimate place. Through the two examples we have developed (the discrete line and exponentiation by squaring), we have illustrated that it is legitimate to question the place that discrete mathematics occupies in university mathematics, for different reasons:

- it allows to develop situations for mathematical reasoning, mathematical heuristics, and problem solving (by its nature, but also by contrast with traditional continuous mathematics),

- many objects and techniques of discrete mathematics are required knowledge for computer science curricula; these contents must be identified and analysed from a didactical point of view, to design appropriate activities and situations,
- discrete mathematics involves specific questions and types of problems (such as complexity questions, combinatorial problems, etc.) that must be studied in order to understand their place in university curricula.

The DEMIPS network, through the topic group *arithmetic, discrete mathematics and algorithmics*, aims at addressing these questions. We pointed out the necessity to develop a didactical research on the topic of discrete mathematics at university level and its articulation with other fields of mathematics and other disciplines. This didactical research must rely on an institutional analysis of the situation in universities, and most importantly on a thorough epistemological study of discrete mathematics and its specific branches. It also requires to select and develop appropriate theoretical frameworks. Such work, started in the DEMIPS topic group, requires a plurality of viewpoints and interactions between (discrete) mathematicians, computer scientists, and didacticians of mathematics.

NOTES

1. Problems that can be identified as belonging to discrete mathematics can be found in many books aiming at developing “mathematical thinking”, such as (Mason, Burton & Kaye, 1985).

2. Didactique et Epistémologie des Mathématiques, et liens Informatique et Physique dans le Supérieur: Didactics and Epistemology of Mathematics, and links with Computer Science and Physics in University Mathematics - with the support of CNRS.

3. Here a is assumed to range over floats, and n over positive integers. Note that in Python 3, n//2 computes the quotient of n by 2, whose value is n/2 if n is even and (n-1)/2 otherwise.

REFERENCES

Brousseau, G. (1998). *Théorie des Situations Didactiques.* La Pensée Sauvage.

DeBellis, V. A., & Rosenstein, J. G. (2004). Discrete Mathematics in Primary and Secondary Schools in the United States. *ZDM, 36 (2)*, 46-55.

Epp, S.S. (2016). Discrete mathematics for computer science. *Proceedings of the 13th International Congress on Mathematical Education - TSG 17*. Hamburg.

Goldin, G.A. (2016). Discrete mathematics and the affective dimension of mathematical learning and engagement. *13th ICME - TSG 17*. Hamburg.

Grenier, D., & Payan, C. (1998). Spécificités de la preuve et de la modélisation en mathématiques discrètes. *RDM*, *18*(1), 59-100.

Hart, E.W & Sandefur, J. (in press). *Teaching and Learning Discrete Mathematics in the School Curriculum Worldwide*. An ICME-13 Monograph. Springer.

Heinze, A., Anderson, I., & Reiss, K. (2004). Discrete Mathematics and Proof in the High School, Introduction. *ZDM, 36 (2)*, 44-45.

Knuth, D. E. (1974). Computer Science and its Relation to Mathematics. *The American Mathematical Monthly*, *81*(4), 323-343.

MAA (1992). Statistical abstract of undergraduate programs in the mathematical sciences and computer science in the United States, 1990-1991, CBSM Survey, First Two Years, *MAA notes n°23*, Washington DC.

Mason, J., Burton, L., & Kaye, S. (1985). *Thinking Mathematically*. Workingham: Addison-Wesley Publications.

Maurer, S. B. (1997). What is Discrete Mathematics? The many answers. *Discrete Mathematics in the Schools* (pp. 121-132). American Mathematical Society.

Modeste, S. (2012). Enseigner l'algorithme pour quoi ? Quelles nouvelles questions pour les mathématiques ? Quels apports pour l'apprentissage de la preuve ? (PhD dissertation). Université de Grenoble.

Modeste S. (2016), Impact of informatics on mathematics and its teaching. On the importance of epistemological analysis to feed didactical research, in Gadducci, F., Tavosanis, M. (Eds.) History and Philosophy of Computing, Series : IFIP Advances in Information and Communication Technology, Vol.487 (Springer).

Ouvrier-Buffet, C. (2006). Exploring mathematical definition construction processes. *Educational Studies in Mathematics, 63*(3), 259-282.

Ouvrier-Buffet, C. (2011). A mathematical experience involving defining processes: in-action definitions and zero-definitions. *Educational Studies in Mathematics, 76*(2), 165-182.

Ralston, A. (1989). Discrete Mathematics in the First Two Years. M.A.A. Notes, 15.

Rosenstein J.G, Franzblau D.S., Roberts F.S (Eds) (1997) *Discrete Mathematics in the Schools*. DIMACS Series in Discrete Mathematics & Theoretical Computer Science Vol. 36. American Mathematical Society & NCTM.

Schuster, A. (2004). About Traveling Salesmen and Telephone Networks - Combinatorial Optimization Problems at High School. *ZDM, 36 (2)*, 77-81.

The Joint Task Force on Computing Curricula, Association for Computing Machinery (ACM), IEEE Computer Society (2013). *Computer Science Curricula 2013*, ACM publication.

Discrete mathematics and Proof in the High School (2004). *Zentralblatt für Didaktik der Mathematik*, Vol. 36 (2), 44-84 and Vol. 36 (3), 82-116.

Vergnaud, G. (1990). La théorie des champs conceptuels. *Recherches en Didactique des Mathématiques*, *10*(2.3), 133-170.

Wu, L.D. (1982). On the chaincode of a line. *IEEE Trans. on Pattern Anal. Machine Intell.* PAMI 4, 347–353.