

Discrete Mathematics/Structures: How Do We Deal With the Late Appreciation Problem?

David Gries	Ali Erkan ¹	Michael Eckmann	James Heliotis
Professor	Assistant Professor	Assistant Professor	Professor
Computer Science	Computer Science	Computer Science	Computer Science
Cornell University	Ithaca College	Skidmore College	RIT
gries@cs.cornell.edu	aerkan@ithaca.edu	meckmann@skidmore.edu	jehcs.rit.edu

Panel

One big challenge of teaching discrete mathematics/structures (“discrete math” from here on) is not students’ appreciation of the material but the timing of this appreciation. Some (many?) students ask “*Why do we have to learn all this?*” as they study sequences, mathematical induction, counting, probability, etc., relatively early in their college years. Unfortunately, the most convincing answer comes during the subsequent semesters, and it comes in many iterations. Aside from the courses that obviously depend on this branch of mathematics (i.e. Algorithms, Data Structures, etc.), concepts and notation first introduced in discrete math are utilized in technical treatments of computer networking ([1], [5]), computer organization [2], natural language processing (for probabilistic arguments), web sciences (for network structure arguments [3]), modeling, etc.

Individually, we develop ways to deal with this issue but our efforts should be organized as a group as well. This panel will start with a definition of the problem followed by the techniques and experiences of the panelists, each representing a different type of school. We plan to address the following questions:

- When is the right time to teach discrete math? Should it be an introductory course? Could the requirement to take discrete math as a co-requisite to a course that frequently makes use of mathematical ideas (such as data structures) increase its appreciation?
- How should we manage the trade-off between rigor and motivation? How should we manage the trade-off between breadth and depth?
- Could a two-course sequence be a solution to late appreciation? Or, going to another extreme, could discrete math be eliminated as a course and all its content be taught in a hands-on manner in the right context in other courses?
- Could a hands-on component reveal to students that discrete math is not limited to a mathematical plane?
- Could there be a “Greater Ideas in Computer Science” course that is driven by topics in discrete math?
- Is there a consensus between Computer Science faculty members as to what hinges on the proper conduct of this course?

¹Contact Person

The panelists will speak only as much as to get an interesting discussion going. With the assumption that the conversation creates enthusiasm, we also plan to create a wiki that is seeded with the highlights of the panel. This wiki can then grow beyond the conference and serve as a repository for anyone who is interested in teaching discrete math.

Panelists

Dr. David Gries represents a Ph.D. granting institute as well as a different philosophy on discrete math. As he explains in [4], logic should be taught as a useful tool for people, which is then used in teaching all other topics –set theory, recurrence relations, theory of integers, etc. At the same time, interesting applications *can* provide the needed motivation.

Dr. Ali Erkan represents an undergraduate institute that serves a mixed population of students (humanities & science + professional schools). Since Fall 2004, he has taught discrete math as well as numerous courses that depend on it (data structures, computer organization, computer networks, network modeling, algorithms, structure and function of complex networks, scientific computing). He has been a sponsor of research oriented student projects that reveal the applicability of discrete math in a diverse range of fields (e.g. application of recurrence relations in deriving the closed forms of recurrent integration rules).

Dr. Michael Eckmann, representing the liberal arts college environment, has been teaching Computer Science courses for five years. During this time, he has taught discrete math as well as courses that depend on it (CS I, CS II, Computer Graphics). His perspective includes a recent restructuring of Skidmore’s discrete math to serve mathematics + Computer Science majors and the development of an “intensive writing” course based on proof writing.

Dr. James Heliotis, representing a technical institute, has interests in computer science education, software architecture, design patterns, aspect-oriented design and implementation, programming languages and tools, and real-time systems. He has industrial experience in both software and electrical engineering. His reflections on discrete math includes the consequences of having to work with a course offered by RIT’s mathematics department.

References

- [1] D. P. Bertsekas and R. Gallager. *Data Networks*. Prentice Hall, second edition, 1991. ISBN: 0132009161.
- [2] R. E. Bryant and D. R. O’Hallaron. *Computer Systems: A Programmer’s Perspective*. Prentice Hall, 2002.
- [3] D. S. Callaway, J. E. Hopcroft, J. M. Kleinberg, M. E. J. Newman, and S. H. Strogatz. Are randomly grown graphs really random? Technical report, Santa Fe Institute, 2001.
- [4] D. Gries and F. B. Schneider. *A Logical Approach To Discrete Math*. Springer Verlag, 1993.
- [5] A. Leon-Garcia and I. Widjaja. *Communication Networks: Fundamental Concepts and Key Architectures, 2nd Edition*. McGraw Hill College Div, 2003.