Course Name: Mathematical Foundations of Computer Science

Course Number: CS208

Credits: 3-1-0-4

Prerequisites: IC-150 Computation for Engineers

Intended for: UG

Distribution: Compulsory for CSE; CS elective for EE and ME

Semester: 3rd

Preamble: The proposed new curriculum for CSE includes six discipline core courses:

1. Mathematical Foundations of Computer Science
2. Advanced Data Structures and Algorithms
3. Paradigms of Programming
4. Computer Organization
5. Information Systems
6. Communicating Distributed Processes

The proposed course follows the new CS curriculum design approach that requires covering all the mathematical concepts that are fundamental for entire CS in one basic course. For example, it includes some basic concepts about Finite State Machines (FSM), which are usually covered in a course on Formal Languages and Automata Theory (FLAT). However, as FLAT is an elective course as per the new CSE curriculum, a student who does not take FLAT may not be exposed to a topic as fundamental as FSM. The proposed course replaces erstwhile CS-203 Discrete Structures that was not in complete agreement with the new curriculum design approach.

The proposed course is developed in consultation with the faculty members of the SCEE, especially Prof. Timothy Gonsalves, Dr. Sukumar Bhattacharya, Dr. Arti Kashyap, and Dr. Anil Sao, and Dr. N. S. Narayanaswamy from IIT Madras.

In the proposed contents, there are some topics that are marked "advanced" and/or "optional" ... these topics can be introduced depending on background and interests of the class and if the time permits. Rest of the topics form the essential core of this course that must be covered comprehensively, with lots of examples, and practice exercises, and weekly tutorials.

Objective: During the last two decades, there has been a major paradigm shift in processing, communication, and storage of information from predominantly analog domain to digital domain for the reasons such as ease of implementation, better efficiency, greater robustness against noise, and enhanced performance and security. This shift has not only resulted in generation of ever increasing volumes of digital data but also an acute need to efficiently process, store, and communicate it. To address this need the focus of CS curriculum must shift from introducing the traditional elementary discrete structures to represent and process the data to more abstract structures provided by abstract algebra and graph theory. This will enable the students to better understand and appreciate the current developments at the frontiers of CS, both in theory and application, and prepare them to contribute to further advancement of such frontiers.

On completion of this course, students should be able to demonstrate their understanding of and apply methods of discrete mathematics in CS to subsequent courses in algorithm design and analysis, automata theory and computability, information systems, computer networks. In particular, students should be able to

- use logical notation to define fundamental mathematical concepts such as sets, relations, functions and various algebraic structures, reason mathematically using such structures, and evaluate arguments that use such structures.

- model and analyze a computation or communication process and construct elementary proofs based on such structures

Syllabus:

1. Fundamental structures:

   Functions - surjections, injections, inverses, composition. (2 contact hours)

   Relations - reflexivity, symmetry, transitivity, equivalence relations. (2 contact hours)

   Sets - Venn diagrams, complements, Cartesian products, power sets, finite and infinite sets, introduction to lattices. (4 contact hours)

   Abstract orders: quasi-order, partial order, well-order, (Advanced, optional topics: Zorn's lemma, Koenig's theorem.) (2 contact hours)

2. Combinatorics: Counting arguments/techniques; pigeonhole principle; cardinality and countability, the inclusion-exclusion principle, recurrence relations, generating functions. (5 contact hours)

   Basics of graph theory: graph as a discrete structure, graph coloring and connectivity, traversal problems, and spanning trees. (5 contact hours)

   Advanced, optional topic: Probabilistic method in combinatorics.

3. Logic: Propositional and predicate logic: syntax, semantics, soundeness, completeness, unification, inferencing, resolution principle, proof system. (6 contact hours)

   Proof techniques (negation, contradiction, contraposition, mathematical induction) and the structure of formal proofs; efficiency of proof-systems. (4 contact hours)

4. State machines: Introduction, minimization, grammars, languages. (4 contact hours)

5. Algebra: Motivation for algebraic structures, the theory of some algebras such as monoids, groups (finite, cyclic, permutation, matrix), cosets, subgroups, Lagrange's theorem, discrete logarithm. (8 contact hours)

Optional topic:

6. Number Theory: Elementary number theory, fundamental theorem of arithmetic, gcd, unique factorization, Euler's function, modular arithmetic, Fermat's little theorem, Chinese remainder theorem, modular exponentiation, RSA public key encryption.

Suggested Reference Books:

1. E. Lehman, F. T. Leighton, and A. R. Meyer, *Mathematics for Computer Science*, 2013. Available online at: http://courses.csail.mit.edu/6.042/spring13/mcs.pdf

2. R. L. Graham, D. E. Knuth, and O. Patashnik, *Concrete Mathematics*, Pearson, 1994. Also, available online at: www.maths.ed.ac.uk/~aar/papers/knuthore.pdf

3. A. Aho and J. Ullman, *Foundations of Computer Science*, W. H. Freeman, 1992. Available online at: http://infolab.stanford.edu/~ullman/focs.html

4. I. N. Herstein, Topics in Algebra, 2/e, Wiley, 1975.

5. A. Tucker, *Applied Combinatorics*, 6/e, Wiley, 2012.

6. C. Liu and D. P. Mohapatra, *Elements of Discrete Mathematics*, 3/e, Tata-McGraw Hill, 2008.

7. T. Koshy, *Discrete Mathematics with Applications*, Academic Press, 2003.

8. J. Hein, *Discrete Structures, Logic, and Computability*, 3/e, Jones and Barlett, 2009.