1. CS305: Advanced Computing
2. Credits: 3, Contact Hours: 3
3. Richard Scherl
4. No textbook.
   1. Supplemental materials useful for the instructor
      * Hopcroft, John, Motwani, Rajeev, and Ullman, Jeffrey. *Introduction to Auotmata Theory, Languages and Computation ( 3rd edition).* Pearson, 2016.
      * Palach, Jan. *Parallel Programming with Python: Develop efficient parallel systems using the robust python environment*. Packt Publishing. 2014.
      * Prasad, Sushil and Gupta, Anshul, and Rosenberg, Arnold and Sussman, Alan and Weems, Charles (eds). *Topics in Parallel and Distributed Computing: Introducing Concurrency in Undergraduate Courses.*  Waltham, MA: McGraw-Hill. 2015
      * Rosen, Kenneth. *Discrete Mathematics and its Applications (Eighth Edition).* New York, NY: McGraw Hill. 2015.
      * Sipser, Michael. *Introduction to the Theory of Computation (3rd edition)*. Boston, MA: Cengage Learning, 2012
5. Specific course information
   1. Catalog description of the content of the course
      * Introduction to fundamental concepts of computer science theory and methods of parallel and distributed programming. The course covers automata theory (including finite-state machines and Turing machines), algorithm complexity (including the distinction between P and NP problems), BNF specification of programming languages, methods of parallel programming, methods of distributed programming, measuring the speedup obtained by parallelization, and methods of addressing NP completeness through approximation. The course will provide experience in the implementation of parallel and distributed programming.
   2. Prerequisite: CS205
   3. Required course
6. Specific goals for the course
   1. specific outcomes of instruction:
      * Students will be able to design a finite state machine to accept a specified language
      * Students will be able to design a regular expression to formally specify a language.
      * Students will be able to explain how finite state machines and regular expressions are related..
      * Students will be able to design a context free grammar to formally specify a language.
      * Students will be able to explain how pushdown automata and context free grammars are related.
      * Students will be able to explain the difference between a finite state machine and a pushdown automaton and describe why a pushdown automaton can accept more languages than a finite state machine.
      * Students will be able to create simple programs on a Turing machine.
      * Students will demonstrate an understanding of the differences between parallel, distributed, and concurrent programming.
      * Students will demonstrate ability to write parallel and distributed programs.
      * Students will demonstrate an understanding of the differences between decidable and undecidable problems and of the difference between tractable and intractable problems.
      * Student’s will be able to describe a Turing machine and explain its significance.
   2. ABET Criteria
      * Student Outcomes 1: Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions.
      * Student Outcomes: 6. Apply computer science theory and software development fundamentals to produce computing-based solutions.
   3. ABET Curriculum Requirements
      * Curriculum: 1 Substantial coverage of algorithms and complexity, computer science theory, concepts of programming languages, and software development.
      * Curriculum 3: Exposure to computer architecture and organization, information management, networking and communication, operating systems, and parallel and distributed computing.
7. Brief list of topics to be covered
   1. Finite state automata and regular languages
   2. Pushdown automata and context-free languages
   3. Turing machines and recursively enumerable languages and grammars
   4. Computability, decidability, undecidability
   5. Complexity, P vs NP
   6. Approximation as a strategy for NP complete problems
   7. Parallel programming in Python, speedup obtainable through parallelization
   8. Distributed programming and MapReduce