Course Title: Heuristics

Course Number: COREI-AD 24J

Course satisfies the Experimental Discovery in the Natural World subcategory of The Ideas and Methods of Science core category.

Location: NYU Abu Dhabi.

Credit hours: 4

Prerequisites: None

Co-requisites: None

Office hours: By appointment.

Name and Contact Information of Instructor:

Prof. Dennis Shasha
Department of Computer Science
Courant Institute of Mathematical Sciences
New York University
251 Mercer Street
New York, N.Y. 10012-1110
U.S.A.
Email: shasha@cs.nyu.edu
Web: http://cs.nyu.edu/cs/faculty/shasha/index.html

Brief Course Description:

Many problems in science, business, and politics require heuristics -- problem solving techniques that often work well but give imperfect guarantees. This course teaches heuristics as they apply to the design of scientific experiments, the resolution of economic or political negotiations, and in the construction of engineering devices in hostile environments. Students will work in small teams that will solve puzzles, conduct experiments, and build protocols for a competitive auction game. Students will use and learn computational tools, but the course has no programming prerequisite. The intent is to make you better able to face complex problems in any field you choose.

Learning Outcomes:

Design approaches to problems that are quantitative, procedural, and adaptive, even when the problems suffer from incomplete information or are only approximately described.

Understand heuristic techniques from computer science, biology, and human history.

Design social, physical, or computational experiments that apply heuristic techniques.

Teaching and Learning Methodologies:

Lectures, readings, experiments designed by students that will use equipment in the physics lab. Team-designed social, physical and computational experiments.

Methods and Dates of Assessments:
Class and presentation work 30% (evaluated based on interaction with other presenters and quality of student's own presentation)
week 1 project 20%
week 2 project 20%
week 3 project 30%
In-class quizzes

Information on out-of-class assignments: Daily readings and programming assignments.

Project 1: master one puzzle-based game, design a strategy that is likely to win and be prepared to take on challengers.

Project 2: design and demonstrate a physics-based or social experiment and write a description of the goals, techniques, and conclusions.

Project 3: compete in the design of algorithms for a simulated extra-terrestrial rover that encounters a hostile environment.

Course texts:

1. The Prism and the Pendulum
Robert Crease
Publisher: Random House Trade Paperbacks (October 12, 2004)
Language: English
ISBN-10: 0812970624
ISBN-13: 978-0812970623

2. Puzzles for Programmers and Pros
Dennis Shasha
Publisher: Wrox; 1 edition (May 7, 2007)
Language: English
ISBN-10: 0470121688
ISBN-13: 978-0470121689

3. Natural Computing
Dennis Shasha and Cathy Lazere
Paperback: 288 pages
Publisher: W. W. Norton & Company (May 17, 2010)
Language: English
ISBN-10: 0393336832
ISBN-13: 978-0393336832

4. *Statistics is Easy*, second edition
Dennis Shasha and Manda Wilson
Publisher: Morgan & Claypool [can be obtained online]
Language: English
ISBN-10: 1598297775

ISBN-13: 978-0393336832

5. *Guns, Germs, and Steel*
Jared Diamond
Paperback: 496 pages
Publisher: W. W. Norton & Company (April 1, 1999)
Language: English
ISBN-10: 0393317552
ISBN-13: 978-0393317558

Course topics and Contents on a Week-by-Week Basis:

Week 1: Thinking quantitatively -- estimation in the natural and human-constructed world (e.g. how many MacDonalds in Abu Dhabi). Heuristic inference of causes in Natural History. Thinking heuristically -- use of puzzles to approach difficult-to-solve and possibly adversarial problems systematically even under incomplete information. Start teaching python through graduated examples.

Primary readings: 1. Puzzles for Programmers and Pros 2. The website http://cims.nyu.edu/drecco 3. Guns, Germs, and Steel

End-of-week project: Be able to be the expert on some puzzle game and see if someone else can beat you at it.

Day 1: Motivation: when do we need heuristics? For unknowable information, incomplete information, situations where improvisation is inevitable. Estimation and adaptation examples. Separate students into groups.
Introduction to puzzle games on drecco site. Groups decide which puzzles to be expert at. Install python.

Day 2: Dynamic programming approach by hand and computationally. Sweet tooth dynamic programming puzzle Review and assign chapters from Crease and Natural Computing book. Python ?hello, world? with command line arguments.

Day 3: Chapters 2, 6, Epilogue from *Guns, Germs, and Steel* Notions of probability: Byzantine betters puzzle, Lucky roulette, Disease puzzle. Students present game they expect to master. Python arithmetic and conditionals.

Day 4: More probability: bait and switch, understanding dice, feedback dividends. Sudoku and the strategy of backtracking. Python loops and the twenty questions game.

Day 5: First contest. Strategic thinking puzzles: social games, optimizing the worst case, check amounts. Python functions along with recursion. Mergesort.

Week 2: Approaches to the design of great experiments. How does one construct a model of some target question well enough so that carrying out the experiment is fairly simple (e.g. that electrons move through a conductor and not protons)? We look at problems in physics, e.g. measuring the speed of electrons in metal. In parallel, we will teach more elements of the Python programming language including elements of interprocess communication.

Primary readings: 1. The Prism and the Pendulum 2. Statistics is Easy

End-of-week project: Students will construct their own experiment and demonstrate in class.

Day 1: Basic notions of statistics using a resampling approach. Student presentation of Crease chapters one and two. Python maps. Chapters 1 and 2 of *Statistics is Easy*

Day 2: Student presentation of Newton’s decompostion of light. Student presentation of Foucault’s pendulum. Students decide on their own experiment and discuss with me. Python file input/output.

Day 3: Student presentation of Millikan's measurements of subatomic particles. Student presentation of Milgram experiments. Computational game-playing techniques. Python interprocess communication.

Day 4: Complete other student presentations chosen from Crease runner-up chapter. Python interactive tic-tac-toe.

Day 5: Student presentations of their own experiment with demonstration in class. Game-playing techniques. Python interactive wordsnake game given common list of words.

Week 3: Approaches to the design of engineered devices that must face unpredictable hazards. Biological notions of feedback and evolution can contribute to a new form of adaptive and robust design. The discussion will center on non-examples like the Bhopal chemical plant as well as examples like marketplaces like the date and nut center near campus.

Primary reading: Natural Computing

Project: Student teams will build and compete with one another in a race of a simulated rover in a hostile environment designed by another team.

Day 1: Student presentations of Rodney Brooks and Glenn Reeves. Python backtracking and Sudoku.

Day 2: Student presentations of Louis Qualls and Jake Loveless. Python genetic algorithms and knapsack.

Day 3: Student presentations of Nancy Leveson and D. E. Shaw. Python simulated annealing and knapsack.

Day 4: Competitive Auction Game

Crosslisting: relevant to social scientists, natural scientists, and computer scientits mostly

Pedagogical methods to keep student interest: Course will be very interactive with lots of in-class time devoted to student presentations and projects.

Cultural diversity is not explicitly addressed, but diversity in approaches are fundamental to heuristic approaches to engineering. Field trip to local research labs studying water or energy use is a possibility.

Relationship of this course to others: useful to students in the social, natural, computational or engineering sciences. Not specifically redundant to any other course.