Harvey Mudd College

Computer Science 81

Computability and Logic

Spring 2007

Instructors: Professor Bob Keller, x 18483, keller@cs.hmc.edu
Professor Elizabeth Sweedyk, x 78360, z@cs.hmc.edu

Grader/Tutor: Adrian Sampson

Catalog Description:

> An introduction to some of the mathematical foundations of computer science, particularly logic, automata, and computability theory. Develops skill in constructing and writing proofs, and demonstrates the applications of the aforementioned areas to problems of practical significance. Prerequisites: Math 55, Computer Science 60. 3 credit hours. (Both semesters.)

Further Description:

> First we'll look at logic, both propositional and predicate forms, with an emphasis on the distinction between validity (truth) and proofs. We'll gain some experience in doing proofs in a style known as "natural deduction", both for propositional and predicate logic. This will be helpful as a way of outlining proofs for the rest of your careers. We will also look at several more mechanical methods of assessing the validity of a formula.
>
> We will next review finite-state automata and their relationship to languages and regular expressions. We'll look at the applications and limitations of these machines, then move on to the more powerful pushdown automata. We will show how pushdown automata exactly characterize the larger family of languages known as context-free languages. These are the languages generated by grammars of the type you studied in CS 60. We'll look at the applications of these automata, and also show their limitations.
>
> Next we'll look at Turing machines, which are even more computationally powerful. We'll discover what kind of grammars are equivalent to Turing machines as well as some specialized Turing machine models. We will also discuss the lambda

calculus, an alternate basis for computability and the basis for modern functional programming languages.

As with other families of machines, Turing machines have their limitations. We will show how to prove that certain problems cannot be solved by a Turing machine.

We will close by examining the connections between logic and computability in more depth, and conclude with a look at Godel's famous Incompleteness Theorem.

Grading: Homework 50%,
Exams 40% (1 midterm, 1 final)
Oral participation 10%

Collaboration Policy: To be explained in class.

Exam Policy: Closed book with single cribsheet.

Late Policy: No late work accepted, unless indicated by the instructor.

Required text:

James L. Hein: *Discrete Structures, Logic, and Computability*, **2nd Edition**, Jones and Bartlett, 2002, ISBN 0-7637-1843-2.

Note that we will not "cover" all of the text in this class, but much of the material not covered will likely have been seen in other classes, particularly Math 55.

Supplementary text:
Dirk Van Dalen: *Logic, and Structure*, **4th Edition**, Springer Verlag, 2004, ISBN 3-540-30879-8.

CS 81 Outline (Draft, Sept. 2007)

Approximate Day	Topics	Reading
(please review on your own in Hein)	Elementary Notions and Notations: Proof Primer; Sets;	1.1-1.2 1.3.1-1.3.2
1	Propositional calculus semantics Refutation tree method (not in text)	6.1; 6.2.1-6.2.3
2	Formal Reasoning Systems (Natural Deduction)	6.3, 6.4
3	Predicate Logic Semantics	7.1, 7.2
4	Formal Proofs in Predicate Calculus	7.3
5	Refutation tree method (not in text)	
6	Reasoning with Equality **Application**: Program Correctness	8.1 8.2
7	Computational Logic: Resolution (propositional)	8.3, 9.1
8	Computational Logic (predicate) **Application**: Automated Reasoning **Application:** Logic Programming	9.2
9	Soundness and completeness	supplement
10	**Midterm Exam**	
11	Regular Languages and Finite Automata, part 1 **Application**: Pattern Matching	11.1 11.2
12	Regular Languages and Finite Automata, part 2 Pumping Lemma for Regular Languages Abstract states of a language Myhill-Nerode Theorem	11.3 11.4
13	Context-Free Languages Pushdown Acceptors **Application**: Language Parsing	12.1 12.2
14	Parsing Techniques, part 1 Normal Forms	12.3
15	Parsing Techniques, part 2 CYK Algorithm	12.3 supplement
16	Pumping Lemma and Other Properties	12.4
17	Turing Machines Church-Turing Thesis	13.1 13.2
18	Lambda calculus	supplement
19	Computability and Semi-Computability (Recursive Enumerability) The Halting Problem	14.1

20	Reductions **Application**: "Every-day" Unsolvable Problems Rice's Theorem	supplement
21	Language Hierarchy Context Sensitive Languages	14.2
22	Undecidability of predicate logic	supplement
23	Completeness and incompleteness of predicate logic	supplement
24	**Final Exam**	