CS 550: Semantical Foundations of Programming Languages

Spring, 2009

Instructor: Deepak Kapur, kapur@cs.unm.edu

Course Contents

This course is a sequel to CS557, with a focus on semantics. CS557 thus serves as a prerequisite. The course will cover axiomatic and denotational approaches to semantics of programming languages, whereas structural operational semantics and lambda calculus is assumed to have been covered in CS557. The second part will cover logic programming and constraint logical programming. This will be followed by an overview of the abstract interpretation framework and its application to program analysis. Type reconstruction and type inference is assumed to have been covered in CS557. Besides lectures, the course work will include an extensive set of home work problems, a midterm, a term paper/project and a final exam.

Here is a list of topics and lectures; some of the material may end up taking more than 1 lecture.

1. Course overview, expectations, background information of participants and grading policy.
2. Lecture 2: Syntax vs Semantics. What is semantics?
3. Axiomatic Semantics: A bit about propositional connectives and quantifiers. Characterizing Program States using Formulas/Properties.
4. Programs as Predicate Transforms.
5. Hoare Triples. Axiomatic semantics of a toy language.
6. Backward vs Forward Semantics: The case of assignment statement.
7. Importance/Significance of Loop Invariants. Illustrations.
8. Generating Loop Invariants Automatically/Semi-Automatically.
9. Overview of Denotational Semantics.
10. Fix-Point Theory; examples of fixed points; Domain Theory.

11. Fixed Points of functionals and recursive definitions (recursion and higher-order functions presumed covered in CS557).
12. Information Theoretic Orderings; Least Fixed Point. Proving whether a given function is a fixed point (least) of a functional.
13. Computing Fixed Points.
14. Denotational Semantics of a Toy Language using recursive functions.
15. Mid-term Exam.
16. Relationship between Denotational Semantics and Axiomatic Semantics.
17. Logic Programming: Horn Clauses, Relational Programming; Reversible Computation.
18. Rules of Evaluation; left to right; top to bottom.
19. Review of Unification (presumed covered in CS557). Unification as Constraint Solving.
20. Constraint Logic Programming.
21. Semantics of Logic Programming.
22. Abstract Interpretation Framework: Galois Connection. Widening.
23. Examples of Program analysis using Abstract Interpretation Framework.
24. Student presentation on their projects/papers.

Recommended Books:

H.R. Nielson & F. Nielson, *Semantics with Applications: An Appetizer.* http://www.springerlink.com/c

A. Bradley & Z. Manna, *Calculus of Computation.* http://www.springerlink.com/content/wv0127/

J. Loeckx & K. Sieber, *The foundations of program verification.* Wiley-Teubner, 1987.

Reade, *Elements of Functional Programming.*

R. Bird, *Introduction to functional programming using* HASKELL. Prentice Hall, 1998.

S. J. Thompson, HASKELL*: The craft of functional programming.* Addison-Wesley, 1999.

Abelson and Sussman, *Structure and Interpretation of Computer Programs.*

Schmidt, *Denotational Semantics Methodology.*

Horowitz (eds.), *Programming Languages: A Grand Tour.*

Sethi, *Programming Languages: Concepts and Constructs.*

Kamin, *Programming Languages: An Interpreter-based Approach.*

First state whether you did CS591 last semester offered by Prof. Stefanovic, and how well you thought you understood the material in the course.

For each of the topics below, discuss to the best of your ability, how well you understand it. In case, you feel that your understanding is inadequate, discuss subtopics you feel uncomfortable with and you desire a possible review.

- Higher-order and recursive functional programming.

- Lambda Calculus

- Typed Lambda Calculus

- Type checking, inference and reconstruction.

- Type Polymorphism.

- Unification.

- any other topic discussed in CS591 not listed above.