Teaching Discrete Structures with SML

- Dr. Christelle Scharff, Pace University, NY, USA
- Dr. Andrew Wildenberg, SUNY Stony Brook, NY, USA
Outline

- SUNY Stony Brook Curriculum presentation
- Discrete Structures I
- SML in Discrete Structures I
- Students Feedbacks
- WeBWorK for delivering Online homeworks for Discrete Structures and SML
- Future directions
SUNY SB CS curriculum

- ACM standards for CS education and Discrete Structures
- Sequence of introductory courses of CS at SUNY Stony Brook

Discrete Structures I (around 1200 students each year until Spring 2002)

↓

Programming with JAVA

↓

Discrete Structures II

⇑

Data Structures with JAVA

- Curriculum created by Dr. Peter Henderson.
- SML is introduced in Discrete Structures I.
Discrete Structures I

- Propositional Logic
- Number Theory – illustrated through the use of SML
- Set Theory – illustrated through the use of SML
- Functions – illustrated through the use of SML
- Recursion – illustrated through the use of SML
- Induction
- Correctness of functions – illustrated through the use of SML and induction
- Automata
- Trees
- Graphs
Use of SML

- Students have no programming experience (60%).

- Discrete Structures I is NOT a programming class. SML is introduced as a tool.

- Why SML?
  - SML uses mathematical notations.
  - Syntax is not verbose.
  - Semantics is easy to understand.
  - Interactive environment.
  - Quick start.

- Minimal subset of SML:
  - types and inference types
  - list, tuples
  - function definitions
  - pattern matching
  - high-order functions
  - polymorphism
  - mutual recursion
Pedagogical goals

- Encourage mathematical thinking
- Convince students that the mathematics in Discrete Structures is relevant to their careers by teaching mathematics and computing together.
- Improve the understanding of recursion
- Introduction to problem solving
- Introduction to prototyping
- Introduction to software quality (modularity and reliability)
- Introduction to testing
- Introduction to formal proofs
User feedback and analysis

• Lots of students said they did not like SML.

• 54% admitted to bringing a negative preconception of SML into the class.

• The top 2 reasons to explain their difficulties in SML were:
  – problem solving (40%)
  – recursion (75%)
  – indicating the difficulties may not actually be related to SML.
SML and online homework

- WeBWorK
  - University of Rochester
  - Web-based system for delivering and evaluating homework problems
  - Features: encourage experimentation, user authentication, instant feedback, instant grading, individualized problem sets, evaluation of free-form symbolic answers, distribution of solutions...

- WeBWorK, functions and SML
  - Functions (Domain, Co-domain, composition...)
  - Problems on fundamentals and terminology of SML
  - Evaluation of recursive functions
  - Typing

- WeBWorK is very popular among students. (22% of students preferred paper-based homeworks).
Our records show problem 2 of set 2 has not been attempted.

(5 pts) Problem 1: Determine the results of the evaluation of the following SML functions.

1. `fun f(L) = if L=[] then [] else 11*hd(L):f(tl(L))`. Compute `f([1,2,3,4])`.

2. `fun f(L) = if L=[] then [] else 11*hd(L):f(tl(L))`. Compute `f([1])`.

Problem 2: Determine the SML types of the following functions and expressions.

1. `g` is a function of type `real->int*int`. What is the type of `f? fun f(x: nil) = (#2(g(x))):nil | f(L) = (#1(g(hd(L)))):f(tl(L))`?

2. `f` is a function of type `real->real`. What is the type of `map f`?

Problem 3: Determine what the following function does. c. `fun f(L) = if L=[] then 0 else hd(L)+f(tl(L))`.

A. `f` returns the length of the list `L`.
B. `f` returns the sum of 2 lists.
C. `f` returns the sum of the elements of the list `L`.
D. `f` returns the smallest element in the list `L`.
E. NDF
F. `f` returns all the elements of `L`.

Note: You can earn partial credit on this problem.
Conclusion and future directions

- Use of SML in discrete mathematics courses
- Use of SML through the curriculum
  - Languages
  - Software Engineering
- Environment for testing SML functions and returning appropriate error messages