

UMM Computer Science Assessment report

22 October 2009

Assessment in CSci 1000-level courses

The Computer Science discipline offers six 1000-level courses. Five of these all satisfy the Mathematics and Symbolic Reasoning (M/SR) general education requirement:

- CSci 1001 (Introduction to the computing world)
- CSci 1101 (Dynamic web programming)
- CSci 1201 (Introduction to digital media computation)
- CSci 1301 (Problem solving and algorithm development)
- CSci 1302 (Foundations of computer science)

The remaining course (IS 1091: Ethical and Social Implications of Technology) satisfies the Ethics and Civic Responsibility (ECR) requirement.

The catalog description of the goal of the M/SR category is "To strengthen students' ability to formulate abstractions, construct proofs, and utilize symbols in formal systems". Each of CSci 1001, 1101, 1201, and 1301 satisfy this requirement in similar ways, and the goals are assessed in similar ways as well. (There are important differences between these courses, summarized at the end of this document, but these differences aren't important to how they satisfy the M/SR requirement.)

Each of these courses introduces students to two key notions relevant to this requirement:

- The idea of a formal syntax
- The development and expression of algorithms (solutions to problems) in a formal syntax suitable for interpretation or execution by a computer

These tasks require the students to formulate abstractions as they apply different problem solving approaches. Each of these courses requires students to develop and express algorithms (as programs). These have to be particularly precise since they are being interpreted by a computer, and successful execution requires that a solution is both syntactically and semantically correct. Their understanding of these concepts is then assessed by evaluating their formalized solutions (programs). These are evaluated on a number of properties, including their syntactic and semantic correctness, clarity, and organization.

CSci 1302 also satisfies the M/SR requirement, although it does so more through proof than programming. Key relevant topics in CSci 1302 include:

- Translating English statements and requirements into formal statements in symbolic logic, and vice versa
- Writing formal proofs in symbolic logic
- Understanding tools for formally specifying and proving simple properties about computer programs

Assessment here is similar in that the students are required to develop solutions to a variety of problems, but this time in the formal language of symbolic logic rather than a programming language. Proofs and related exercises are again assessed throughout the course on properties such as syntactic and semantic correctness, clarity, and organization.

Lastly, the CSci discipline regularly teaches IS 1091 (Ethical and Social Implications of Technology), which satisfies the ECR general education requirement. The catalog statement of the goals for this requirement is "To broaden and develop students' capacity to question and reflect upon their own and society's values and critical responsibilities, and to understand forces, such as technology, that cause them to modify these views and often mandate creation of new ways to resolve legal, social, and scientific issues".

In this course students explore a series of readings and case studies that explore the question of what actually constitutes a technology, what impacts those technologies have, and how our responses to technologies both reflect and impact our collective and individual values. The students' growth is typically measured through ongoing assessment of their ability to analyze and effectively communicate their understanding of the issues; this can include in-class discussion, on-line posting or discussion, writing assignments, and in-class presentations. Important evaluation criteria here include clarity, relevance, and an ability to understand and describe (potentially subtle) interactions between technologies, and among technologies and societies.

Assessment of recursion

The CSci discipline also has a particular interest in assessing coverage of a specific problem solving technique (recursion) in the introductory courses that provide a gateway to our major (primarily CSci 1201 and CSci 1301, but also CSci 1302 to some degree). Assessment in the mid-90's made it clear that the then traditional introductory courses weren't providing our students with enough exposure and experience with recursion, leading to struggles in later courses. This observation was central to a major revamping of our introductory curriculum in 1995, including a shift in the rough analogue of our current CSci 1301 to a different teaching approach that (among other things) placed a much stronger emphasis on recursion. This proved very successful, and students in this system have been much more comfortable with recursion and related concepts, and more able and likely to use them when appropriate.

In recent years we've been exploring alternative entry points to our curriculum such as CSci 1201. One of our assessment goals has been to track the impact of these changes on students' understanding of recursion and their ability to apply it in subsequent work. Recursion is a less natural fit in CSci 1201 than in CSci 1301, for example, and we have made a special point of including additional material on recursion and assessing students' progress in that area. (See the "Examples" section for some specific data.) To date we've been focusing on assessment in our introductory courses. We also, however, plan to follow that up with assessment in relevant higher-level courses (e.g., CSci 3501 and CSci 3601) to verify that all our introductory options are providing adequate preparation in this area.

Impact of assessment on our 1000-level courses

Assessment of student experience has played a key role in the creation and structure of several of these courses.

CSci 1101 was created in response to broad interest in increased opportunities to learn advanced web development techniques. Steady interest in this course since its introduction suggests that we are serving that need in an important way.

One of the reasons for the creation of CSci 1201 was to provide an alternative entry point to the computer science major that focused on media computation (e.g., the manipulation of images and sounds). This was based on evidence from both here at UMM and nationwide that interest in traditional introductory computing courses was waning, and that many students responded better to introductions that were less abstract and provided the ability to apply the concepts in domains of interest (such as media computation).

Last year we added CSci 1001 as a pre-requisite for CSci 1101 based on assessment of student learning in CSci 1101. The data indicated that we were probably trying to cover too much in CSci 1101 (a 2 credit course), and that students with little prior background in web development were struggling to keep up. By making CSci 1001 a pre-requisite, students should come to CSci 1101 with experience in several key areas (especially HTML and CSS), allowing CSci 1101 to focus more cleanly on the web development concepts and their associated problem solving techniques and tools. The course has only been taught once since the introduction of this pre-requisite, so it's too early to say anything definitive about the impact, but it does look like we're seeing an improvement in success rates.

Examples of specific assessment activities and outcomes

Assessing understanding of formal syntax

One of the key goals for the M/SR category is to “formulate abstractions ... and utilize symbols in formal systems”. We cover this in most of our courses by having the students learn and use a formal syntax, either as a programming language or as the formal syntax for a proof system. In the Fall 2008 offering of CSci 1301, for example, the first quiz (“Scheme syntax quiz”) was specifically designed to assess students’ grasp of the formal syntax of the programming language they were learning. Of the 28 students that took that quiz, only five students scored less than 80%. The second quiz (“Constructing and extracting values from lists”) addressed some more complex syntactic issues, and there 17 students scored less than 80%, with 9 scoring less than 70%. The scores on this second quiz were worrying, as over half the class wasn’t demonstrating a level of understanding needed to succeed in subsequent material in the course. As a result additional class time was spent on that material, and related problems were included on subsequent assignments and exams. The following assignment (“Lists and conditionals”), for example, had problems that covered very similar material, and on that assignment only 7 students scored less than 80%, and only 4 less than 70%. The mid-term then had two questions specifically assessing their understanding of our formal syntax. The average scores on those questions were 90% and 92% respectively. The few students who continued to struggle with those concepts also struggled with other course material; difficulties with those questions correlated very strongly with

poor scores on the exam (almost everyone who got less than a 70% on the exam got a poor score on at least one of these two questions), and also correlated significantly with final grades in the course.

Assessing understanding of recursion

As discussed above, understanding of recursion is a significant priority for the Computer Science discipline, and there has been a particular effort to assess students' understanding of this concept in CSci 1201. In the Fall 2008 offering, for example, a specific section of the final was designed to assess their understanding of recursion and ability to apply it as a problem solving tool. The average score on that section was 73%, with only 2 of 20 students scoring less than 62%.

Differences between CSci 1001, 1101, 1201, and 1301

While each of these courses satisfies the M/SR requirement in similar ways (through the discovery of solutions and their expression as algorithms in formal languages), there are crucial differences in the details, mostly in the programming languages and tools used, and the problem domains emphasized.

The point of having this set of courses is to provide students with a variety of options, and to make computing more appealing to a broader audience at UMM. Our traditional introductory majors course (CSci 1301), for example, has worked very well for us and appealed to students that were more comfortable with abstraction. There were, however, some students who found it too abstract and wanted something with clearer applications to tasks they were interested in. This led directly to the introduction of CSci 1101 and CSci 1201, both of which provide introductions that focus on specific application domains that were known to be of interest to students (dynamic web programming and digital media computation, respectively). Enrollments suggest that these courses are working well as a group. This semester, for example, we're teaching both CSci 1001 and CSci 1301, and both had waiting lists at the beginning of the semester.