MSU Departmental Assessment Report
Spring 2011

Department: Computer Science

Department Head: John Paxton

Assessment Coordinator: John Paxton

Date: July 5, 2011

Degrees/Majors/Options Offered by Department

- B.S.
  - Professional Option
  - Interdisciplinary Option
- M.S.
- Ph.D.

Computer Science
Assessment Report – What happened in AY 2011

By August, 2011, we will have collected the following information:

1. Minutes from our weekly departmental meetings
2. Minutes from our departmental annual retreat in August 2010.
3. Minutes from our industry advisory board meeting in February 2011.
4. Course evaluation summaries for Fall Semester 2010 and Spring Semester 2011 courses.
5. The results of the Major Field Test, taken by graduating seniors in Fall Semester 2010 and Spring Semester 2011.
6. A graduating senior survey summary, constructed by Carolyn Plumb, conducted in April 2011.
7. A custom designed, graduating senior exam that graduating seniors took at the end of Fall 2010 and Spring 2011. The exam is designed to measure how well our graduating seniors are meeting our expected program outcomes.
8. Portfolios from students taking the professional option capstone (CSCI 468, Compilers) and interdisciplinary option capstone (CSCI 483, Interdisciplinary Project). These portfolios will be graded in the summer of 2011 to see how well students are meeting our expected program outcomes.

This information will be used at our August 2011 retreat to generate ideas for curricular improvements during AY 2012.

The rest of this report summarizes changes that the CS faculty made to our undergraduate curriculum in AY 2011 as a result of information that we collected during AY 2010.

Significant course enhancements in AY 2011:

1. In Spring 2011, our students were able to take a CSCI 491 Special Topics course on Cryptography and Cryptanalysis. The course was offered remotely through the Burns Telecommunications Center by a UM instructor, Mike Rosulek. (Source of change: #1 and #2 below.)
2. CSCI 441 and CSCI 541, Computer Graphics, were co-convened and offered during Spring 2011. (Source of change: #1 below.)
3. A new course, CSCI 447 Machine Learning: Soft Computing will be offered for the first time in Fall 2011. (Source of change: #1 below.)

Source of changes

The information that led to (and corroborates) the above changes includes

1. More state-of-the-art practice is needed in our curriculum. Sources: *Graduating Senior Survey Summaries* (Spring 2005, Fall 2005, Fall 2007, Spring 2009), *Alumni Survey Summary* (Spring 2009), *Course Evaluation Summaries* (Fall 2007).
2. A course on security is needed. This **qualitative** feedback has led to a security course, CS 455, being developed. Sources: *Graduating Senior Survey Summaries* (Spring 2005, Spring 2008, Spring 2009), *Alumni Survey Summary* (Spring 2009), *Employer Survey Summary* (Spring 2009), *Town Meeting Summary* (Spring 2009).

Assessment-based course changes in AY 2011

The following weaknesses were identified at our retreat in August 2010:

- On the graduating senior exam, question 1 identified a **quantitative** performance weakness in the area of program outcome a - an ability to apply knowledge of computing and mathematics appropriate to the discipline. Source: *Graduation Senior Exam Summary* (Spring 2010).

  Resulting change: Items 4 and 5 in the Appendix

- On the graduating senior exam, question 2 identified a **quantitative** performance weakness in the area of program outcome b - an ability to analyze a problem, and identify and define the computing requirements appropriate to its solution. Source: *Graduation Senior Exam Summary* (Spring 2010).

  Resulting change: Items 4 and 5 in the Appendix

- The Spring 2010 portfolio was completed by students in the interdisciplinary option capstone course (CSCI 483), but not in the professional option capstone course (CSCI 468). As a consequence, there was limited data (2 portfolios) to evaluate.

  Resulting change: In AY 2011, the compilers course will require students to submit their final project in the context of the portfolio requirements.

- Portfolio question 5 identified a **quantitative** performance weakness in the area of program outcome i – an ability to use current techniques, skills, and tools necessary for computing practices. Source: *Capstone Portfolio* (Spring 2010).

  Resulting change: Items 6 and 7 in the Appendix

- Portfolio question 6 identified a **quantitative** performance weakness in the area of program outcome j – an ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices. Source: *Capstone Portfolio* (Spring 2010).

  Resulting change: Throughout the curriculum, instructors will discuss design decisions when they arise so that students can better identify them in the context of their capstone projects.

  Design decisions occur in contexts too numerous to collect. Performance on the relevant capstone portfolio question will continue to be monitored for improvement.

- On the major field test, no **quantitative** performance weaknesses were identified. Source: *Major Field Test Summary* (Spring 2010).

- CSCI 320, Numerical Methods needs to have more MATLAB assignments. Source: *Student and instructor feedback* (Spring 2010).

  Resulting change: Item 1 in the Appendix

- CSCI 338, Theory of Computation, needs to show the relevance of the material whenever possible. Source: *Student and instructor feedback* (Spring 2010).

  Resulting change: Item 2 in the Appendix

- CSCI 361, Computer Architecture needs a larger assembly project. Source: *Student and instructor feedback* (Fall 2009).

  Resulting change: Item 3 in the Appendix

Appendix

Item 1 → CSCI 320, Numerical Methods: there need to be more substantial MATLAB assignments. (Year Back)

To utilize Matlab more in the class, the following changes had been implemented:

1. A hand-on lab session on Matlab was given in EPS 109 at the beginning of the semester. In this lab session, students implemented the Matlab commands on the computer to appreciate the power of Matlab in solving diverse numerical computation problems.
2. Matlab was installed in the computer in class (Gaines Gall). Then the tutorial on Matlab, Chapter 2 and Chapter 3, was given using Matlab at the beginning of the semester.
3. A number of homework problems, which were required to solve using Matlab, were given.
4. Matlab solutions were demonstrated in class throughout the semester.

Item 2 → CSCI 338, Computer Science Theory: show the relevance of the material whenever possible. (Binhai)

Several examples are covered for practical relevance. For example, in covering undecidability the generation of random numbers was covered; for NP-completeness, several examples in computational biology and computational geometry (like sequencing a genome from short reads and packing polygons in a box) were covered.

Item 3 → CSCI 361, Computer Architecture: there should be a larger assembly project. (Year Back)

The Fall 2010 project:

In this project each team of two students built a simple programmable digital computer to help them understand the inner workings of a computer at the hardware level. The project kit contains necessary components to build a computer. There is a manual, too. The whole project consists of seven steps.

1. Timing Signal Generation
2. Program Counter and Arithmetic Unit
3. MAR, Data Register A and Data Register B
4. Random Access Memory and Instruction Decoder

5. Control Signal Generator
6. *Interrupt Processing*
7. *Operating a Combinatorial Circuit ALU*

Students were required to implement up to step 5; steps 6 and 7 were optional. The TA supervised the progress of the project throughout the semester.

Item 4 → Time complexity and statement counts will have graded assignments in CSCI 132, CSCI 232 and CSCI 468 during the upcoming year. At the end of the year, instructors of these courses will be solicited to provide the assignments.

132 (Hunter):
A full lecture on time complexity for an O(n^2) sort was given, followed by a quiz on the topic. Another lecture was used to cover sample test questions on time complexity. Time complexity was also used in a lecture when trees were introduced. One lab emphasized time complexity and then the lab was covered in lecture. The final exam contained a time complexity question. A document that Hunter prepared with more extensive information is available upon request.

132 (Brendan):

The topic was covered in lecture. Two labs were on time complexity. The final exam contained a question on time complexity. These two labs and the final are available upon request.

232 (Binhai):

In CSCI 232, a whole lab assignment was on time complexity (including both recurrence relation solving and coding under some time complexity constraints). The assignment was lab 1 and is available on request.

For the solving of recurrence relations, we spent extra time covering the topic (right before the final exam). As a matter of fact, close to 75% of the students got it right in the final, which has never occurred before (usually it is at most 30%). The final exam question is

Q3: 5 points out of 30. Solve the following recurrence relation and prove your claim by induction:

T(1)=1
T(n)=4T(n/4) + 3n^2, for n >= 2.

468 (Rocky):

A pdf available on request shows a lecture in which time complexity was discussed. It's only O(n), but two other interactive lectures ask students for an approach to, say, implement a symbol table with various actions, such as insert and lookup, and then analyze and justify the time complexities of the various algorithm choices. When it's all done, it is discussed that the real time complexity of every choice is essentially constant, as the symbol table itself is of fixed size. That leads to the fact that the real time complexity has to do with the number of identifiers in the program, so the choice of a fast algorithm for lookup is important, but may not be the expected algorithm, because the constants are not large in terms of the algorithms, whereas they might be in terms of the number of identifiers that must be inserted and looked up.

Item 5 → All instructors are aware that time complexity and statement counts are difficult for our students and will incorporate them when relevant. At the end of the year, instructors of all courses will be solicited to provide example assignments.

132 (Hunter, Brendan), 232 (Binhai), 468 (Rocky): see Item 4.

446 (John S.):

I went back and reviewed all of the assignments and exams for CSCI 446, and none of them include anything that explicitly assesses determining time or space complexity. Note, however, that we do discuss complexity issues in class.

One might be able to argue that the two programming assignments require students to make certain design decisions, but these are not explicitly addressed (other than through doing things like comparing simple minimax to alpha-beta or value iteration to Q-learning).

Item 6 → UML will continue to be emphasized in the software engineering courses.

SE 322 and SE 422 (Clem): UML is taught and used by an instructor with 17 years of industry experience and a Ph.D. in software engineering.

Item 7 → Students will be reminded to design with UML in the capstone courses.

CSCI 482/483: Students were required to submit UML designs with their capstone portfolio.

CSCI 468: Students were required to submit UML designs with their capstone portfolio.