

Mapping Grinnell's CS Curriculum to CS 2013: A Narrative

Notes by Henry M. Walker

In July 2013, Grinnell's computer science faculty engaged in a careful review of the computer science curriculum at Grinnell College.

- Meeting during three mornings, the faculty examined data from exit interviews from graduating seniors, surveys from alumni, and teaching observations from faculty.
- In pairs, the faculty mapped existing CS courses to CS 2013.
- The faculty built upon on-going curricular discussions in areas of project-based courses, programming languages, and systems.
- The entire faculty identified strengths and opportunities for new directions, hypothesized possible new or revised course offerings, and developed preliminary mappings for possible new revised offerings.

This narrative describes several highlights of this curricular review process, with special emphasis on both the mapping of existing Grinnell CS courses to CS 2013 and conclusions from the mapping process. The narrative has contains five main sections:

0. Context
1. Preliminary faculty discussions (before the CS 2013 mapping)
2. Notes on filling in the CS 2013 spreadsheet
3. Observations on the Grinnell CS curriculum — both current and future
4. Next steps

0. Context

Being a liberal arts college, Grinnell College limits majors to 32 credits in the major program (typically 8 four-credit courses — possibly plus a few supporting courses in other departments). Students can take additional courses (up to 48 credits in a department can count toward graduation), but major requirements cannot exceed 32 credits in the major department. With these constraints, Grinnell's CS major has the following outline:

- multi-paradigm, 3-course introductory sequence
 - § 1 course each: functional, imperative, and object-oriented problem solving
 - § 12 credits total
- 1 supporting mathematics course (4 credits)
 - § as a supporting course, this does not count toward the 32-credits in the major department (it does count toward graduation)
- 2 mid-level, core courses: algorithms, theory of computation: 8 credits
- 1 systems course (4 credits)
 - § architecture or operating systems
 - § only one required, but students advised to take both to satisfy CC2001 recommendations
- 1 languages course (4 credits)
 - § programming language concepts or compilers
- 1 project-based course (4 credits)
 - § software design or databases and Web application design

Several electives are offered, but cannot be required. Although the major can only require one course in an area (e.g., systems, languages, project), students sometimes take both options.

For the most part, students taking both systems courses (rather than the one that can be required) cover the full range of topics recommended by CS 2001. Electives and a substantial program of student-faculty research provide additional opportunities. Overall about 50% of recent graduates have had internship experiences and 83% have had a research experience.

1. Preliminary faculty discussions (before the CS 2013 mapping)

During a July 2013 CS Curriculum Workshop, the CS faculty reviewed feedback from students, communications from alumni, experiences from faculty, and other data concerning Grinnell's curriculum. All sources indicated Grinnell's multi-paradigm, 3-course introductory sequence works well — particularly within a liberal arts setting. Similarly, core course on Computer Organization/Architecture, Operating Systems/Parallel Algorithms, and Analysis of Algorithms are well received and cover core areas.

The major has emphasized theory of computation and programming languages beyond national curricular guidelines, but these topics fit well in a liberal arts setting, and students respond well to courses in these areas. (Students may not be wildly enthusiastic studying the theory of computation, but they appreciate the need to understand fundamental concepts.)

Project-based courses (software design and databases/Web design) are generally well reviewed by students and alumni. However, the 1-semester time frame of these courses seriously limits the ability to cover both foundational areas within software development and allow students to work collaborative on a significant project.

A networks course elective is offered in alternate years, and alumni identify the topic as being very important, but enrollment in the course often is low. Also, students, alumni, and CS faculty have interest in developing an offering related to computer security.

To facilitate further departmental discussions, the CS faculty decided to complete the CS 2013 spreadsheet for 18 current, revised, or hypothesized courses:

- 3-course Multi-paradigm, introductory sequence (CSC 151/161/207) (4-credits each)
- 1 supporting course: CSC 208 *Discrete Structures*
- 2 on-going, required, 4-credit courses
 - § CSC 301 *Analysis of Algorithms*
 - § CSC 341 *Theory of Computation*
- 2 systems courses (1 required, both strongly recommended)
 - § CSC 211 *Computer Organization and Architecture*
 - § CSC 213 *Operating Systems and Parallel Algorithms*
- 2 4-credit project-based courses (1 required)
 - § CSC 323 *Software Design*
 - § CSC 325 *Databases and Web Design*
- 2 4-credit languages courses (1 currently required) (may be replaced by 2-credit elective)
 - § CSC 301 *Programming Language Concepts*
 - § CSC 362 *Compilers*
- 2 new or revised 2-credit courses on networks and security
 - § *Computer Networks* (revised from 4 credits to 2 credits)
 - § *Computer Security* (new)
- 2 new 2-credit courses to support on-going, community-based, software projects
 - § *Tools and Principles of Software Development* (new)
 - § *Community-based Projects* (new)
- 2 4-credit electives
 - § CSC 261 *Artificial Intelligence*
 - § CSC 295 *Computer Vision*

Since data from these courses include current, revised, and potential courses — both required and elective, the overall spreadsheet identifies outcomes covered somewhere in the curriculum, but does not specify guaranteed outcomes for every student.

2. Notes on filling in the CS 2013 spreadsheet

Completing the CS 2013 spreadsheet is an intimidating exercise.

- CS faculty must complete 1052 cells for each course under consideration. For Grinnell's 18 courses, the faculty had to review 18,936 cells.
- Natural questions arise regarding how many Tier 1 and Tier 2 outcomes are met in each course and how many are met somewhere — in some course. Addressing these questions required insertion of formulae into another 1185 cells.

Altogether, Grinnell's CS faculty made decisions on entries to about 20,000 spreadsheet cells.

The current CS 2013 spreadsheet does not relate directly to earlier CS 2013 spreadsheets.

- An earlier CS 2013 spreadsheet (completed for the review of potential exemplar courses) focused upon topics and hours rather than learning outcomes.
- Terminology between topics and outcomes does not always match. For example, buffer overflow appears as a Tier 1 topic in IAS/Defensive Programming, but no learning outcomes for that Knowledge Unit mention buffer overflow.
- The current CS 2013 spreadsheet requires review of at least 1052 cells (for outcomes), in contrast to the earlier CS 2013 spreadsheet that required review of hours for 171 Knowledge Units. Thus, the current spreadsheet requires review of over 6 times the cells of the earlier version.

The CS 2013 spreadsheet contained an off-by-one computational error (in cell C1393), so manual counts of compliance with Tier 1 outcomes did not match the totals obtained by the formula (−13 in the formula should be −14). Finding this error was time consuming.

Several additional issues arose throughout this exercise:

- Some learning outcomes seemed repeated under multiple Knowledge Units, and it was not clear if different skills were intended.
- When topics are covered in several courses, recording is problematic. Faculty may enter “Familiarity” and/or “Usage” in some cells to track the development of ideas through the curriculum. A level of “Assessment” may be achieved in a later course, but achieving that level of mastery is not isolated in one course. However, any entry in a Tier 1 or Tier 2 cell changes the color of the cell, so entries that reflect the evolution of learning may not be reflected in the row color within the spreadsheet.
- Since faculty want to know coverage for the entire curriculum, not just for individual courses, it was natural to add a column computing how many courses cover each learning outcome. However, addition of such a column with a formula caused the spreadsheet to consider every learning outcome was covered. This led to the use of two spreadsheets — one giving colors (as distributed by CS 2013) and one giving counts of how many learning outcomes were achieved overall.
- Some stated CS 2013 areas, particularly in the area of Social Issues and Professional Practice (e.g., Social Context, Professional Ethics, Intellectual Property, Privacy, Civil Liberties), represent themes that pervade many Grinnell CS courses. Students see these issues often, but faculty cannot identify a specific course that achieves a designated outcome. As a result, the CS faculty are confident that students graduate with considerable background in these areas, but cannot record this outcome in a specific spreadsheet cell.
- The Curriculum Summary sheet displays only whether a collection of courses meet ALL outcomes for a Knowledge Unit. If one of many outcomes is missed for an outcome requiring just a few hours, then the section is not highlighted as being covered. This seems to potentially misrepresent coverage of some Knowledge Units and undermines the effectiveness of the Curriculum Summary.
- Most importantly, despite the efforts of the CS 2013 committee to describe learning outcomes, interpretation of the outcomes is subjective for individual courses.

Overall, the completion of the CS 2013 spreadsheet should be considered as an approximation of topic coverage. Elements of interpretation arise for many cells. Further, the scope of the spreadsheet suggests that some details will likely be missed.

Grinnell's CS faculty consider the current spreadsheets (one with color, one with counts) as mechanisms to help identify general patterns within the curriculum. However, specifics are subject to interpretation and should not be considered as necessarily precise.

3. Observations on the Grinnell CS curriculum — both current and future

The CS 2013 spreadsheet covers both existing and potential courses. Students are required to take several core courses, but some topics are covered in electives.

Overall, Grinnell's CS program covers all, or almost all, Tier 1 and Tier 2 learning outcomes in many areas identified by CS 2013:

- Algorithms (all Tier 1 and 2 outcomes)
- Architecture (all Tier 1 outcomes, many Tier 2 outcomes)
- Computational Science (all but 1 Tier 1 outcome, [no Tier 2 outcomes recommended])
- Discrete Structures (all but 1 Tier 1 and all but 1 Tier 2 outcomes)
- Human-Computer Interaction (all Tier 1 outcomes, all but 1 Tier 2 outcomes)
- Information Assurance and Security (all Tier 1, many Tier 2 outcomes covered by a new course under development)
- Intelligent Systems (Tier 1 and Tier 2 in elective AI course; limited coverage elsewhere)
- Networking and Communications (Tier 1 and 2 covered in current 4-credit elective and in a new 2-credit course under development)
- Programming Languages (extensive coverage of Tier 1, Tier 2, and beyond)
- Software Development Fundamentals (all Tier 1 and 2 outcomes)

Grinnell's CS program covers a moderate amount of learning outcomes in some areas:

- Graphics and Visual Computing (about half of the few Tier 1 and Tier 2 outcomes)
- Operating Systems (all but 2 Tier 1 outcomes, many Tier 2 outcomes)
- Parallel and Distributed (all but 2 Tier 1 outcomes, many Tier 2 outcomes)
- Software Engineering (all but 2 Tier 1 outcomes, many Tier 2 outcomes)
- System Fundamentals (all but 3 Tier 1 outcomes, many Tier 2 outcomes)

In a few areas, Grinnell's CS program is spotty:

- Information Management
- Social Issues and Professional Practice (recurrent theme among many courses, but not assigned to specific courses)

Comparing all learning outcomes to those covered by Grinnell courses:

- Grinnell's existing, revised, or developing courses cover 208 of the 247 Tier 1 outcomes identified by CS 2013 (84%)
- Grinnell's existing, revised, or developing courses cover 210 of the 283 Tier 2 outcomes identified by CS 2013 (74%)

Looking more deeply, a few of the CS 2013 learning outcomes seem peculiar. For example, the last outcome in AR/Machine-level representation of data is "Write simple programs at the assembly/machine level for string processing and manipulation [Usage]." Grinnell courses require students to work with strings as arrays of characters, understand coding schemes, implement string algorithms, etc. Other courses require students to write assembly/machine language programs. It is unclear, however, why string processing must be in assembly or machine language, so there is little incentive for Grinnell's courses to meet this outcome.

4. Next steps

The current CS 2013 spreadsheet and analysis provides insight on topics and outcomes in Grinnell's CS curriculum. Completion of individual cells are open to interpretation and revision, but the spreadsheet provides some insights regarding the overall curriculum.

Looking ahead, further analysis and development will likely take place in three areas:

- **Networks/Security:** Replacing the current 4-credit Networks course by a 2-credit networks course and a new 2-credit course on computer security would meet requests from students and alumni and also cover several areas that have emerged as vital within CS 2013. The position of these courses within the CS major will have to be considered.
- **Programming languages:** this area represents a strength of the current program, and the current CS major requires a course in this area. Replacing two 4-credit courses meeting a requirement with a 2-credit elective might allow interested students to pursue work in this area while opening possibilities in other areas.
- **Project-based Course:** On some other campuses, project-based activities are split into two components: discussion of software engineering principles and practices, and active involvement in a client-based team project. An analysis of outcomes for both the current courses and courses under consideration may provide insight as to what might be gained and what lost with this new approach to courses related to software development.
- **Once possible adjustments in the curriculum evolve,** requirements of the CS major will need review. As already noted, Grinnell College severely restricts the number of courses allowed for a major. Coverage of topics within all CS courses may help guide what can and should be required for the formal CS major. As with the current major requirements, the formal CS major may identify core areas, but additional courses may be recommended to meet the full CS 2013 guidelines.

Altogether, this exercise has reinforced beliefs of the CS faculty that the overall Grinnell CS curriculum is quite strong and covers many of the learning outcomes identified by CS 2013. A specific area of weakness involves computer security, and this review process helped the faculty to brainstorm about a new course on that subject that might coordinate with a revised course on computer networks. Over the years, Grinnell has been particularly strong in the area of programming languages, and discussions among the faculty suggest a possible cutting back in that area. The curriculum workshop and the CS 2013 review also helped the faculty rethink team projects within the curriculum.

Finally, constraints within Grinnell College severely limit the size of a major. Once new or revised courses are identified and the CS curriculum stabilizes, the spreadsheet is expected to help the faculty review which courses should be required and which identified as elective. It also seems possible that the formal CS major may list some courses and alternatives, while a separate statement will help advise students seeking careers in computing on course that cover the full range of recommended CS 2013 topics.