Materials Development in Support of Mathematical Thinking

Peter B. Henderson (Chair)
Butler University
phenders@butler.edu

Bill Marion
Valpariso University, USA
Bill.Marion@valpo.edu

Sister Jane Fritz
St. Joseph's College, NY, USA
fritz@sjcny.edu

Charles Riedesel
University of Nebraska-Lincoln, USA
riedesel@cse.unl.edu

John Hamer
University of Auckland, NZ
j-hamer@cs.auckland.ac.nz

Christelle Scharf
Pace University, USA
cscharff@pace.edu

Lew Hitchner
California Polytechnic State University, USA
hitchner@falcon.csc.calpoly.edu

1 Topic Overview/Motivation

There is evidence that rigor is fading from computer science curricula [7,12,16]. Possible reasons for this include: a broader perception of the discipline of computer science (e.g., word processing to heavy mathematical applications), pressure to increase enrolments at universities, and faculty who don't have the necessary resources or guidelines for teaching more rigorous computer science courses. An objective of this working group is to help reverse this trend by providing materials that educators can use to convey the importance of mathematics in their courses, and to make the connections between mathematics and computer science topics.

The ITiCSE 2001 working group "Striving for Mathematical Thinking" [8] accomplished several goals, including identifying a working definition of mathematical thinking[1], the specification of three areas of focus regarding mathematical thinking in computer science education (i.e., gathering supporting evidence, basic ideas underlying mathematical thinking, and ways to promote mathematical thinking), and recognition of the need for materials in support of the instruction of more mathematically oriented computer science courses, especially in the first two years [13]. Accordingly, an important next step is the collection, categorization and development, and dissemination of instructional materials.

It is important that these materials conform to certain useful guidelines. In particular, the emphasis must be on the applications of mathematical concepts in the context of computer science. In addition, the materials should illustrate connections between each entry and other entries. For example, the 'select' data base operation could be implemented by linear search.

Our working group identified an on-line repository as the best mechanism for organizing and disseminating these materials. Accordingly, we began looking at similar repositories and models for repositories, such as CITIDEL [2] and CSTC [4] and JERIC [11] for computer science educators, GEM [6] and NSDL [15], and Cut the Knot [5] and Math Forum [14] for mathematics educators. The proposed repository was divided into three primary areas, Discrete Mathematics, CS I and CS II, and other courses, including CS 0 and computer literacy. The anticipated users are computer science faculty, math faculty, and pre-college teachers. A similar repository for students would be a natural extension once this basic repository has proven useful and effective.

[1] "Applying mathematical techniques, concepts and processes, either explicitly or implicitly, in the solution of problems."

A preliminary metadata format (template) for a web page for each problem was developed and there were discussions of ways to populate and sustain this effort. Each member of the group completed example problems for the proposed template and this exercise was used as feedback to further refine the template. Issues relating to security, classification of material, copyright, peer review of material, appropriate metadata and associated tools for creating and searching the repository were discussed, but require further evaluation before final decisions are made. The final repository will be developed using an existing educational metadata system such as ones based upon the Dublin core [18], IEEE learning object metadata [10], or the UK's metadata for educational group [17].

To ascertain the usefulness and use patterns of similar on-line repositories, a survey of the audience at the ITiCSE 2002 working group presentation was conducted. The survey and its results are given in Section 8.

2 What is Mathematical Thinking?

In the same way that many introductory students believe "CS = programming", some CS educators equate mathematical thinking with formal methods. Our working definition of mathematical thinking "Applying mathematical techniques, concepts and processes, either explicitly or implicitly, in the solution of problems" is much more general. Using this definition every problem solving activity can be viewed as an application of mathematical thinking. Our goal is to raise the level of consciousness of the important role mathematics can play in the education of computer science and software engineering students. Our hypothesis is that an undergraduate program rich in mathematics, not just mathematics courses, but mathematics integrated throughout the curriculum, will produce software developers who are better problem solvers, who can think more generally (i.e., outside the box), who can learn new concepts quicker and who are lifetime learners. This hypothesis is not yet proven, but there is significant evidence that facility with mathematics is important [8]. Indeed, one eventual goal, and a potential future working group, is the development and undertaking of studies that support or refute this hypothesis.

To make this discussion more concrete, consider one concept: iteration invariants. In current curricula this concept is introduced once or twice, but rarely reinforced. Also, the connections with mathematics (e.g., logic and induction) are usually not made. Assume for the moment that foundational logic and discrete mathematics is introduced early in our curricula and iteration invariants are introduced as an application of mathematics in introductory courses and then reinforced in all subsequent computer science and software engineering courses. The concept and use of iteration invariants could be introduced intuitively and then progressively more formally. This is similar to the way that calculus serves as a foundation for teaching and understanding engineering principles. After four years would students with this more mathematical perspective have a greater propensity to write correct iterations and be able to debug faulty ones? We think so. We believe that, mathematics, when taught and used properly, provides a set of powerful intellectual tools and leads to "clarity of thought[2]".

3 The Audience

This working group has identified three primary target audiences for the proposed repository: mathematics teachers, CS I and II teachers, and upper level computer science teachers. Discrete mathematics teachers include high school and college mathematics faculty, and computer science faculty. Seeing how to modify their mathematics as well as computer science courses to aid the computer science bound students can be immensely helpful. Perspective computer science majors will have a head start if they have seen and appreciate the mathematical connections to the discipline of computer science.

College level discrete mathematics teachers frequently face frustrated students who are looking to them for reasons to study the material. Examples and exercises that relate the mathematics to topics in computer science will help to dispel this frustration. Other examples in the repository will make connections between logic and programming, will reinforce the view of algorithms as constructive proofs, and consider applications of mathematical concepts to computer science (e.g., relations for relational database systems, Boolean algebra for computer architecture, grammars for compilers, finite state machines for modeling and design, etc.).

[2] Personal correspondence with Richard Rasala.

CS I and II teachers, including pre-college computer science teachers, constitute the second audience. Most introductory computer science textbooks focus on programming and basic concepts, but fail to make the connections with mathematics at the stage where students are just forming their conceptions of what computer science is. We believe consistency and continuity in the presentation of a mathematically rigorous view of computer science can alleviate problems later in the students' study of computer science. This repository can fill the gap in the texts as students begin to anticipate the more advanced topics in computer science. The examples in the repository will include many examples of computer science applications from all the Curriculum 2001 discrete mathematics sub topics.

Teachers of advanced computer science courses will be in a position to make the connections complete. Building upon, reinforcing, and taking advantage of the efforts of the earlier material in the repository can be of help at this level. Students will be prepared for the relational algebra in Databases, the deadlock algorithms in Operating Systems, the routing algorithms in networks, the logic in Artificial Intelligence, etc. The repository will be organized to show the progressions in prerequisite knowledge and difficulty among the examples, so finding appropriate material for any level is made more easily.

4 Populating the Repository

Members of the "Math Thinking" group [1] will design a set of teaching resources and sample problems to initialize the repository. These provide the initial seed, which will encourage ongoing contributions to the repository from both the mathematics and computer science communities.

Several structures have been created to encourage people to submit materials. Editors have been selected and assigned to each of the large topic areas (roughly corresponding to the Discrete Structures topics [3] and the various target audiences.) They will be responsible for soliciting and collecting the materials from their areas, proofreading and peer reviewing them, and placing them in the repository. They will also maintain and update that area of the repository to assure that it remains current. To facilitate the submission of materials, a downloadable metadata form (template) will be provided that can be filled in and emailed or posted to the appropriate editor. This template will ensure consistency, minimize search time and make it easy to submit materials. Useful connections to other appropriate pages with pedagogy, solutions, additional references, etc. can be provided through links on each template.

As we envision it, this repository will be more than just a collection of useful examples. It will be the shared expertise of the teaching community, designed around pedagogical techniques and it will emphasize the relationships between mathematics and computer science. Another value of this structured repository is that it may encourage us all to reflect more on the pedagogy and learning objectives involved in our examples, exams, assignments and course materials. As a result of this reflection, we anticipate that teachers will organize and evaluate their own materials, and share those that have been effective in their courses via submission to the repository.

5 Framework of the Repository

5.1 Purpose

Each entry in the repository will be an example and solution that demonstrates effective use of mathematics within the context of computing science and software engineering. Examples may be used directly as lecture material, homework or laboratory assignments, group activities, projects, or exam problems. Or they may be used as seeds for new examples. Eventually, instructor organized collections of entries may also be used by students as a study guide.

5.2 Structure

Repository entries will be structured with single level topic headings chosen to facilitate searching, collecting, and comparing them. The presentation format for each entry will include a small number of required and optional headings followed by a limited amount of material that can be displayed on one browser page. Additional

information will be accessible through web links that may be one or more pages in length. In most cases, the main page will contain text only. Linked pages may be text, image, or other documents. Non-textual pages will contain meta tags to support searching. The main page will also include a list of keywords and example classification to facilitate searching.

5.3 Required Headers

1. **Title + status:** A title should be concrete and specific rather than abstract and general. 'status' indicates the review status of the material (e.g, not reviewed, reviewed, class tested).
2. **Keywords.**
3. **Type:** The type is the category of example or problem. These include:
    - **a.** Lecture example
    - **b.** Homework problem or programming assignment
    - **c.** Laboratory exercise
    - **d.** Exam or quiz question
    - **e.** Individual or group project
    - **f.** Supplementary material
    - **g.** Other
4. **Description:** The description is a narrative explanation of the problem/example that ideally should be limited to one browser page, and may include links to other supplementary description information such as figures, etc.
5. **Solution:** The main web page will only contain a link to the actual solution. The solution will be complete and may include alternatives that illustrate different levels of sophistication based upon various criteria. Solutions will be encouraged to include graphical figures to support them whenever possible.
6. **Learning Outcomes:** The learning objectives and expected outcomes will be listed in a one-sentence description or a list of keywords.
7. **Prerequisite Knowledge:** The required knowledge will be listed in a keyword style list of topic areas.
8. **Contributor:** The contact information for the person or persons who contributed this entry will be listed.
9. **Source:** The original source of this entry will be listed. For example, some entries may be from published textbooks, workbooks, or papers or they may be from other repositories or web pages.

5.4 Optional Headings

1. **Level:** The level is the age or course category for the audience of students. This will include secondary school, first year university, or an advanced year of university.
2. **Pedagogical Notes:** Guidelines for the instructor on how to use the example material. These guidelines will include one or more of the following: Bloom's Taxonomy classification, description of student misconceptions about the problem solution, descriptions of a mental model required by students to solve the problem, hints the instructor can supply to the students, hints to the instructor for ways to scaffold the problem, and notes about practical experience with the problem.
3. **Assessment Techniques:** The main web page may contain a link to notes for the instructor about assessment techniques. These will include ways to examine students on their comprehension of the problem and its solution. Assessment techniques may include exam or quiz questions as well as exercises that could be performed on a computer.
4. **Connections:** Pointers to useful related material, links to other examples in the repository and supporting software. A connection is intended lead an instructor to a related problem that uses the same mathematical technique or to a different application domain that builds on the concepts learned in this problem.

5.5 Example

A sample entry "IF-THEN-ELSE logic" is shown below. Other preliminary examples, using a basic web based template, can be found at http://blue.butler.edu/~phenders/iticse2002/. These are simply examples, and do not reflect the final version of the proposed repository. In the example below, Keywords, Type, Prerequisite Knowledge, Solution, Pedagogy, Learning Outcomes, Source and Connections are hyperlinks to the corresponding sections below.

IF-THEN-ELSE logic (Review Pending)
Keywords Type Prerequisite Knowledge Solution Pedagogy Learning Outcomes Source Connections
Contributed by: Peter B. Henderson (phenders@butler.edu)

The following pseudo code algorithm fragment will print "Y" and nothing else when Boolean condition B1 is true.

```
if  B1  then   print "Y"
else
   if   B2  then
     begin
        if  B3  then  print "Z"
        else   print "X"
     end
  else   print "X"
```

Question 1: Give the logical expression (in terms of B1, B2 and B3) when "Z" is printed.

Question 2: Give the logical expression (in terms of B1, B2 and B3) when "X" is printed.

Keywords: Propositional logic, IF-THEN-ELSE

Type: Lecture example, homework and/or lab assignment, exam question

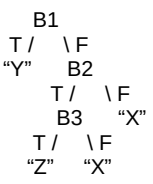
Solution: Question 1: $\sim B1 \wedge B2 \wedge B3$ Question 2: $(\sim B1 \wedge B2 \wedge \sim B3) \vee (\sim B1 \wedge \sim B2)$

Pedagogical Notes: Students might claim to understand the logic of flow of basic flow of control structures; however, requiring them to express the logic explicitly will ensure a better understanding of the relationship between logic and programming.

There are numerous variations of this problem you can make up yourself, including reversing the question. For example, ask them to give the pseudo code corresponding to the following given three Boolean variables B1, B2 and B3:

Print "X" when B1 is true
Print "Y" when $\sim B1 \wedge B2 \wedge B3$ is true
Print "Z" when $(\sim B1 \wedge B2 \wedge \sim B3) \vee (\sim B1 \wedge \sim B2)$ is true

Another variation is constructing/using a truth tree where it might be easier for students to see the flow:



Learning Outcomes: Learn the logical correspondence between if-then-else flow of control structures and propositional logic.

Connections: Truth tables, logical flow of control, decision tables

Prerequisite Knowledge: Basic propositional logic, IF-THEN-ELSE control structure

Source: Dr. Richard Botting, email posting the math-thinking list server 7/22/02

6 Sustaining the Effort

Based on our own observations and those of others, there will need to be a long-term effort to maintain, update and populate the repository. Getting people to volunteer to be topic editors and asking them to develop their own examples and to solicit examples from others will help, but it will not be enough if the goal is to have a large repository for use by mathematics and computer science educators.

All too often there is an initial burst of enthusiasm for building a repository which leads to some good material being developed, but the effort tends to decline when the reality of what it takes to maintain such a repository sets in. (We are all very busy people.) What is required is an approach that will involve a much larger group of contributors than those who are members of this Working Group or even the Math-Thinking Group [1].

What follows is a list of suggestions for sustaining the effort. All of these will involve in some form or other developing materials to be placed in the repository.

- Organize Birds of a Feather Sessions at SIGCSE and regional conferences of CCSC
- Organize a Special Session, Tutorial or Nifty Assignments Session at SIGCSE
- Organize a Contributed Papers Session at a Winter or Summer meeting of the MAA
- Develop an NSF CCLI proposal for a series of workshops with participants from the undergraduate computer science and mathematics communities
- Develop proposals to obtain funding at the state and local levels for workshops with participants from the high school community
- Submit a proposal to a federal agency, such as NSF or FIPSE, to support further development and population of the repository. A significant amount of money and effort has been devoted to creating and populating similar repositories for science, engineering and mathematics education.
- Ask authors of current textbooks in computer science and discrete mathematics to contribute examples
- Invite the SIGCSE membership to scour the Web for relevant and useful examples
- Ensure other computer science, engineering and mathematics repositories provide links to the repository

7 Other Issues

The previous sections provide an overview of the proposed repository. Developing a fully functional implementation will require significantly more effort and addressing many issues. These include, among others, developing a meaningful yet flexible problem classification system, learning about metadata systems and their associated tools, defining the metadata template, initial creation and population of the repository, security, mechanisms for review of material, use of consistent terminology, copyright issues, and defining long range plans.

For the discrete mathematics component of the repository, the CC2001 classification scheme can be used. However, for material at the interface between mathematics and computer science, where it might be used for either mathematics and/or computer science classification might be a bit harder. Consider the IF-THEN-ELSE-logic example. For students who have taken a programming course before discrete mathematics, this example would fit nicely into the latter to illustrate the relevance of mathematics. However, the reverse might also be the case, and this example could be used in CS I. Some of the material many not fit conveniently into existing classification schemes, and we will have to define our own.

Members of the working group, having studied various metadata-based repositories, must still understand how to create a functional repository. This endeavor will entail developing a prototype repository that will be evaluated by potential users and fine-tuned accordingly. We anticipate the repository will be at one central site, with mirror sites. Distributed repositories are difficult to maintain due to broken links (e.g., server down, URL changes, etc.) and uncontrolled updates can quickly lead to potentially inconsistent and insecure material. As noted earlier, topic editors will have responsibility for adding/updating material related to their area.

Editors will be provided with a set of guidelines to ensure consistency of material with respect to level of detail, presentation, terminology, etc. This should address some of the concerns noted by survey respondents presented in Section 8.

Security has several meanings in this context. One is ensuring that the site cannot be corrupted by others. The second refers to access to the material. Here there are several options: open access to everyone, user/password access without user verification such as used in CITIDEL, and verification that a user is a bonafide instructor. Our current plan is to place this as a subdirectory of CITIDEL, so the second option will be used. If the repository becomes widely used and instructors have concerns about student access to solutions, then one option is to require an additional level of security for access to solutions.

Quality control is an important issue for repository users. The old saying "One bad apple spoils the whole bunch" holds for repositories. The proposal for the SIGGRAPH educational material repository [9] requires several levels of peer reviewing. Our thoughts are less rigorous (shouldn't a math group be more rigorous?). As currently envisioned there will be three progressive classifications of material: 1) un-reviewed, 2) peer reviewed by at least two people, and 3) class tested. Un-reviewed material might be distributed – that is, stored at the contributors' web site. It will be clearly noted as un-reviewed -- users beware, and the search engine will only include un-reviewed contributions upon request.

Issues associated with copyrighted material must be dealt with. Many textbook problems are generic; however, some problems developed by authors or posted on textbook resource pages may require permission to include in the repository. It is hoped that this repository will serve as an incentive for authors to post their best problems as a way of gaining recognition for their textbooks.

8 On-line Repository Survey Results

The working group conducted a survey of the audience at its preliminary presentation at ITiCSE 2002. The primary objective of this survey was to determine if computer science educators make use of on-line repositories of educational material, and if so, to evaluate their experiences and impressions of these repositories.

We surveyed 54 educators at the working groups presentations. 90.7% of them are aware of on-line educational repositories of materials for computer science education. But, only 38.8% of the respondents found them convenient to use, and only 35.1% have used them effectively to provide examples or problems for their courses. Educators using repositories mainly use them for lecture examples (29.6%), supplementary material (27.7%), laboratory assignments (16.6%) and homework assignments (11.1%).

For technologically savvy educators, these results are a bit discouraging. However, computer science repositories are relatively new, it does take a while for users to discover and feel comfortable with them, and there is an abundance of educational resources available in printed form – text books, instructor manuals, etc.

We went further in our study in trying to understand why educators do/do not use and find repositories convenient, and what features they would like for on-line repositories. This information is very important for our purpose. Those using repositories like the fact that on-line information is easily accessible by everybody, both instructors and students, and often present different points of view. As limitations respondents cited a lack of range of information, the difference of terminology, the difficulty to find appropriate and high-quality material and to adapt material to local situations. Constant updates, better search, quality control, coherent structures and organizations, and examples of use are additional important features they would like for an on-line repository.

The complete survey, with a more detailed summary of results, is presented below.

SURVEY
Working Group: Materials Development in Support of Mathematical Thinking

1. Are you aware of on-line educational repositories (including publishers') of materials for CS education? Yes 49 No 7

2. Have you used any of these on-line repositories to provide examples or problems for your courses? Yes 19 No 34

3. Have you generally found these on-line repositories convenient to use? Yes 21 No 17

4. Please list the names and/or URL's of the repositories you have used:

5. For what purposes did you use the repositories? Circle all that apply

Lecture examples 16 homework assignments 6 lab assignments 9

Exam/quiz questions 0 supplementary material 15 group/team projects 0

Other (please specify): 0

6. What did you find most convenient/useful about the repositories you have used?

On-line = easy access, open standards, depth of material, getting ideas, time saving/reusability, quick overview, textbook figures/illustrations/graphics to help prepare teaching materials, students can explore on their own, obtaining PowerPoint presentations, range of materials.

7. Give one difficulty you encountered with a repository you have used.

Difficult to adapt to a desired situation, varying underlying systems & languages, finding materials that are appropriate, too much material to select from, no relevant material found for desired topic or course, finding high quality materials, different components of archive use different formats, lack of breadth, inconsistent terminology, poor search tools, slow, complex to use, required passwords.

8. What additional features would you like in an on-line repository?

Homework problems, lab projects, quality control, breadth and depth, frequent updates, better indexing/browsing, example of how material can be used, links from other resource portals (e.g., ACM), complete packages – including slides, easy to find, convenient to update, (subject, keywords, examples, goals, solutions), organized, discussion groups, broader range of materials, hints on how material is organized, better search capabilities, comparison with other institutions.

References

[1] Baldwin, D. and Henderson, P. Math-thinking discussion group web site, http://www.math-in-cs.org.

[2] CITIDEL (Computing and Information Technology Interactive Library Project), http://www.citidel.org.

[3] Computing Curricula 2001, http://www.acm.org/sigcse/cc2001. December 2001.

[4] CSTC (Computer Science Teaching Center), http://www.cstc.org.

[5] Cut the Knot, http://www.cut-the-knot.com/.

[6] GEM (Gateway to Educational Materials), http://www.geminfo.org/.

[7] Glass, Robert L., "A New Answer to 'How Important is Mathematics to the Software Practioner'?" IEEE Software, November/December 2000, pp. 135-136.

[8] Henderson, P.B., Baldwin, D., et al, ''Striving for Mathematical Thinking,'' ITiCSE 2000 Working Group Report, SIGCSE Bulletin - Inroads, Vol. 33, No. 4, Dec 2001, pp. 114-124.

[9] Hitchner, L., Personal Report of the Computer Graphics Education 2002 Workshop.

[10] IEEE Learning Objects MetaData, http://www.ischool.washington.edu/sasutton/IEEE1484.html.

[11] JERIC (Journal of Educational Resources in Computing), http://www.acm.org/pubs/jeric/.

[12] Kelemen, C., Tucker, A.B, Henderson, P.B, Bruce, K., Astrachan, O., "Has our curriculum become Math-phobic? (an American perspective)", Proceedings of ITiCSE 2000, pp. 132-135.

[13] Kelemen, C., Tucker, A.B, Henderson, P.B, Bruce, K., Astrachan, O., Baldwin, D., Skrien, D., Van Loan, C., Computer Science Report to the CUPM CurriculumFoundations Workshop in Physics and Computer Science, http://www.cs.swarthmore.edu/~cfk/cupm2.pdf.

[14] Math Forum, http://mathforum.org/.

[15] NSDL (National Science, Technology, Engineering, and Mathematics Digital Libraries), http://www.nsdl.nsf.gov.

[16] Tucker, Allen B., Kelemen, Charles F., and Bruce, Kim B., "Our Curriculum Has Become Math-Phobic! ", Proceedings of the Thirty Second SIGCSE Technical Symposium on Computer Science Education, Charlotte, North Carolina, Feb. 21-25, 2001, pp. 243-247.

[17] The UK's Metadata for Education Group, http://www.ukoln.ac.uk/metadata/education/.

[18] The Dublin Core Metadata Initiative, http://dublincore.org/.