Editors: Michael Dennin, mdennin@uci.edu
Steven Barrett, steveb@uwyo.edu

POSTPROCESSING IN AUTOMATED GRADING SYSTEMS, PART 1

By Peter A. Knipp and S. Raj Chaudhury

In this first installment on postprocessing, we discuss the manner in which the standard grading algorithm can be generalized in order to increase the power and effectiveness of automated grading systems.

Online homework systems such as WebAssign (www.webassign.net/info) provide instant feedback to students, eliminate manual grading by instructors or teaching assistants (TAs), and reduce solution sharing among students. For most systems, the grading algorithm uses a straightforward comparison of the student's submission with an answer key, but in many situations, this algorithm is less appropriate. Over the next three installments, we discuss an alternate approach to automated grading that uses "postprocessing." In this first installment, we provide an introduction to postprocessing; in the second installment, we'll discuss how to most effectively use postprocessing with lecture class assignments; and, in the third installment, we'll show how instructors can use postprocessing in lab courses.

Background

Class assignment grading is an important course component. Despite the consensus that working through homework problems is valuable for learning science and engineering, the completion rate of ungraded homework is much lower than that of graded homework. Most instructors would agree that students are far less likely to complete homework if it's not collected and graded. The time cycle for homework is usually about one week, and historically, TAs grade it manually. However, because of timing and labor issues, only one round of grading is feasible, so the student doesn't receive any feedback until after an assignment deadline. As such, the assessment formed by the homework process isn't formative—that is, the student rarely goes back and corrects the assignment based on the grader's feedback. The advent of online automated homework-grading systems such as WebAssign has considerably improved the situation for instructors by eliminating manual grading and for students by providing immediate feedback and the possibility of multiple submissions.[1,2] Here, we describe the various ways in which we use relatively simple postprocessing topologies to grade assignments that the basic paradigm can't handle.

Standard Online Homework

Most online homework systems facilitate the logistics of individualizing, distributing, collecting, grading, and keeping records of assignments for numerous students. Figure 1 shows an example of a standard question that consists of two independent parts, which most homework systems can handle (www.pcs.cnu.edu/~pknipp/cise/supplemental). The flow of information in such a question is as follows:

- *Coder*. Instructor defines quantities or functions used in the algorithm to assess the correctness of a student's submission. The system usually bases the calculation of these entities on input values that it displays to the student in the question's text, and it randomizes the input values to prevent students from copying the correct submissions from one another. For example, the question might ask, "What's the area of a square whose side length is L?" where L is a different number for each student. In this case, the value of the answer key X equals L^2.
- *Student*. Students enter their submissions. For WebAssign, this submission is in the form of a number (for a numerical question), a text string (for a fill-in-the-blank, symbolic, or essay question), the selection of a radio button (for a multiple-choice question), or a combination of toggles (for a multiselect question). For simplicity's sake, most of our notation, nomenclature, and figures will assume the student's submission is a number.
- *Algorithm*. The system determines the correctness of a student's submission, which usually involves calculating the absolute value of the difference between the key X (calculated by the coder) and the student's submission x and comparing this difference with a tolerance the

 1521-9615/09/$25.00 © 2009 IEEE

instructor determines. For WebAssign, the default tolerance setting is 2 percent of the key.

- *Grade*. The grade G is the algorithm's output—usually, $G = 0$ (if incorrect) or 1 (if correct).

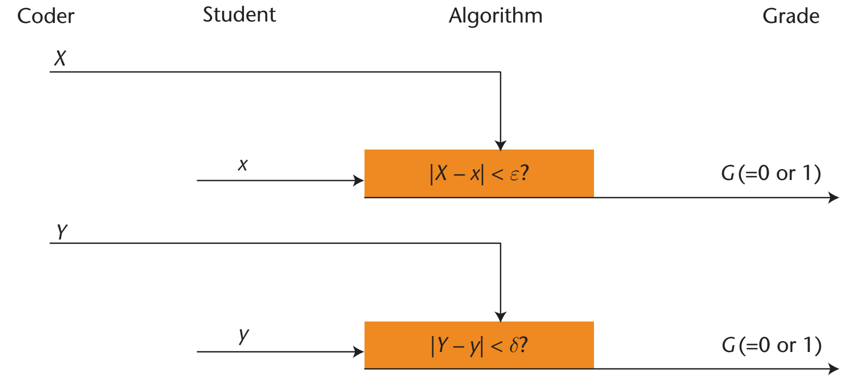

Figure 1. A standard two-part question flow chart. Central to each part is its grading algorithm, whose inputs are the key X (from the question coder) and the submission x (from the student) and whose output is the grade G.

Figure 1 shows a system that can assess most high school and lower-level college lecture course content. However, timely and detailed assignment grading is just as important in lab courses and upper-level lecture courses, and the grading paradigm for those assignments doesn't always map well onto the steps that the figure shows. Thus, here we extend online homework grading algorithms beyond the basic topology depicted in Figure 1.

We use the term *postprocessing* to describe any algorithm whose use of the student's submission x doesn't simply involve a determination of whether x is sufficiently close to the answer key X. With WebAssign, postprocessing always involves the variables `$thisresponse`, `$thisnum`, `$thisunit`, `$thisanswer`, or the function `get_data` (each of which captures the student's submission). It often involves `$THIS_SCORE` (a flag whose initialization overrides WebAssign's default grading algorithm), and it occasionally involves `$MARKOFF` (a flag that turns off WebAssign's correctness indicators √ and ×), `$HINT` (an intuitively named string discussed later), or `$HINT_ON_CORRECT` (a flag that signals WebAssign to display the value of `$HINT` even if the student's submission isn't incorrect). In most cases, the answer field at the question level needs fewer than six additional lines of Perl (WebAssign's engine) to implement postprocessing in a question on a typical science assignment.

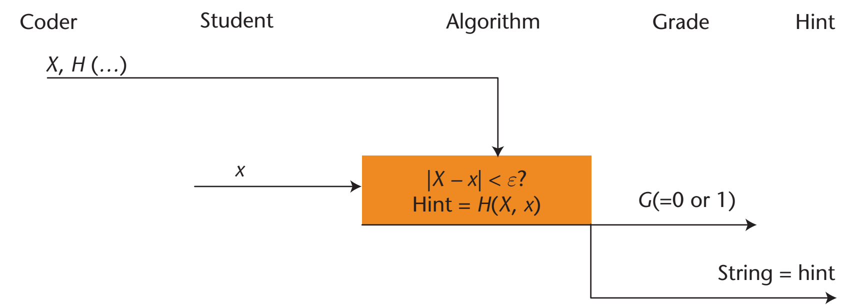

Figure 2. Flow chart of a question that provides conditional feedback to the student. In addition to checking if the student's submission x is sufficiently close to the coder's key X, the algorithm also executes the coder's function H to determine whether x is close to one of several anticipated common mistakes to provoke a corresponding hint response.

The Automated Grading System Paradigm

The most widespread way that online homework systems use postprocessing is by enabling hints. Figure 2 shows a typical application of such a hint, which in this case is a string of text that the system displays to the student to provide guidance as to why the submission is wrong. For example, if the question asks the student for the area of a square whose side is length L, the function H detects whether the student's submission x is sufficiently close to either $4L$ (the perimeter) or $\sqrt{2}L$ (the diagonal); in either case, the value of H is a corresponding string (`$HINT`), such as "You've calculated the perimeter, not the area." If x differs sufficiently from both the key X and all anticipated common mistakes, then the best possible string would be either the null string or "It isn't clear why you submitted this answer." We'll demonstrate more specialized hint implementations over the next two installments.

To demonstrate the power and flexibility of postprocessing, consider the type of homework question for which it's much more difficult to generate the key X than to confirm that a student's submission satisfies the question's statement, such as *inverse problems*. Physics courses (especially upper-level ones) often incorporate inverse problems, and postprocessing can readily handle most of these. For example, it's usually much easier to confirm that a particle's

Other Online Homework Systems

Although this list is by no means exhaustive, it captures the different flavors of systems that are prevalent amongst most science and engineering departments. Some customized components of course management systems such as Blackboard and Moodle might possess some of the functionality of dedicated homework systems such as WebAssign, but we haven't encountered the public view of such solutions.

- Andes Physics Tutoring System[1] appears to have some of the postprocessing features we describe in the main text, but this system evolved from intelligent tutor software and has had limited institutional exposure,
- LON-CAPA[2] (Learning Online Network with computer-assisted personalized approach) is an open source e-learning platform that provides access to extensive problem sets with strength in physics and chemistry.
- Mastering Physics (www.masteringphysics.com) is a commercial product with features that include coaching students on problem-solving techniques and extensive data management and diagnostic tools.
- OWL (www.cesd.umass.edu/CESD/OWL) provides delivery and grading of electronic homework assignments, rich content authoring, and student management tools.
- University of Texas Homework Service,[3] is a free service that maintains question banks on a variety of topics, from fifth grade mathematics through calculus-based college physics. An authenticated registration procedure is required for instructors.
- WebHW,[4] is a Web-based system designed primarily to facilitate submission, grading, and return of homework assignments in electronic format. Its focus is on being a simple tool for instructors, TAs, and students, and
- WeBWorK (webwork.rochester.edu/docs/docs) is a free, Web-based system for generating and delivering individualized homework problems to students; its goal is to make homework more effective and efficient; it has a modular structure and is based on Perl and thus holds potential for implementation of the kinds of techniques described in these installments.

References

1. K. VanLehn et al., "The Andes Physics Tutoring System: Lessons Learned," *Int'l J. Artificial Intelligence Education*, vol. 15, no. 3, 2005, pp. 147–204.
2. G. Kortemeyer et al., "Experiences Using the Open Source Learning Content Management and Assessment System LON-CAPA in Introductory Physics Courses," *Am. J. Phys.*, vol. 76, nos. 4–5, 2008, pp. 438–444.
3. Homework Service, Univ. of Texas, https://hs.utexas.edu/bur/overview.html.
4. C. Bennett, "Multipart Homework Problems over the Web," *Computing in Science & Eng.*, vol. 2, no. 1, 2000, pp. 87–91, 95.

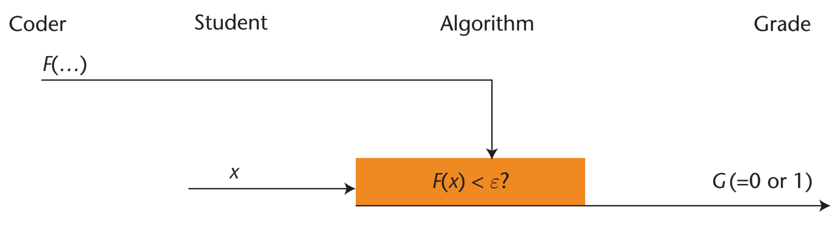

Figure 3. Flow chart for an inverse problem. The grading algorithm simply uses the student's submission x as the input for a given function F to confirm that x satisfies the problem statement.

trajectory $x(t)$ satisfies Newton's second law (a differential equation) than to solve the differential equation itself. Consider the following simpler example: "Find a root of the polynomial $x^5 - x + 1$." Although the solution $X = -1.1673\ldots$ can't be obtained by an analytic procedure, Figure 3 shows a straightforward way to confirm that the student's submission x satisfies the desired criterion—namely, to calculate $F(x) = |x^5 - x + 1|$ and to ascertain whether F is sufficiently small.

In this context, $F(x)$ becomes a generalization of the function $|X - x|$ used in Figures 2 and 3. This postprocessing strategy is also viable for a question that has nonunique answers, such as finding any root of $x^5 - 2x + 1$ (which has three real roots).

In this installment, we've introduced some ways in which the ability to postprocess information in an automated grading system expands the set of problem types instructors can assign. In the next two installments, we'll cover the following topics:

- Awarding a grade G ($0 \leq G \leq 1$) which has a continuous dependence upon the student answer x, rather than G's simply equaling 1 if $|X - x|$ is arbitrarily small or 0 otherwise. (In some circumstances, this continuous grading method might set a more realistic expectation for the student's performance.).
- Grading simultaneously two or more question parts, either to discourage guessing or to handle a question whose solution is a nonunique array of data.
- Passing information from ear-

Erratum

In March/April 2009, Vol. 11, No. 2, the Education department article ("Drawing Chemical Equipment with Adobe, Part 3: Gradients, Retouching, and More Objects," by Daniel Tofan) has an incorrect URL. The correct URL is http://opac.ieeecomputersociety.org/opac?year=2009&volume=11&issue=2&acronym=cise. We regret the error.

lier questions to later questions to award partial credit in a fair and accurate manner for a multistep question in which errors in a student's submission for earlier parts would otherwise doom his or her answers to later parts.
- Handling the randomization of equipment in lab courses.
- Letting a student enter an unspecified quantity of data in a single answer box.

We've used WebAssign to implement these techniques in introductory physics laboratory courses and in some upper-level physics lecture courses, although we haven't yet explored the potential for this type of customization in any other automated grading system. These techniques have been successfully tested at CNU over the past eight years in course sections taught by a dozen different instructors and have helped to provide student feedback in labs and reduce the grading burden in all cases. We hope that open source homework systems (such as LON-CAPA or WeBWorK—see the sidebar) will include the programming features that make postprocessing possible in WebAssign.

References

1. A.P. Titus et al., "Web-Based Testing in Physics Education: Methods and Opportunities," *Computers in Physics*, vol. 12, 1998, pp. 117–123.
2. R.J. Dufresne, J.P. Mestre, and K.A. Rath, "The Effect of Web-Based Homework on Test Performance in Large Enrollment Introductory Physics Courses," *J. Computers in Mathematics and Science Teaching*, vol. 23, no. 3, 2002, pp. 229–251.

Peter A. Knipp is an associate professor of physics at Christopher Newport University. His research interests include surface physics, semiconductor nanostructures, and learning technologies. Knipp has a PhD in physics from University of Chicago. Contact him at pknipp@pcs.cnu.edu.

S. Raj Chaudhury is the associate director of the Biggio Center for the Enhancement of Teaching and Learning at Auburn University. His research interests include learning technologies, science education, and faculty professional development. Chaudhury has a PhD in physics from University of California, Los Angeles. Contact him at schaudhury@auburn.edu.