Adaptive Rubrics

Marco Carmosino
marco@ntime.org
Simon Fraser University
Burnaby, British Columbia, Canada

Mia Minnes
minnes@eng.ucsd.edu
University of California San Diego
La Jolla, California

ABSTRACT

Grading is a notoriously difficult and time-consuming part of teaching. For open-ended programming, mathematical, or design problems, assigning consistent scores and giving useful feedback can be very challenging. Large classes compound this difficulty. Adding TAs to the team can help parallelize the process but may impede grading consistency and quality. We present an adaptive rubric creation and application process to enable high-quality responses to student work, at scale. This process uses exploratory data analysis to discover common patterns in student responses to a problem, then tailors a rubric and feedback to address these patterns. Our method is supported by current grading tools, which allow calculation of the simple population-level statistics we need to extract meaningful features from a corpus of student work. In this case study, we describe using adaptive rubrics for a discrete math class for CS majors: the grading team found that this process produced concrete and transparent justifications of student scores and that it facilitated conversations around grading that were grounded in course learning objectives and values.

CCS CONCEPTS

• **Social and professional topics** → **Student assessment**; • **Applied computing** → *Learning management systems*;

KEYWORDS

Grading, rubrics, educational data mining, large classes

ACM Reference Format:
Marco Carmosino and Mia Minnes. 2020. Adaptive Rubrics. In *The 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20), March 11–14, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3328778.3366946

1 ATTENTION IS VALUABLE AND LIMITED

The more complex a problem, the more difficult it is to grade. Grading involves both evaluation of specific student work and communicating with students about this evaluation. Grading decisions should reflect the extent to which student work achieves the criteria for the assignment, informed by the priorities of the course.

SIGCSE '20, March 11–14, 2020, Portland, OR, USA

ACM ISBN 978-1-4503-6793-6/20/03…$15.00
https://doi.org/10.1145/3328778.3366946

One of the difficulties of grading is making these decisions *consistently*. When classes simultaneously scale up and assign complex problems — as in computer science (CS) assignments that require rich responses like proofs, design documents, or architectures — it is common to use rubrics and to enlarge the grading team (by adding TAs or student graders). Typically, consistent application of the rubric requires multiple rounds of instructor-grader conversations, since the values of the course (and their instantiation as point allocations) are ultimately the instructor's responsibility. In this project, we propose a grading framework with limited, high-utility grader-instructor interaction rounds. Through this process, a rich lexicon emerges that serves two important purposes:

- **Articulate learning objectives and gaps:** Construct effective and efficient feedback for students to facilitate learning.
- **Assess population:** Cluster common approaches to the assignment and identify common misconceptions using succinct and justifiable properties of student work.

We reuse a generic rubric, by adapting and customizing it for each assignment and each set of students in a principled, data-driven process. In these adaptive rubrics, meta-data accompany rubric items to encode rich information about assessment decision boundaries used to classify student submissions. In tandem, we record student-facing comments using concrete vocabulary and advice for incremental improvement. This framework is inspired by exploratory data analysis. Instead of pre-populating a rubric with hypothetical predictions about student work, the actual patterns in student responses inform the definition of rubric items. Point assignments are based on assessing common responses using the course learning outcomes. Though we describe our method in the context of a CS Discrete Math course, it could be adapted to any course or major.

2 BACKGROUND AND RELATED WORK

Grading serves two important purposes: feedback on student work can enhance learning [5, 11, 14] and the scores assigned to student work can certify whether students demonstrate required learning outcomes [22, 23]. Balancing these important functions of grading with practical constraints is hard: SIGCSE members lament challenges and crowd-source grading advice [1, 2]; Kumar's opinion piece in ACM Inroads cautions against the efficiency of auto-graders in the face of uncertain impacts on student learning [10].

In CS, open-ended problems often combine some objectively grade-able components (does the submitted code pass test cases? is each step of the candidate proof an application of valid reasoning?) with subjective judgments informed by community values (are functions modular? is the proof elegant?). Grading with rubrics can help evaluations be fair, consistent, and reliable [3, 9, 18].

Rubrics typically describe criteria for student work with associated levels of achievement for each criterion. Crating rubrics tailored to specific assignments may be time-consuming [4, 6]. For example, to design a reliable rubric for a programming assignment, Stegeman iterated a candidate rubric through several stages of review and grading by multiple instructors until enough detail could be incorporated [21]. Examples in the Physics (e.g. [12, 16]) and Mathematics (e.g. [8, 13]) education literature demonstrate that opinions around justification, rigor, and beauty inform grading choices (even in the presence of rubrics) and can lead to inconsistent student assessment. Grading with adaptive rubrics is designed to facilitate consistency while making the process efficient enough to be practical in the weekly or bi-weekly assignment cadence of a typical course. Our adaptive process refines the exam grading by clustering method of [15]; in the context of formative weekly homework, more granular feedback to the students is appropriate.

We also take inspiration from the open source community norms for structuring discussion of complex source-code artifacts on text-only mailing lists [17]. Through the adaptive rubric process, precise, informative, and explicit questions are formulated for grader-instructor conversations, and answers are recorded.

The adaptive rubric grading process is supported by emerging educational technology. Learning management systems (LMS) and other tools now facilitate (many aspects of course delivery, including) grading. Vocareum, Canvas, Blackboard, and Gradescope each have mechanisms for creating, editing, and applying rubrics to student submissions. During this case study, we used Gradescope [7] because our institution has a site license and the students are comfortable with it. Students uploaded their homework submissions and course TAs read, gave feedback, and assigned scores through the Gradescope interface. Any LMS that supports marking assignments using custom rubrics is compatible with our process.

3 CASE STUDY

We used the adaptive method to grade one problem per homework in a 10-week undergraduate Discrete Mathematics for CS course at a large public research-intensive institution in the US. This course is required for all CS majors and is typically taken by freshmen and sophomores. It is taught in relatively large lectures, with approximately 150 students per lecture section and multiple lecture sections offered each term. A central learning outcome of the course is precise communication about mathematical objects, algorithms, and arguments. Weekly assignments require students to fluently translate between different notations and levels of formality, critically evaluate claims, and provide evidence for arguments.

During Spring 2019, each of the eight homework assignments included at least one open-ended question. We selected the problem that seemed to require the richest response (usually a proof) for the adaptive rubric grading method. This case study focuses on the grading of a proof by structural induction about the linked list data structure. This question was assigned two-thirds of the way through the course. There were 233 distinct submissions: 296 students submitted work, with some working in pairs. One TA was assigned to grade all submissions for this problem. This TA had been developing adaptive grading across five previous terms and had significant experience grading proofs in general. They spent approximately 8 hours grading this problem, over one week. Below, we include the relevant definitions and question.

Definition 3.1 (Linked Lists). The set of linked lists of natural numbers L is defined by:

Basis Step:	$[] \in L$
Recursive Step:	For $l \in L$ and $n \in \mathbb{N}$, $(n, l) \in L$

Definition 3.2 (List Functions). The function $len : L \to \mathbb{N}$ that computes the length of a list is:

$$len(\ell) = \begin{cases} 0 & \text{if } \ell = [] \\ 1 + len(\ell') & \text{if } \ell = (n, \ell') \text{ where } n \in \mathbb{N} \text{ and } \ell' \in L \end{cases}$$

Definition 3.3 (List Increment). The function $inc : L \to L$ that adds 1 to each element of a linked list is defined by:

$$inc(\ell) = \begin{cases} [] & \text{if } \ell = [] \\ (1 + n, inc(\ell')) & \text{if } \ell = (n, \ell') \text{ where } n \in \mathbb{N} \text{ and } \ell' \in L \end{cases}$$

Definition 3.4 (List Sum). The function $sum : L \to \mathbb{N}$ that sums all the elements of a list is defined by:

$$sum(\ell) = \begin{cases} 0 & \text{if } \ell = [] \\ n + sum(\ell') & \text{if } \ell = (n, \ell') \text{ where } n \in \mathbb{N} \text{ and } \ell' \in L \end{cases}$$

Homework Question: Prove or disprove the following statement. You may **not** use $\cdots$ or $\sum$ notation:

$$\forall l \in L\,(sum(l) + len(l) = sum(inc(l)))$$

Using these definitions, students practice the "full loop" of communication and reasoning skills which are a core learning outcome of the course. By disallowing the use of $\cdots$ (dots) or Σ (summation) notation, the question exercises technical facility with and understanding of induction, another core learning outcome of our course. The question is difficult for students to complete. They must take multiple steps to even understand what is being asked and to realize that the statement is true. The correct solution is then a proof by induction, which is more abstract than earlier proof techniques.

The question is difficult for instructors to grade. Because responses are completely free-form, we must recover the intended proof from whatever text students submit. Our rubrics need take both correctness and clarity into account; but how?

In this experience report, we detail the grading of this specific question as an example of the adaptive rubric process. We reflect on our experience using this process, highlighting the ways in which it improved the quality and consistency of grading decisions: we were able to develop a rich and specific lexicon, informed by actual student data, to discuss the nuances of the space of student responses and then make grading judgements informed by both these data and course learning objectives and values.

3.1 Generated Initial Tags

In the first phase of the adaptive rubric process, we recorded features and statistics to describe the specific question and how students responded to it by assigning *tags* to student responses. Each tag has three parts: a name, a description, and a list of *justifications*. Each justification is a short sentence that connects specific text in a response to the tag. For each new tag we defined, we created a

(placeholder) rubric item in Gradescope with its name and description, and we used a text editor (Bear [19]) that supports outlining features to keep track of its list of justifications.

We began with a sample canonical solution (generated by the course instructor) and a generic rubric for induction problems:

5 pts Proof by structural induction clear, complete, and correct: variables clearly declared, base case and induction step each labelled, assumptions and goals clearly articulated, calculations well supported and explained.
4 pts Proof by induction mostly clear, complete, and correct.
3 pts Proof by induction has many required components but missing major component, e.g. basis step missing or wrong, or IH not stated or used, or wrong proof.
2 pts Some indication of structure of proof by induction but main arguments missing or irrelevant.
0 pts Incorrect or blank.

Tagged the Canonical Solution. The grader generated tags describing the canonical solution, see Table 1, along with short justifications. These tags formed a "checklist" of predicates aligned with the **5 pts** generic rubric item. The process of carefully exploring "decision boundaries" associated with each tag for the specific problem being graded drives adaptive grading.

Tag	Description	Kept
`#IT`	Indicate statement is true	★
`#IH`	Clearly indicate a correct inductive hypothesis	★
`#BS`	Clearly indicate a correct basis step	★
`#CCB`	Correct conclusion from basis step	
`#RS`	Clearly indicate a correct recursive step	★
`#CCR`	Correct conclusion from recursive step	
`#CC`	Clear and correct calculations overall	★
`#LD`	Correct usage and reference to list definitions	★
`#MI`	Mention "induction"	★
`#CS`	Correct inductive structure	

Table 1: Tags generated from canonical solution. The Kept column flags whether this tag was used in the final rubric.

At this phase of grading, the only justification in the list of justifications for each tag was the specific way features appeared in the canonical solution. The next stage of the process added new justifications representing alternate approaches.

Tagged a Sub-Sample. The grader drew a random sub-sample (about 25%) of student submissions to continue the initial tagging process; this sample-size seems (empirically) to ensure that the Initial Tags are expressive enough to produce a good rubric. We wrote a javascript bookmarklet for the Gradescope "list assignments" page to open a random subset of assignments in new browser tabs. For each response in this sub-sample, we attempted three tasks:

- **Assigning existing tags.** We attempted to fully describe the response using existing tags, adding justifications to the list to record variations in how student work specifically exhibited the tag. For example, two different formulations of the induction hypothesis are correct for this statement; each would be a different justification in the list for the `#IH` tag. As our understanding of the patterns in student responses shifted, the justifications for tags evolved and sharpened.
- **Identifying Mistakes.** For responses that didn't meet the **5 pts** criteria, we created tags (referencing the generic rubric, and including specific justifications), see Table 2, to identify exactly why a response was lacking. We were not scoring the responses during this phase; we were simply asking *why* they were wrong and capturing the answer with a tag.

Tag	Description	Kept
`#MIH`	Malformed inductive hypothesis	
`#MEq`	Malformed equational reasoning	★
`#ExRC`	One-element example as recursive step	
`#?LDC`	Link a definition without case OK?	
`#???`	Totally baffling	

Table 2: Tags generated from sub-sample. The Kept column flags whether this tag was used in the final rubric.

The tag `#MIH` described inductive hypotheses with serious type errors or other failures to parse: the recorded justifications for this tag distinguished between these cases. The other "mistake tags" in Table 2 will be described below to illustrate additional stages of the process.

- **Concentrating Uncertainty.** Sometimes, tags flagged a specific grading question that appeared in multiple student assignments. For example, the question tag `#?LDC` recorded confusion about the decision boundary for the tag `#LD`. The tag `#LD` meant that the definitions for linked lists given in the problem were clearly cited and correctly used to justify any manipulation of list constructions. Some student submissions referenced these definitions but did not specify which part of the definition (base case or recursive step) was germane to the justification. Did a response with some, but not fully detailed, use of the list definitions earn the `#LD` tag? The tag `#?LD` and other question tags recorded this question for subsequent grader-instructor conversations.
  Sometimes, it was very difficult to understand student responses. If the grader could not make sense of a submission quickly (our threshold was under a minute), they assigned the special `#???` tag and did *not* track justifications. This "timeout" let us focus on overall patterns without getting stuck attempting to decode bizarre responses. Mercifully, the `#???` tag is not too common. For this case study, about 12 assignments of the 58 in the sub-sample received that tag. Grading of submissions tagged with `#???` was deferred until the very end of the grading process, at which point course staff have a better chance of interpreting these responses and giving students meaningful individualized feedback.

A heuristic for setting the initial sample size is to count how many responses have been observed without making a new tag. The larger this count, the more likely that the tag-set is "stable."

Computed Population Statistics and Pruned Tags. Once the sub-sample was tagged, we viewed aggregate population statistics about the occurrences of each tag. The statistics page for each question in

a Gradescope assignment displays bar charts plotting the frequency of each tag. As soon as we consulted these charts, patterns emerged.

It was clear that the #?LDC tag was common enough to warrant a conversation between the grader and the instructor. Other "question tags" that had arisen (not listed here) either had too few occurrences to be a discussion priority, or could be resolved by the grader after working through enough of the grading.

We also pruned other tags: #CCB, #CCR, and #CS were dropped at this point. These "correct conclusion" and "correct structure" tags were perfectly correlated with other tags so they added no information. The pruning of tags is specific to the actual question being graded. For the property of linked lists in this homework, and this specific class of students, whether students articulated the IH correctly was perfectly correlated with successfully applying it. This may not be the case with other induction problems.

Another tag was pruned at this stage, for a different reason. Only a small fraction of the sub-sample was tagged malformed inductive hypothesis #MIH. For the most part, students produced a correct IH or no IH at all. This observation shed light on students' general understanding of linked lists and induction and was discussed by the instructional team in the next phase of the process.

3.2 Made Course-Values Decisions

Once the sub-sampled responses had been tagged and summarized, the grader formulated a concise and pointed list of questions to refine the decision boundaries for tag applications. These questions were discussed among course staff. Often, resolving questions meant classifying what should constitute a mistake: establishing these course norms is the responsibility of the instructor. In our example, the instructor determined that "deep references" into recursive definitions by cases were adequately emphasized by other assignments and so responses tagged #?LDC could instead be tagged #LD (and the tag #?LDC removed).

The common tag #MEq prompted a longer discussion. This tag represented student submissions which structured proofs by *first* asserting the desired conclusion, and then rewriting it by manipulating both sides of the equation. This approach often looks deceptively similar to the goal-rewriting proof strategy, except that students tend to omit any indication that what is being rewritten is the *goal* (and has not yet been established). Without such caveats, this type of reasoning can quickly become circular. Since these submissions indicated an underlying misconception about the different roles of premises and goals in proofs, the instructor decided that #MEq *would* be reflected in the final rubric as a mistake.

3.3 Developed a Tree Rubric

We then built a decision tree to encode how different features of student responses (represented by tags) determine scores for these responses, see Figure 1. The grader drafted the tree and then worked with the instructor to associate point values to each leaf node. Alternative grading policies correspond to different decision trees. Comparing these trees helped resolve grading decisions efficiently. For example, we needed to decide if mistakes about equational reasoning and list definition justifications compound: would an assignment tagged #MEq and not tagged #LD score 4 points or 3 points? In other words, do we connect the F child of the ¬#MEq node to the AND node or the MAJ node? Formulating this grading policy question using trees helped us settle it in under 15 minutes.

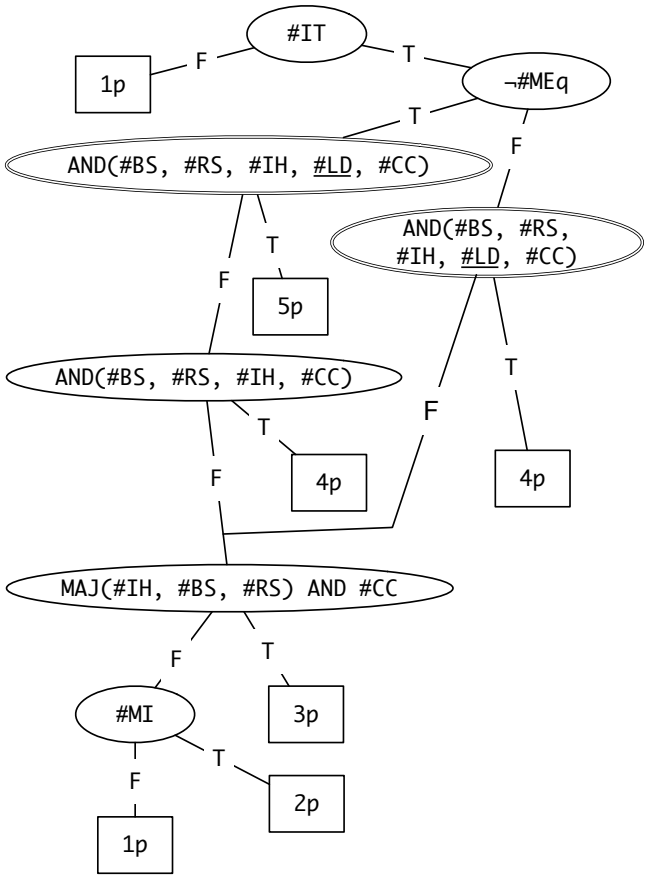

Figure 1: The decision tree from our case study. Labels for nodes are functions of the tags from Table 1 and Table 2. The two nodes with double-outline have the same decision label on separate paths. They differ from the node one level down on the tag #LD.

The grader transformed each justification for a high-frequency mistake tag into standardized comments that would be applied to student assignments. The distinction between justification and comment is important: the justification is an internal tool among course staff which may use jargon to precisely define relevant features; the comment is a student-facing message intended to help them learn from feedback. To quickly deploy comments during the grading process, we stored them in TextExpander [20], a program that automatically replaces typed abbreviations with pre-determined strings. We made an abbreviation for each (tag, justification, comment) tuple that expanded out into the appropriate comment. In this way, we were able to quickly annotate student work with long and consistent comments that were standadized based on the tags.

3.4 Applied the Tree Rubric

To actually assign scores, we randomly ordered all the student responses and visited each one in a final pass. For each response, we performed three steps:

(1) Assigned tags to the response. Even if the response was part of our initial sub-sample, we re-assigned tags using the justifications from the *end* of the Initial Tagging phase.
(2) Evaluated the Tree Rubric. Using the tags, we walked the decision tree to see which leaf the current response would fall into to determine the relevant student-facing rubric items.
(3) Applied the comments. We composed and applied long-form feedback by concatenating standard comments associated with each tag (and occasionally also custom prose).

The leaves in the decision tree map to student-facing rubric items and descriptions. This rubric is more detailed than the generic reference rubric for induction problems, and certain levels are split into separate items to highlight common mistakes. The number of submissions (and the percentage of total submissions this represents) for which each item was applied is listed below.

5 pts Correct: Indicates the statement is true. Proves it using structural induction with a clearly identified basis step and recursive step, with the correct IH. Uses clear and correct calculations and references to definitions in both steps, including using the IH, to conclude both. **(28 of 233; 12%)**
4 pts Partial Credit: Correct except for malformed equational reasoning. **(60 of 233; 26%)**
4 pts Partial credit: Indicates the statement is true. Proof has correct structure but mis-applies or doesn't reference key definitions of lists to justify basis and recursive steps. **(9 of 233; 4%)**
3 pts Partial credit: Indicates the statement is true. Proof has the right structure aside from missing one key component (only has one step rather than basis and recursive, missing IH, doesn't apply or mis-applies IH) that makes the argument incomplete or inconsistent. **(42 of 233; 18%)**
2 pts Partial credit: Indicates the statement is true, and applies a partially working proof that mentions induction. **(47 of 233; 20%)**
1 pt Partial credit: Indicates the statement is true, but only applies a witness or other inapplicable proof strategy **(36 of 233; 15%)**
1 pt Partial credit: Indicates the statement is false, and proceeds by (incorrect) counterexample. **(5 of 233; 2%)**
0 pts Incorrect or blank. **(8 of 233; 3%)**

An example application of comments appears in Figure 2. The two comments are associated with the tags for this student response: the comment "which part" comes from `#LD` and the comment about equations comes from `#MEq`. Notice that rubric item descriptions don't explicitly mention list definitions (the tag `#LD`). This is an example of how tags are never "wasted," even if they don't affect the final score; we can still map the relevant comments into student assignments and give them consistent and useful feedback.

Once all other student submissions were graded, the grader considered each response tagged `#???`, writing customized feedback and assigning a score. These idiosyncratic responses were easier to handle consistently now that the grader had worked through all the patterns of student responses.

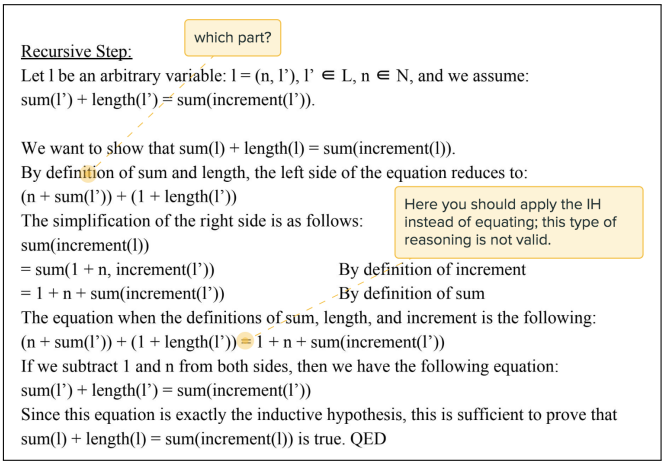

Figure 2: Sample student work with comments (in highlight boxeds) based on tags, as seen by students in Gradescope.

Before publishing the grades to students, the grader removed placeholder rubric items for tags in Gradescope so that only the final rubric was visible. The grader spent approximately 1.5 hours on the pre-grading phases of the process (generating initial tags, discussing with the instructor, and creating the tree rubric) and 6.5 hours applying the rubric and comments to student submissions.

3.5 Extracted Insights and Acted on Them

Our careful analysis of student responses is useful even if the students *never look at their grades*. For example, the tag `#ExRC` flagged the mistake where the correct recursive argument was performed on a two-element list instead of the full generality required in the recursive step. While `#ExRC` was both serious and widespread, it was correlated with other tags and so was pruned before the creation of the tree rubric. However, we spent time in class explaining why this type of "recursive" step is actually just an example, not a component of an inductive proof. Targeting the misconceptions about structural induction that led to this error helped reduce its appearance in follow-up assignments.

Even though difficult questions often generate many regrade requests, there were only 2 regrade requests for this question (out of 233 submissions). One request was a simple student misunderstanding. The other was grader error: the student response had a particularly egregious example of the `#ExRC` mistake and this caused a cascading failure: the grader, distracted by this error, set `#IH` to false even though the student had a correct inductive hypothesis. Responding to the regrade request was quick and easy: we simply reevaluated the tree rubric, setting `#IH` to true. Confidently addressing a small number of regrade requests helps restore any eroded trust students may feel from grading errors, and saves time.

For context, across the eight assignments in this class, the mean regrade count for ad-hoc graded problems with non-zero regrade requests was 2.3 (median = 2) and the mean number of regrade requests for adaptive graded problems was 2.2 (median = 2). Since

we picked the most complex problems in each assignment for adaptive grading, it would not be surprising if these problems generated higher than average regrade requests. Instead, the conditional regrade rate for the adaptive graded problems was very close to that of ad-hoc graded *simpler* problems.

4 ADAPTIVE RUBRIC CONSTRUCTION

The objective is to *consistently* justify grades using (i) readily observable features of student responses and (ii) course norms. The prerequisites are a reference generic rubric Q for the question type, a sample canonical solution, and a system for traversing student assignments and tracking tags, such as Gradescope. Summarizing and generalizing the case study, we get the method below.

(1) **Generate Initial Tags**
   (a) **Tag the Canonical Solution.** Grade the canonical solution according to the generic rubric Q. Ask *why* or *why not* each rubric item applies. Describe the answers with simple Boolean predicates, along with justifications *specific* to this assignment: these are the initial tags.
   (b) **Tag a Sub-Sample.** For each response r in a random subsample of student responses, perform three tasks:
      - *Assign Existing Tags.* Assign applicable tags from the existing collection to r. If necessary, add or edit a justification for each tag assigned.
      - *Identify Mistakes.* If r is not a "perfect" response, use the generic rubric Q and ask *why* or *why not* items from that rubric were applied; the answers define new tags. This mistake identification is dual to how the canonical solution induces tags that explain *correctness*.
      - *Concentrate Uncertainty:* Tag specific questions about the rubric (eg, `#?LDC`) and use a "catchall" tag (eg, `#???`) for responses that take too long to classify.
   (c) **Compute Population Statistics and Prune Tags.** Use a histogram of tag occurrence counts to identify common mistakes and solution techniques. Prune tags that don't carry enough information to help cluster responses.
(2) **Make Course-Value Judgments.** Reference course learning outcomes to answer accumulated questions (made concrete by the recorded justifications) in popularity order.
(3) **Develop a Tree Rubric.** Using tags as Boolean variables, write a decision tree[1] whose leaves are Q-realizations. Tags representing essential features or catastrophic mistakes should be placed early in the tree to reflect their value. Write a student-facing comment for each justification for each tag. Assign scores to the leaves of the decision tree.
(4) **Apply the Tree Rubric.** Clear all applied tags except `#???`. For each response (in random order) without this tag: (1) apply tags, (2) evaluate the tree rubric and assign the resulting score, and (3) mix together and apply the pre-written comments. Then, revisit assignments bearing the exceptional `#???` tag. The rich mental context from passing through the rest of the population will help you understand and provide sensible feedback for long tail work.
(5) **Extract and Act on Insights.** Repeat the population statistics analysis from the initial sub-sample. Use common patterns and mistakes to tailor future student interactions.

[1] Any classifier may be selected, but Decision Trees are a "sweet spot" of expressive power and interpretability. For simpler problems, additive models may be preferable.

5 BENEFITS AND LIMITATIONS

5.1 Structuring the grading workflow

The adaptive rubric process decouples fundamentally unrelated cognitive activities: pattern recognition and value judgment. In the sub-sampling and tagging phase, the grader observes subtle differences and patterns in student responses, while recording the statistical structure of the data so that this initial effort isn't wasted. By starting with exploratory data analysis to describe the student responses and deferring value judgments, graders are more likely to make progress on their grading tasks with fewer backtracking detours: these detours cost the grader in both time and cognitive load. Future work could use experimental methods to assess this process rigorously.

5.2 Synchronization barriers

The structure of the adaptive rubric grading process adds milestones during grading where decisions and communication are batched. These points are synchronization barriers where the grader and senior course staff must make decisions together. Requiring that these barriers are unlocked relatively early in the grading process means graders need to start well in advance of the grading deadline and avoid "marathon" grading. This may be a good thing.

Our process has a *single* grader work on each problem, communicating only with senior course staff. In addition to recording *why* tags are assigned (via justifications), this grader develops implicit procedural knowledge of *how* to assign tags. Future work could develop methods to synchronize tagging procedures across multiple graders; we could scale out the process while maintaining consistency at some cost in communication overhead.

5.3 Tool support for grading

In the case study, we mentioned three different software tools used to support the adaptive rubric process: Gradescope, Bear, and TextExpander. Other learning management systems, text editors, and text snippet managers may work well too. To our knowledge, there is no single tool that supports tagging with internal justifications and external comments and calculates population-level statistics.

6 CONCLUSIONS

In this experience report, we contributed a new framework for structured grading of open-ended problems, along with a case study of its use in an undergraduate CS course. Grading with adaptive rubrics lets us focus on deeply understanding student work in aggregate before deciding on point values. This approach yields insights on the population-level distribution of student mastery of the material. The meta-data produced during this process create an efficient shorthand among the course staff at the same time as helpful, explanatory feedback to students. Aligning assessment with learning outcomes, the adaptive rubric process extract evidence of student learning and allows the assessment process to inform the shape of the course itself.

ACKNOWLEDGMENTS

The authors thank the rest of the instructional team for the class in Spring 2019, specifically Joe Politz for helpful conversations. We also thank the anonymous reviewers, whose comments and questions improved the presentation and clarity of this manuscript.

REFERENCES

[1] 2019. Giving copious feedback on coding assignments in reasonable time. SIGCSE members mailing list, August 22, 2019.
[2] 2019. Python autograders. SIGCSE members mailing list, August 20, 2019.
[3] Heidi Goodrich Andrade. 2005. Teaching With Rubrics: The Good, the Bad, and the Ugly. *College Teaching* 53, 1 (2005), 27–31. https://doi.org/10.3200/CTCH.53.1.27-31
[4] Veronica Cateté, Erin Snider, and Tiffany Barnes. 2016. Developing a Rubric for a Creative CS Principles Lab. In *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '16)*. ACM, New York, NY, USA, 290–295. https://doi.org/10.1145/2899415.2899449
[5] Carol Evans. 2013. Making Sense of Assessment Feedback in Higher Education. *Review of Educational Research* 83, 1 (2013), 70–120. https://doi.org/10.3102/0034654312474350
[6] Sue Fitzgerald, Brian Hanks, Raymond Lister, Renee McCauley, and Laurie Murphy. 2013. What Are We Thinking when We Grade Programs?. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education (SIGCSE '13)*. ACM, New York, NY, USA, 471–476. https://doi.org/10.1145/2445196.2445339
[7] Gradescope. 2019. Gradescope. http://gradescope.com
[8] Matthew Inglis and Andrew Aberdein. 2014. Beauty Is Not Simplicity: An Analysis of Mathematicians' Proof Appraisals. *Philosophia Mathematica* 23, 1 (07 2014), 87–109. https://doi.org/10.1093/philmat/nku014
[9] Anders Jonsson and Gunilla Svingby. 2007. The use of scoring rubrics: Reliability, validity and educational consequences. *Educational Research Review* 2, 2 (2007), 130 – 144. https://doi.org/10.1016/j.edurev.2007.05.002
[10] Deepak Kumar. 2018. REFLECTIONS: Tools from the Education Industry. *ACM Inroads* 9, 3 (Aug. 2018), 22–24. https://doi.org/10.1145/3233246
[11] Anastasiya A. Lipnevich, Leigh N. McCallen, Katharine Pace Miles, and Jeffrey K. Smith. 2014. Mind the gap! Students' use of exemplars and detailed rubrics as formative assessment. *Instructional Science* 42, 4 (2014), 539–559. https://doi.org/10.1007/s11251-013-9299-9
[12] Emily Marshman, Ryan Sayer, Charles Henderson, Edit Yerushalmi, and Chandralekha Singh. 2018. The challenges of changing teaching assistants' grading practices: Requiring students to show evidence of understanding. *Canadian Journal of Physics* 96, 4 (2018), 420–437. https://doi.org/10.1139/cjp-2017-0030
[13] Robert C. Moore. 2016. Mathematics Professors' Evaluation of Students' Proofs: A Complex Teaching Practice. *International Journal of Research in Undergraduate Mathematics Education* 2, 2 (01 Jul 2016), 246–278. https://doi.org/10.1007/s40753-016-0029-y
[14] Ernesto Panadero and Anders Jonsson. 2013. The use of scoring rubrics for formative assessment purposes revisited: A review. *Educational Research Review* 9 (2013), 129 – 144. https://doi.org/10.1016/j.edurev.2013.01.002
[15] Cassandra Paul, Wendell Potter, and Brenda Weiss. 2014. Grading by Response Category: A simple method for providing students with meaningful feedback on exams in large courses. *The Physics Teacher* 52, 8 (2014), 485–488. https://doi.org/10.1119/1.4897587
[16] Heather L. Petcovic, Herb Fynewever, Charles Henderson, Jacinta M. Mutambuki, and Jeffrey A. Barney. 2013. Faculty Grading of Quantitative Problems: A Mismatch between Values and Practice. *Research in Science Education* 43, 2 (April 2013), 437–455. https://doi.org/10.1007/s11165-011-9268-8
[17] Eric Steven Raymond and Rick Moen. 2014. How to ask questions the smart way. http://www.catb.org/~esr/faqs/smart-questions.html
[18] Y. Malini Reddy and Heidi Andrade. 2010. A review of rubric use in higher education. *Assessment & Evaluation in Higher Education* 35, 4 (2010), 435–448. https://doi.org/10.1080/02602930902862859
[19] Shiny Frog. 2019. Bear. https://bear.app/
[20] Smile. 2019. TextExpander. https://textexpander.com/
[21] Martijn Stegeman, Erik Barendsen, and Sjaak Smetsers. 2016. Designing a Rubric for Feedback on Code Quality in Programming Courses. In *Proceedings of the 16th Koli Calling International Conference on Computing Education Research (Koli Calling '16)*. ACM, New York, NY, USA, 160–164. https://doi.org/10.1145/2999541.2999555
[22] Briana E. Crotwell Timmerman, Denise C. Strickland, Robert L. Johnson, and John R. Payne. 2011. Development of a 'universal' rubric for assessing undergraduates' scientific reasoning skills using scientific writing. *Assessment & Evaluation in Higher Education* 36, 5 (2011), 509–547. https://doi.org/10.1080/02602930903540991
[23] Todd Zimmerman. 2017. Grading for Understanding – Standards-Based Grading. *The Physics Teacher* 55, 1 (2017). https://doi.org/10.1119/1.4972500