Board 43: Designing Boosters and Recognition to Promote a Growth Mindset in Programming Activities

Prof. Stephen H Edwards, Virginia Tech

Stephen H. Edwards is a Professor and the Associate Department Head for Undergraduate Studies in the Department of Computer Science at Virginia Tech, where he has been teaching since 1996. He received his B.S. in electrical engineering from Caltech, and M.S. and Ph.D. degrees in computer and information science from The Ohio State University. His research interests include computer science education, software testing, software engineering, and programming languages. He is the project lead for Web-CAT, the most widely used open-source automated grading system in the world. Web-CAT is known for allowing instructors to grade students based on how well they test their own code. In addition, his research group has produced a number of other open-source tools used in classrooms at many other institutions. Currently, he is researching innovative for giving feedback to students as they work on assignments to provide a more welcoming experience for students, recognizing the effort they put in and the accomplishments they make as they work on solutions, rather than simply looking at whether the student has finished what is required. The goals of his research are to strengthen growth mindset beliefs while encouraging deliberate practice, self-checking, and skill improvement as students work.

Mr. Zhiyi Li, Virginia Tech/Department of Computer Science

I am a Ph.D. graduate student in Department of Computer Science in Virginia Tech since Fall, 2013. My research interests is computer science education. Before that, I worked as a research staff in School of Medicine in University of Virginia from 2007 to 2013. I hold a Master degree in Computer Science in Virginia Tech. Master degree in Computer Science and Chemistry in Georgia State University in Atlanta, GA. I obtained my Bachelor degree of Engineering in East China University of Science and Technology in Shanghai, China.

Designing Boosters and Recognition to Promote a Growth Mindset in Programming Activities

April 29, 2019

Abstract

When one first learns to program, feedback on early assignments can easily induce a fixed mindset—where one believes programming is a fixed ability you either have or you don't. However, possessing a fixed mindset perspective has negative consequences for learning. The alternative is to foster a growth mindset, where one believes ability can be improved through practice, effort, and hard work. However, automated grading tools used on programming assignments currently focus on objectively assessing functional correctness and other performance-oriented features of students' programs. Unfortunately this encourages students to adopt performance-oriented goals, which are characteristic of a fixed mindset. By building on existing measures of "productive effort", we design a new kind of feedback approach that focuses on recognizing, encouraging, and rewarding diligence and productive actions based on those indicators. The goal is to add such elements to existing feedback in an emotionally supportive way that recognizes the efforts of a student expending and valuing these practices. The feedback design presented here consists of two main components: textual/verbal feedback that recognizes productive effort students spend on a problem, or that encourages students to be strategic about expending effort to improve their own skills. The point of this feedback is to convey to the student that constructive practice to improve one's skills is valued and recognized, independently of the final product they are creating. In addition to the textual feedback, the feedback also includes boosters, or rewards in the form of perks that enhance parts of the student work experience. By taking inspiration from video game psychology and other sources, we designed a booster-based reward system that recognizes hard work without tacitly promoting performance-oriented (score-oriented) motivation. In addition to describing the design of the reward and recognition feedback strategy and the variable ratio reinforcement schedule on which the strategy is based, we also present a post hoc analysis of the results obtained when applying this strategy to existing student submission data. This allows investigating what feedback or boosters would have been earned by individual students in a real-life situation to validate the feedback design before live deployment.

Keywords

Rewards, incentives, motivation, intrinsic motivation, programming assignments, grading, automated grading, feedback.

Introduction

Automatic programming assessment systems such as Web-CAT[5 6 7] are widely used in courses where students learn to program. Providing rapid feedback on solutions gives students a more concrete understanding of whether their program works, and allows for a larger number of corrections and re-submissions, leading to greater opportunities for learning[8]. The feedback produced by such automated graders is often performance-based, however, focusing on evaluating program correctness and other measures of solution quality, without regard for the amount of students' effort in trying to solve the problem. Unfortunately, this feedback does nothing to encourage a *growth mindset*, and may instead implicitly reinforce a *fixed mindset*.

Students with a *fixed mindset* believe their intelligence is a predetermined asset that they are "born with"[4]. In the context of programming, this is sometimes known as the "geek gene", which is hypothesized as: that programming ability is an innate talent, rather than a learned skill[14 12]. However, students with such beliefs are more likely to avoid academic challenges, perform more poorly, and give up sooner than students with a *growth mindset*. Students with a *growth mindset* believe that they can increase their abilities through hard work and effort[4]. Individuals with a fixed mindset also focus more on performance-oriented goals such as grades or assignment scores, because they equate academic achievement with an indication of their intelligence or ability. This is in contrast to growth mindset individuals, who are more likely to adopt learning goals that challenge them to increase their own abilities.

This paper reports on research to augment existing automated grading feedback so that it encourages and reinforces a growth mindset view by explicitly recognizing the productive effort that students put into their solutions as they develop them. This recognition is an add-on in addition to the objective performance-based scoring that an automated grading system already performs. However, rather than using marks (or points) to direct student attention, the aim is to foster growth mindset beliefs through non-points-based feedback, in university and college level introductory programming courses CS1 or CS2. This work may be extended to apply AP programming courses in K-12 education in future.

The feedback design presented here consists of two main components: textual/verbal feedback that *recognizes* productive effort students spend on a problem, or that *encourages* students to be strategic about expending effort to improve their own skills. The point of this feedback is to convey to the student that constructive practice to improve one's skills is valued and recognized, independently of the final product they are creating. In addition to the textual feedback, the feedback also includes *boosters*, or rewards in the form of perks that enhance parts of the students' work experience. By taking inspiration from video game psychology and other sources, we designed a booster-based reward system that recognizes hard work without tacitly promoting

performance-oriented (score-oriented) motivation. In addition to describing the design of the reward and recognition feedback strategy and the variable ratio reinforcement schedule on which the strategy is based, we also present a post hoc analysis of the results obtained when applying this strategy to existing student submission data. This allows us to investigate what feedback or boosters would have been earned by individual students in a real-life situation to validate the feedback design before live deployment.

Related Work

Rewards, Operant Conditioning, and Intrinsic Motivation

Applying rewards to increase student motivation has been widely researched in areas including psychology, education, and video games[1,2,17]. In 1940's, Skinner formed the theory of *operant conditioning* to explain how a behavior can be shaped by its resulting consequences[15]. By studying the behavior of animals in experiments, Skinner formulated the underlying ideas of positive and negative reinforcement through rewards and punishments. *Operant conditioning* theory is often applied in students' learning process in classroom. For example, a teacher can encourage students to answer questions by positive reinforcements such as praise if they answer correctly. Even though *operant conditioning* has been applied in learning process, it is limited because it does not include other factors such as inherited and cognition factors and cannot explain the human learning process completely. Recent research work studies the role of cognition in classical and operant learning[10].

While *operant conditioning* has been shown to affect behavior in some circumstances, in the classroom teachers are also concerned with motivation. Human motivation often is classified into two categories: intrinsic motivation and extrinsic motivation. When people engage in an activity, intrinsic motivation is internal and based on the individual's own interests. For example, a student who engages in programming because he really likes the activity is intrinsically motivated. Extrinsic motivation, instead, is caused externally by a reward or by avoidance of negative effects such as punishments[1]. For example, a student who attends a programming contest because there is an opportunity to win a scholarship is extrinsically motivated.

Extrinsic rewards such as money, grades, and praise definitely can influence individual's intrinsic motivation. Rewards can be categorized as intangible (e.g., verbal praise, social approval) or tangible (e.g., money, grade). Rewards can also be categorized based on expectancies: expected rewards, unexpected rewards, and no rewards. There are controversial opinions about the effect of extrinsic rewards on intrinsic motivation in the education research community. Judy Cameron et al.[1] review the research about the effect of reinforcement/rewards on intrinsic motivation. Their meta-analysis of 96 studies indicates overall, rewards do not decrease intrinsic motivation. However, Deci et al.[3] claimed rewards undermine a student's intrinsic motivation instead. Dweck[4] has stated that changing a student's mindset requires intrinsic motivation, and therefore extrinsic rewards may not be the best approach. In this work, we present a design that attempts to balance these differing concerns by avoiding mindset-oriented feedback aimed at performance goals or points-based/grade-based rewards. Instead, we focus on reinforcing the idea that course

staff *value* productive effort and practice as students work on solutions, and that students should also value this effort because it leads to improvement.

Rewards in Video Games

Reward strategies are widely used in video games. David explains a number of game reward systems[11]. He introduces a key reinforcement schedule used in many video games: variable-ratio reinforcement. The idea of a variable-ratio reinforcement schedule comes originally from Skinner's pigeon experiments in his *operant conditioning* theory work, where pigeons were more likely to press a lever when they received a food reward only 50% of the time, compared to when they receive the reward every time. A variable-ratio reinforcement schedule is applied in many video game designs because it produces a better response rate than fixed schedules.

Wang et al.[19] gave an overview about how reward systems give positive experiences to players in various video games. They recommend that players can have fun with both rewards and reward mechanisms. Rewards and reward mechanisms in video games foster players' intrinsic motivation with sense of fun and pleasure. Some games may also give extrinsic rewards such as virtual rewards. Wang et al.[19] also describe reward formats in video games. One interesting reward form is feedback messages used as an instant reward. These instant messages can take many forms, including textual/verbal praise, pictures, sound effects, and video clips. Another interesting reward format is story-based animations and pictures that further the plot. This reward is often applied when important events happen such as beating major enemies, advancing to new levels, and finishing the game. In video games, virtual rewards are often applied to enhance players' motivation. Some virtual rewards aim to power up players' characters such as experience points, in-game money, or special character equipment. Others use story content rewards such as cut-scenes or audio logs. Lessons from video games are important, since the industry is based on encouraging players to return over and over to a game to continue playing, and on developing the player's intrinsic motivation to foster a long-lasting play experience.

Sylvester illustrated the relationship of *dopamine* with pleasure and motivation in video games[18]. Dopamine is a brain chemical generated from the brain's reward center based on outside stimuli such as a bit of tasty food, winning money or a look at an attractive mate. In theory of motivation, dopamine is pleasure's messenger. Dopamine is a marker of motivation since motivation is to pursue for pleasure. In video games, both real rewards and virtual rewards are all dopamine driven rewards. An interesting thing is why virtual rewards work because players cannot distinguish virtual rewards from real rewards since the human brain evolved in an environment that lacked modern games[18].

Progress Indicators

The work described here depends critically on being able to measure a student's productive effort. To this end, we build on existing research that has produced a series of 15 indicators that are designed to be applied incrementally to a series of program submissions by a student. Together,

these indicators identify when students are investing productive effort on their solution, as it is being developed[9].

These indicators use the same strategy described by O'Rourke et al., who created a math education game to teach children fraction concepts while also attempting to encourage adoption of a growth mindset[16]. In order to measure effort, they developed a group of simple indicators that individual might suggest a student was productively working on a solution, and developed a (textual) feedback scheme that used the indicators in combination. These fifteen indicators compare features of a program in the current submission with the previous four submissions. For each indicator, only if parameters show an increasing (positive) trend in the current submission, that indicator is treated as positive (triggered). These fifteen indicators include seven general purpose indicators and eight software testing indicators. Seven general purpose indicators are about various perspectives when students construct solutions for their assignments. Eight software testing indicators concentrate on student's testing effort when they are writing better software tests. Among 15 indicators some individual indicators may highly related with other indicators. We applied Pearson Correlation Coefficient (PCC) function in R packages to find highly related indicators, with a historical CS2 programming dataset. 3 Redundant indicators were removed and left 12 indicators were used for later research work [13]. All 12 are then combined using a simple rule, where a student is considered as productively working on their solution if at least K indicators are simultaneously triggered. Here, we adopt this existing work and use a threshold of $K = 4$ to determine if a student is working productively, with the aim of giving students instant feedback to encourage adoption of a growth mindset. The threshold of K is determined by statistic analysis of a historical CS2 programming dataset[7].

Submission Energy

While not directly related to the concept of rewards or mindset adoption, one additional piece of related work is relevant here: the idea of assignment *submission energy*. This concept was inspired by techniques from commercially successful mobile games. These games use structural features that limit the player's play time, and encourage more frequent but shorter play sessions. This usually takes the form of a limited resource that is regenerated over time (lives, energy, gold, fuel, ammunition, etc.), where the resource is required for some aspect of play, and the player then has the responsibility for managing the resource. The idea of a limited resource that regenerates over time but limits the player's actions shapes the way players manage their play time: it deters "binge" play sessions and strongly promotes the use of a much larger number of small, periodic play sessions spread over a longer period of time.

We already use a similar mechanism that we call *submission energy* aimed at achieving the same goals. For each assignment, each student has an *energy bar* that holds up to 3 units of submission energy. Each time the student submits to the automated grading system, one unit of energy is used. The bar will then regenerate energy at a rate of 1 unit per hour until it is full again. If a student's energy runs out, the student cannot submit again until another unit of energy is regenerated. To ensure that lack of energy will not prevent a student from submitting their final work, students are still allowed to submit without energy during the last hour before an assignment deadline.

This energy system is intended to exert a similar effect on how students manage their time in completing assignments. The goal is to shift students to using a larger number of shorter work sessions, spread over a longer period of time. While we are still evaluating the effectiveness of this approach, it also presents some opportunities for certain kinds of rewards or perks given to recognize student achievements in terms of effort expended.

Design

To reinforce growth-mindset beliefs, we have designed a feedback strategy that can be layered on top of an existing automated grading system's existing feedback to augment it with an assessment of productive effort invested by students. The mindset-oriented feedback that will be delivered takes two forms: recognition and encouragement messages that reinforce the value of improving skills through hard work, and boosters (rewards) that provide more tangible process-oriented perks to students.

The design is based on the following principles:

- Both additional forms of feedback must be encountered frequently enough by students so that students see that their hard work is validated and recognized, even if they do not win boosters.
- Boosters in particular cannot be so rare they appear unattainable.
- Alternatively, feedback frequency cannot be so high that students become over-saturated, which will devalue the feedback and/or boosters.
- The reinforcement schedule chosen should promote sustained (rather than decaying) responses from students to the best extent possible.

Recognition and Encouragement

The first component of the feedback design takes the form of textual feedback that is intended to convey praise, social approval, encouragement, and reinforcement of growth-mindset beliefs. The primary purpose of this textual feedback is to convey to the student that productive effort is valued and helps improve one's coding skills.

Fortunately, Edwards et al.[9] suggest a direct way to generate such feedback from any individual indicator from the set of 12 used for assessing student effort. Each indicator captures a specific kind of change to the student's solution: adding code, adding software tests, reducing logic complexity, reducing the average length of methods, correcting formatting errors, and so on. Every indicator can thus be used to generate a feedback statement.

If an individual indicator is triggered on a student submission, it can be used to generate a *recognition* statement that is phrased to recognize the positive achievement a student has made through their efforts. Examples include:

You've clearly worked at extending your solution. As you write more code, your skills improve.

Your effort at increasing the thoroughness of your testing has paid off, which will help you find problems via self-checking.

By working hard, you have improved your solution to meet more of the assignment's expectations.

Similarly, if an individual indicator is *not* triggered on a student submission, it instead can be used to generate an *encouragement* statement that is phrased to encourage the student to work more productively. Examples include:

Consider shortening your methods, which improves your skills at writing understandable code.

Adding more software tests is a good way to self-check your own work and helps to build your code understanding skills.

To strengthen your skills, you can work on removing formatting problems from your code.

In principle, it is possible to generate feedback from every indicator on every submission, producing either a recognition message or an encouragement message for each one. Using a small collection of similarly phrased feedback statements for each indicator can add some variety. However, this approach would be overwhelming to students and would quickly lead to such feedback messages being ignored because they are so common.

Instead, our design makes two choices to reduce feedback to a level intended to meet our design principles. First, we only give feedback on one indicator per submission. Currently, we pick that indicator at random, alternating between a randomly selected indicator that was not triggered (an encouragement message) and then a randomly selected indicator that was triggered (a recognition message).

Second, we employ a *variable-interval* schedule of reinforcement. After one recognition or encouragement message is generated, we randomly pick a time between 30–90 minutes, and do not generate another until that time has elapsed. By experimentally validating against existing student data (see next section), we have found that the median time between submissions for students is 9 minutes, with approximately three quarters of all submissions being made within a half hour of the previous submission. Choosing this time range ensures that students are likely to receive either an encouragement or recognition message in each distinct work session, and will see a couple of messages in a long work session, but they typically will not see messages on many successive submissions on the same assignment.

Because these feedback messages are intended to reinforce growth-mindset ideas and are not

intended as rewards that students work towards, a variable-interval schedule is the best match for ensuring that all students see some feedback, but no individual students get over-saturated by too many messages. Rarity makes the messages stand out and appear more visible, since they only occasionally appear.

Boosters

Boosters are intended as tangible rewards for achievements, but are substantively different than marks or points used for grading because the goal is not to accidentally reinforce performance-based motivation (which is more commonly associated with fixed-mindset beliefs). Boosters are intended to be won only on submissions where students have achieved an appropriate level of productive effort—that is, where at least $K = 4$ of the 12 indicators have triggered on the same submission. While the individual indicators themselves are "noisy" and do not always indicate productive effort, using them in combination decreases (or "averages out") noise and makes it much more difficult for students to "game the system" by trying to focus on maximizing individual indicators.

We have grouped the boosters we have devised into two tiers on increasing value:

- Tier 1 (50% of boosters):
  - *Energy +1*: One unit of submission energy instantly added to energy bar
- Tier 2 (25% of boosters):
  - *Energy refill*: submission energy bar completely refilled
- Tier 3 (15% of boosters):
  - *No waiting*: Jump to the front of the line one time when visiting instructor or teaching assistant office hours
  - *Supercharge*: Energy regenerates at 2X the normal rate for 24 hours
- Tier 4 (5% of boosters):
  - *Expert bug finder*: An instructor will find/explain a single bug in the student's code for the current assignment (the student can pick which bug/problem/issue they wish to receive help on)
  - *Full view*: see *all* of the diagnostics about program behavior generated from instructor-written reference tests on the current submission (normally hidden for our students, to prevent them from using the system to do their own testing)
  - *Late pass*: receive a penalty-free one-day extension on the current assignment (limit one per assignment)
  - *Energy-free 24*: unlimited submissions with no energy consumption for 24 hours

- *Quiz drop*: Drop the lowest quiz grade (limit 1 per semester)
- *Quiz drop*: Drop the lowest lab grade (limit 1 per semester)

Many of the boosters shown here add small modifications to the submission energy rules to allow students greater access to submissions. As in many video games, these allow the student to immediately see benefits that are directly observable in their work actions, but which affect their work process rather than their assignment score. In other classroom situations, different boosters might be chosen that better match the instructional needs and procedural methods used in that situation.

In addition to the range of boosters available and their relative likelihoods, this feedback design uses a variable-ratio reinforcement schedule. That is, on each submission that triggers the minimum number of indicators, there is a 50% chance that a student will actually win a booster. In cases where no booster is won, we fall back to the recognition/encouragement message mechanism.

Evaluating the Reward Strategy

To evaluate the feasibility of this feedback design, we applied it to a historical dataset of CS2 student programs including 20,364 submission attempts by 257 students over two semesters. This included 984 finished programs produced by the students across four assignments each semester.

In this population, the median number of submissions by each student was 16 per assignment (mean = 20.7, s.d. = 16.6). Of these, a median of 5 submissions triggered at least $K = 4$ indicators (mean = 5.37, s.d. = 3.93). These factors were significant influences on the 50% rate chosen for the variable-ratio schedule for boosters. With an average of 5 submissions per assignment demonstrating productive effort, we should expect 2-3 submissions to actually win boosters, with 1-2 being tier 1 boosters, and one being from a higher tier.

To confirm the feedback rates are consistent with our design principles we applied this feedback design to the historic dataset. The results are summarized in Figure 1. With the feedback design presented here, only 38% of students failed to receive any booster across all their submissions on an assignment. Instead textual recognition or encouragement feedback according to the variable-interval schedule were sent to these students. This ensures that even students not earning boosters still periodically see reinforcing messages about working productively to improve their skills (but not too frequently). This confirms the design achieves our first design principle, ensuring the feedback was visible to students.

Figure 1 reflects the proportion of students who won different numbers of boosters across all of their submissions on a single assignment. As mentioned, 38% of students earned no boosters, with 41% of students win a single tier1 booster(first major) across all of their submissions, and 24% earning tier 2 boosters(second major). The stacked bars in Figure 1 indicate what proportion of students earned a booster from each two specific tier. For example, about 24% of students

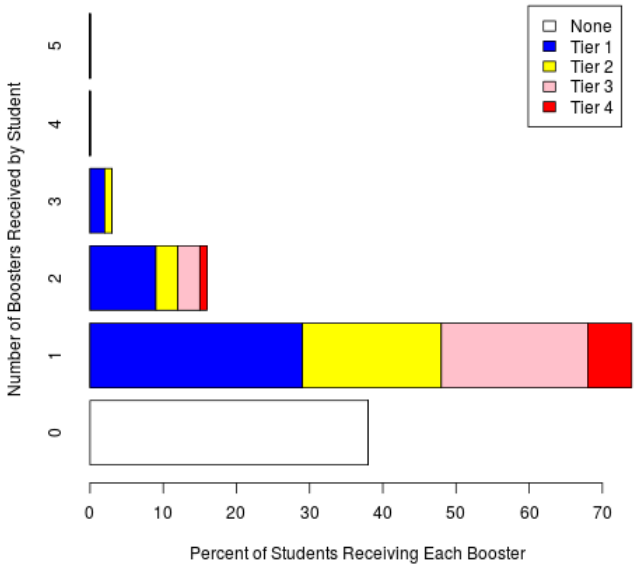

Figure 1: Simulated rewards by student.

earned one booster over all their submissions on an assignment and that booster turned out to be from Tier 1, while 4% of students earning a single booster received a Tier 2 booster.

Overall, 38% of students earned 0 boosters, with 63% (major) earning Tier 1 and Tier 2 on their assignment submissions. Three things are clear from Figure 1. First, boosters are earned often enough so that students will see Tier 1 boosters regularly, and will at least be aware of peers earning higher-tier boosters as well. Second, it is clear that the highest-tier boosters are exceedingly rare. Third, even though some students made many fewer submissions and some students made many more, the bulk of the students only received a small handful of boosters. The structure of the indicators and the use of a minimum number of simultaneously triggered indicators significantly reduces the relationship between the total number of submissions and the subset that qualify for a booster so the relationship is no longer linear. While the small number of students who earned a large number of boosters tended to submit an excessive number of times, their rate of earnings was much lower than typical.

Conclusions and Future Work

In this paper, we present a novel strategy to augment traditional automated feedback with new mechanisms intended to promote growth mindset beliefs by recognizing and encouraging productive effort as students work instead of relying solely on points-based grading for motivation. By taking inspiration from reward mechanisms intended to increase intrinsic motivation, including those found in modern video games and mobile games, we provide a design that combines both textual feedback and boosters (or more tangible rewards). The textual

feedback is intended to reinforce the belief that productive effort helps us improve our skills at coding, and aims to convey that course staff (and the grading system) value this effort independently of the final product. At the same time, by careful use of intermittent schedule of reinforcement, we can balance the need to ensure feedback occurs sufficiently often that students see it, but not so often that it becomes commonplace and devalued.

In addition to textual feedback that reiterates the value of practice for skill improvement, we also include boosters in our design. These are earned more rarely by students. To avoid any preoccupation with performance-oriented (or grade-oriented) extrinsic motivation that may be associated with fixed-mindset beliefs, we instead selected boosters that offer observable "perks" to the student's workflow and process of solution creation without directly affecting the program grade. Indeed, even selecting the term "booster" is an intentional choice aimed at avoiding the connotations of "rewards" or "prizes", and instead conveying the idea that these are benefits that "boost" the way a student operates in completing their assignments rather than trophies. By tiering boosters, we can make students aware of more desirable (and hence, more motivating) boosters, even if those boosters may be relatively rare in practice. Instead, by ensuring students earn a minimum level of lower tier boosters when they make useful progress, we can ensure they are aware of the mechanism and experience it directly, but keep the more desirable boosters far enough out of reach that students are encouraged to consider their development actions to increase their chances of earning.

By applying the feedback design described in this paper to historical data in a post-hoc study, we have shown that students do earn boosters at the designed rates. This is important to meet our design objectives laid out in an earlier section. The use of a variable-ratio reinforcement schedule has been shown to provide effective behavioral incentives for sustained change, and is the best fit for the booster model presented here.

The next steps for this work are to deploy the feedback model in a live classroom to evaluate its impact on student behaviors and on student mindset beliefs. Before going "live" with such a scheme, it is important to verify that its design achieves the objectives that were intended, or any live study would be suspect. Here, we have provided the description of the design and the evidence that it operates as intended in order to lay the groundwork for a live evaluation. While some educators may imagine different boosters they believe are more effective, or may be unable to use some boosters described in this design because of the particulars of the automated grading system used, we believe that the approach embodied in this design will help provide a basis for others to create their own alternative reward schemes based on the same underlying principles.

Acknowledgments

This work is supported in part by the National Science Foundation under grant DUE-1625425. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Judy Cameron and W. David Pierce. 1994. Reinforcement, Reward, and Intrinsic Motivation: A Meta-Analysis. *Review of Educational Research* 64, 3 (1994), 363–423. http://www.jstor.org/stable/1170677

[2] Edward L. Deci, Richard Koestner, and Richard M. Ryan. 1999. A Meta-Analytic Review of Experiments Examining the Effects of Extrinsic Rewards on Intrinsic Motivation. *Psychological Bulletin* 125, 6 (1999), 627 – 668. DOI: http://dx.doi.org/10.1037/0033-2909.125.6.627

[3] Edward L. Deci and Richard M. Ryan. 2001. Extrinsic Rewards and Intrinsic Motivation in Education: Reconsidered Once Again. *Review of Educational Research* 1 (2001), 1–27.

[4] Carol S. Dweck. 2000. *Self-theories: Their role in motivation, personality, and development*. Psychology Press.

[5] Stephen H. Edwards. 2003. Improving Student Performance by Evaluating How Well Students Test Their Own Programs. *J. Educ. Resour. Comput.* 3, 3, Article 1 (Sept. 2003). DOI: http://dx.doi.org/10.1145/1029994.1029995

[6] Stephen H. Edwards. 2003. Rethinking Computer Science Education from a Test-first Perspective. In *Companion of the 18th Annual ACM SIGPLAN Conference on Object-oriented Programming, Systems, Languages, and Applications (OOPSLA '03)*. ACM, New York, NY, USA, 148–155. DOI: http://dx.doi.org/10.1145/949344.949390

[7] Stephen H. Edwards. 2003. Using Test-Driven Development in the Classroom: Providing Students with Automatic, Concrete Feedback on Performance. *Proceedings of the International Conference on Education and Information Systems: Technologies and Applications, International Institute of Informatics and Systemics, 2003* (2003), 421–426.

[8] Stephen H. Edwards. 2004. Using Software Testing to Move Students from Trial-and-error to Reflection-in-action. In *Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education (SIGCSE '04)*. ACM, New York, NY, USA, 26–30. DOI: http://dx.doi.org/10.1145/971300.971312

[9] Stephen H. Edwards and Zhiyi Li. 2016. Towards Progress Indicators for Measuring Student Programming Effort During Solution Development. In *Proceedings of the 16th Koli Calling International Conference on Computing Education Research (Koli Calling '16)*. ACM, New York, NY, USA, 31–40. DOI: http://dx.doi.org/10.1145/2999541.2999561

[10] Irving Kirsch, Steven Jay Lynn, Michael Virgorito, and Ralph R. Miller. 2004. The Role of Cognition on Classical and Operant Conditioning. *Journal of Clinical Psychology* 60 (2004), 369–392. Issue 4. DOI: http://dx.doi.org/10.1002/jclp.10251

[11] David L. 2016. Game Reward Systems. *Learning Theories* (2016). https://www.learning-theories.com/game-reward-systems.html

[12] Clayton Lewis. 2007. Attitudes and Beliefs About Computer Science Among Students and Faculty. *SIGCSE Bull.* 39, 2 (June 2007), 37–41. DOI: http://dx.doi.org/10.1145/1272848.1272880

[13] Zhiyi Li and Stephen H Edwards. 2018. Applying Recent-Performance Factors Analysis to Explore Student Effort Invested in Programming Assignments. In *The 14th Int'l Conf on Frontiers in Education: Computer Science and Computer Engineering, FECS 2018*. Las Vegas, Nevada, USA, 3–10.

[14] Robert McCartney, Jonas Boustedt, Anna Eckerdal, Kate Sanders, and Carol Zander. 2017. Folk Pedagogy and the Geek Gene: Geekiness Quotient. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education (SIGCSE '17)*. ACM, New York, NY, USA, 405–410. DOI: http://dx.doi.org/10.1145/3017680.3017746

[15] Saul Mcleod. 2015. Skinner Operant Conditioning. https://simplypsychology.org/operant-(2015). [Online; accessed 28-December-2017].

[16] Eleanor O'Rourke, Kyla Haimovitz, Christy Ballweber, Carol Dweck, and Zoran Popović. 2014. Brain Points: A Growth Mindset Incentive Structure Boosts Persistence in an Educational Game. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 3339–3348. DOI: http://dx.doi.org/10.1145/2556288.2557157

[17] W.E. Scott. 1976. The effects of extrinsic rewards on "intrinsic motivation": A critique. *Organizational Behavior and Human Performance* 15, 1 (1976), 117–129. DOI: http://dx.doi.org/https://doi.org/10.1016/0030-5073(76)90032-5

[18] Tynan Sylvester. 2013. *Designing Games: A Guide to Engineering Experiences*. O'Relly Media, Inc.

[19] Hao Wang and Chuen-Tsai Sun. 2011. Game Reward Systems: Gaming Experiences and Social Meanings. In *Proceedings of DiGRA 2011 Conference: Think Design Play*. http://www.digra.org/dl/db/11310.20247.pdf