ABSTRACT

ZHEWEI, HU. Scaffolding the Grading Process for Peer Review Assignments: A Pluggable Reputation Web-Service Approach. (Under the direction of Dr. Edward F. Gehringer).

Peer review is commonly used in higher education and in MOOCs. If taken seriously, peer-review offers reviewers a chance to learn from their peers and improve their understanding of the assignment requirements. Educational peer-review can also help teaching staff decide on grades for student work. In order to generate accurate grades from peer-review, we need a way to decide which peer reviewers are reliable. A reputation system is one solution to this problem. A reputation system can quantify the reliability of each peer reviewer by comparing one’s peer-review grades with others’ or with expert grades. Many different reputation systems have been proposed by researchers.

However, having each peer-review system implement a large number of different reputation algorithms is time consuming, or even unrealistic. Although some peer-review systems have already deployed reputation algorithms, no individual system implements more than one. Hence it is quite difficult to compare the performance of different algorithms and to try to select the best one. A reputation web service can be a good approach to solving this problem. Firstly, different reputation algorithms are implemented on the server side. The web service is “pluggable” by client systems, so that they can test these algorithms at any time. The reputation web service is also language and platform-independent. No matter which development environment is used, the client systems only need to wrap the raw data into a standard transmission format and send it to the web service. Secondly, since different reputation algorithms have hardly been compared with each other before, it is possible to use the reputation web service to compare their performance.

In this thesis I present my work on building this reputation web service. Five reputation algorithms have been implemented. I have “plugged” the reputation web service into an existing peer-review system, Expertiza. Five experiments are designed to figure out how different initial reputation values and assignment types influence the accuracy of reputation
algorithms, and to compare the performance of different reputation algorithms acting on the same peer-review records.
Scaffolding the Grading Process for Peer Review Assignments: A Pluggable Reputation Web-Service Approach

by
Zhewei Hu

A thesis submitted to the Graduate Faculty of North Carolina State University in partial fulfillment of the requirements for the degree of Master of Science

Computer Science

Raleigh, North Carolina

2016

APPROVED BY:

Dr. Edward F. Gehringer
Committee Chair

Dr. Min Chi

Dr. Sarah Heckman
DEDICATION

To my parents and all other family members.
BIOGRAPHY

Zhewei Hu was born on March 3, 1992 in Cixi, Zhejiang, China. He attended the Zhejiang Normal University from 2010 to 2014, graduating with a bachelor’s degree in Educational Technology in 2014. He worked as Computer Teacher Intern at Zhejiang Zhenhai High School from March 2014 to May 2014. He joined the graduate program in Department of Computer Science at North Carolina State University in Fall 2014. He has been working with Dr. Gehringer’s research team since November 2014. He worked as research assistant for Dr. Gehringer from May 2015 to August 2015. In the meanwhile, he published 2 papers as the second author. They are “Pluggable reputation systems for peer review: A web-service approach” and “Closing the Circle: Use of Students’ Responses for Peer-Assessment Rubric Improvement.” Then he worked as a teaching assistant for CSC 517 in Fall 2015. This period provided him with lots of challenges and opportunities that helped to shape his future research work. He has already been admitted to the PhD program in the Computer Science department of North Carolina State University. He will work as a Software Engineer Intern at Offers.com at Austin, Texas from May 2016 to August 2016. With the defense of this thesis, he is receiving the degree of Master of Science in Computer Science from North Carolina State University.
ACKNOWLEDGMENTS

I would like to extend my thanks to the following people without whom it would not have been possible for me to achieve this work.

I sincerely thank my parents, for their dedication towards my future and their tireless efforts to expose me to the best of opportunities. I thank them for all the sacrifices they have made to provide me with everything I have ever needed. I also thank to all other family members. Without your help, I cannot complete this work.

I owe my sincere thanks to Dr. Edward Gehringer, my advisor, for his all-time support and guidance. I am very grateful to him for his valuable time, continuous advice and patient feedback. I learned a lot from our weekly meetings and his insights that helped me to generate creative ideas in my mind.

I would like to thank my other committee members, Dr. Min Chi, for helping to write recommendation letter for my PhD application, and her data mining course and machine learning course are excellent and will definitely help my future research. Dr. Sarah Heckman, for being my committee and providing her valuable feedback about the work.

I thank Yang Song, my good teacher and helpful friend, for recommending me to join Dr. Gehringer’s research team, guiding and cooperating with me to finish 2 papers, and going fishing with me.

Finally, I thank all other friends for their encouragement and supports at every moment.
# TABLE OF CONTENTS

LIST OF TABLES .................................................................................................................. vii  
LIST OF FIGURES .................................................................................................................. viii  
Chapter 1  Introduction ............................................................................................................. 1  
Chapter 2  Related Work ........................................................................................................ 3  
  2.1  Validity, Reliability and Spread ......................................................................................... 3  
  2.2  Formative Feedback and Summative Feedback ................................................................. 3  
  2.3  Calibration ....................................................................................................................... 4  
  2.4  Reputation ....................................................................................................................... 6  
Chapter 3  Reputation Systems ................................................................................................ 7  
  3.1  Reputation Algorithms ...................................................................................................... 7  
    3.1.1  Hamer’s Algorithm ..................................................................................................... 7  
    3.1.2  Hamer-expert Algorithm ......................................................................................... 10  
    3.1.3  Lauw’s Algorithm .................................................................................................... 10  
    3.1.4  Lauw-expert Algorithm .......................................................................................... 13  
    3.1.5  Quiz-based Algorithm ............................................................................................. 13  
  3.2  Suggestions for Improving the Accuracy of Reputation Algorithms ................................... 17  
Chapter 4  Reputation Web Service .......................................................................................... 19  
  4.1  Why Web Service ............................................................................................................ 19  
  4.2  Web Service Structure .................................................................................................... 20  
  4.3  Peer-Review Markup Language ...................................................................................... 20  
  4.4  Server Side Design ......................................................................................................... 23  
  4.5  Client Side Design ......................................................................................................... 23  
  4.6  Security of Web Service ................................................................................................. 27  
Chapter 5  Experiments ........................................................................................................... 30  
  5.1  Experiment 1 – How to calculate the aggregated grades based on reputation values? ........ 30  
    5.1.1  Design ................................................................................................................... 30  
    5.1.2  Result analysis ........................................................................................................ 31  
  5.2  Experiment 2 – How much do reputations change by rounds within one assignment? ........ 32  
    5.2.1  Design ................................................................................................................... 32
5.2.2 Results analysis ................................................................. 33
5.2.3 Conclusion ........................................................................... 37
5.3 Experiment 3 – Can reputation be taken from one assignment to a similar one? .... 37
  5.3.1 Design .............................................................................. 37
  5.3.2 Results analysis ................................................................. 40
  5.3.3 Conclusion ........................................................................ 41
5.4 Experiment 4 – Can reputation be taken from one assignment to a different one? . 41
  5.4.1 Design .............................................................................. 41
  5.4.2 Results analysis ................................................................. 44
  5.4.3 Conclusion ........................................................................ 48
5.5 Experiment 5 – Can reputation be taken from calibration to real assignments? ...... 49
  5.5.1 Design .............................................................................. 49
  5.5.2 Results analysis ................................................................. 50
  5.5.3 Conclusion ........................................................................ 52
Chapter 6 Future Work ...................................................................... 53
REFERENCES .............................................................................. 55
APPENDICES .............................................................................. 56
  Appendix A .............................................................................. 57
  Appendix B .............................................................................. 58
  Appendix C .............................................................................. 59
  Appendix D .............................................................................. 61
  Appendix E .............................................................................. 62
LIST OF TABLES

Table 4.1 Entities needed in reputation web service ................................................. 21
Table 5.1 Experiment 1 results from Hamer’s algorithm, Lauw’s algorithm and Quiz-based
algorithm acting on all available second-round peer-review records from writing assignments
fall 2015 ......................................................................................................................... 31
Table 5.2 Experiment 1 results from Hamer’s algorithm, Lauw’s algorithm and Quiz-based
algorithm acting on all available second-round peer-review records from writing assignments
spring 2016 ..................................................................................................................... 32
Table 5.3 Experiment 2 results from Quiz-based algorithm, Hamer’s algorithm and Lauw’s
algorithm acting on second-round peer-review records from writing assignments fall 2015 .... 34
Table 5.4 Experiment 2 results from Quiz-based algorithm, Hamer’s algorithm and Lauw’s
algorithm acting on first-round peer-review records from writing assignments fall 2015 .......... 36
Table 5.5 Experiment 2 results from Quiz-based algorithm, Hamer’s algorithm and Lauw’s
algorithm acting on first and second round of peer-review records from writing assignment
spring 2016 ..................................................................................................................... 36
Table 5.6 Experiment 3 results from Hamer’s algorithm with different initial reputation
values acting on second-round peer-review records from writing assignment b fall 2015 .. 40
Table 5.7 Experiment 4 results from Hamer’s algorithm with different initial reputation
values acting on second-round peer-review records from OSS project fall 2015 ................. 45
Table 5.8 Experiment 4 results from Lauw’s algorithm with different initial reputation values
acting on second-round peer-review records from OSS project fall 2015 ......... 46
Table 5.9 Experiment 5 results from Quiz-based algorithm, Hamer’s algorithm with different
initial reputation values and naïve averages acting on second-round peer-review records from
writing assignment spring 2016 ....................................................................................... 52
Table 0.1 Formative rubric of writing assignments in CSC 517 fall 2015 and spring 2016 .. 57
Table 0.2 Summative rubric of writing assignments in CSC 517 fall 2015 and spring 2016 58
Table 0.3 Peer review rubric of program 1 in CSC 517 fall 2015 ........................................ 59
Table 0.4 Formative rubric of OSS project in CSC 517 fall 2015 and spring 2016 ............ 61
Table 0.5 Summative rubric of OSS project in CSC 517 fall 2015 and spring 2016 .......... 62
LIST OF FIGURES

Figure 2.1 Calibration report for student. For criterion question with score range from 1 to 5, the background color of each score is different. The score in dark green cell is 5; the scores in green cell is 4; the score in yellow cell is 3; the score in orange cell is 2; the score in red cell is 1. .......................................................... 5

Figure 2.2 The inheritance diagram of the types of peer-review rubric questions in Expertiza. .......................................................... 5

Figure 2.3 Calibration report for instructor. The bolded percentages are the most frequent ones for each question. Scores in green cells are the expert scores given by the instructor; scores in light green cells are 1 point away from the expert review score; scores in yellow cells are 2 points away from the expert review; scores in orange cells are 3 points away from the expert review; scores in red cells are 4 points or more away from the expert review. .... 6

Figure 3.1 New quiz creating page. ........................................................................ 14

Figure 3.2 Quiz editing page with one checkbox question and one multiple-choice quiz question. .................................................................................. 15

Figure 3.3 Quiz report with total score in percentage, correct answer(s) for each question and quiz takers’ answers. ................................................................. 16

Figure 4.1 The structure of the reputation web service....................................... 20

Figure 4.2 First part of standard transmission format with sample data. ............. 22

Figure 4.3 Second part of standard transmission format with sample data. ............ 22

Figure 4.4 Third part of standard transmission format with sample data. .............. 22

Figure 4.5 Client side UI in Expertiza. ......................................................... 24

Figure 4.6 Client side UI in Expertiza with partial results from Hamer’s algorithm acting on second-round peer-review records from writing assignment 1a (724). .................. 25

Figure 4.7 Client side UI in Expertiza with partial results from Hamer-expert algorithm acting on second-round peer-review records from writing assignments (724 and 733). .... 26

Figure 4.8 Client side UI in Expertiza with partial results from Lauw’s algorithm acting on peer-review records from OSS project (736) and initial reputation values from Lauw-expert algorithm acting on second-round peer-review records from writing assignments (724 and 733). ........................................... 27

Figure 4.9 Procedure of sending encrypted request. ........................................... 28

Figure 5.1 Reputation algorithm experiment with three sets of aggregated grades calculated based on second-round peer-review records and expert grades sorted ascendingly. The x-axis represents identifiers of different artifacts; y-axis represents the grades in the range between 75 and 100. ......................................................... 34

Figure 5.2 Reputation algorithm experiment with three sets of aggregated grades calculated based on first-round peer-review records and expert grades sorted ascendingly. The x-axis represents identifiers of different artifacts; y-axis represents the grades in the range between 75 and 100. .......................................................... 35

Figure 5.3 Aggregated grades calculation procedure of experiment 3. ..................... 38

Figure 5.4 Venn diagram of students’ peer-review relationship on writing assignments. .... 39

Figure 5.5 Aggregated grades calculation procedure of experiment 4. ....................... 42
Figure 5.6 Venn diagram of students’ peer-review relationships on writing assignments, program1 and OSS project................................................................. 44
Figure 5.7 Reputation algorithm experiment with three sets of aggregated grades and expert grades sorted ascendingly. The x-axis represents identifiers of different artifacts; y-axis represents the grades in the range between 70 and 100. ......................................................... 45
Figure 5.8 Aggregated grades calculation procedure of experiment 5. ............................. 49
Figure 5.9 Reputation algorithm experiment with three sets of aggregated grades and expert grades sorted ascendingly. The x-axis represents identifiers of different artifacts; y-axis represents the grades in the range between 85 and 100. ................................................................. 51
Chapter 1  Introduction

Peer review, also known as peer assessment, occurs between students at a similar level, usually taking the same course. During peer-review, students will consider the value, quality, accuracy, and other aspects of others’ artifacts [1].

Efficient peer-review has many benefits. Firstly, peer-review may offer students a chance to learn from their peers and improve their understanding of the assignment requirements. Secondly, it can help teaching staff decide on grades for student work. This advantage is more obvious in large courses like MOOCs. Tens of thousands students take MOOCs and submit artifacts. By contrast, the number of teaching staff is quite limited. Hence it is difficult for teaching staff to evaluate and give feedback for students’ artifacts, especially for open-ended questions. On these occasions, peer-review is a promising solution that can scale to the grading of assignments with tens of thousands of students all over the world [2]. In order to obtain accurate aggregated grades from peer-review, there are many ways to do quality control, such as calibration, reputation systems, automated meta-reviews, and so on [3].

I will focus on reputation systems in this thesis. Many different reputation systems have been proposed by researchers. But making each peer-review system implement the different reputation algorithms is time consuming, or even unrealistic. A reputation web service can become one solution to this problem. With the web service, different reputation algorithms have already been implemented on the server side. Each client system can test these algorithms at any time, since the reputation web service is “pluggable” [4]. And no matter which programming language is used, client systems only need to wrap the raw data into a standard transmission format and send it to web service. Secondly, since different reputation algorithms have hardly been compared with each other before, the reputation web service can be used to compare their performance.
The rest of the thesis is organized as follows. Chapter 2 discusses related research on reputation, formative feedback, summative feedback, calibration and so on. Chapter 3 introduces each reputation algorithm in detail. Chapter 4 presents the structure of the reputation web service. Chapter 5 describes five experiments and analyses the results. Finally, Chapter 6 discusses the future work.
Chapter 2  Related Work

2.1  Validity, Reliability and Spread

Validity, reliability and spread can be taken as measurements of peer-review quality. Validity presents the difference between one peer-review grade and expert grade when expert grade is available. A lower difference represents a higher validity. There are two kinds of reliabilities, both of which can be measured when expert grades are not available. The first kind is inter-rater reliability. It measures the difference between one student’s peer-review grade and others’ on the same artifact. Lower inter-rater reliability only means the difference is small, but not always that it is better. It is because in some cases “outliers” can even be closer to ground truth. Another kind is intra-rater reliability, which presents a single peer reviewer's grading consistency for one submission in two different time periods. Intra-rater reliability is hardly ever used. So most of time when talking about reliability, I mean inter-rater reliability. The last measurement is spread. “Spread is a measure of the tendency of a reviewer to assign different scores to different work. Generally, a higher spread is better, because it indicates that the reviewer is discriminating between good and bad work.” [3]

2.2  Formative Feedback and Summative Feedback

Peer review is often used for formative feedback only [5]. In the peer-review system named Expertiza,1 many assignments have two rounds of peer-reviews with different rubrics. This mechanism is named, vary-rubric-by-round. In the first review round, a formative review rubric is assigned to peer reviewers. The purpose of formative rubric is to encourage peer reviewer to offer insightful feedback to authors in order to help them improve the quality of the artifact. During the second review round, peer reviewers will work on a summative rubric to check the overall quality of each artifact and give a grade to each of them.

---

1 Expertiza is a web application where students can submit artifacts and peer review others’ work (articles, code, web sites, etc.). It is used in select courses at NC State and other colleges and universities. Expertiza is an open-source project, and the source code is available on GitHub (https://github.com/expertiza/expertiza).
2.3 Calibration

One of the most widely used ways to ensure the peer-review quality is calibration. In later 1990, Calibrated Peer Review is built, which is the pioneer of this area. And calibration is also used in large MOOC platforms, such as Coursera. [3]

In CSC 517 in Spring 2016, a calibration assignment was created before the submission deadline of the first assignment. The instructor chose several representative artifacts from former semesters and submitted them to Expertiza. These artifacts had major differences in quality. After submitting those artifacts, the instructor chose a peer-review rubric and submitted an expert peer-review for each artifact. This peer-review was regarded as “ground truth”; that is, the base for calibrating all other peer-reviews. The instructor then assigned a certain number of prepared artifacts to each student. During the class, students spent some time doing peer-reviews on these pre-assigned artifacts. Expertiza then generated reports for each student and for instructor. The instructor could see how student reviews differed from his reviews.

Figure 2.1 presents the calibration report for one student. For each rubric question, the question title will be displayed at the top. And there is a table for each question to demonstrate the difference between expert review and student’s review, including answers and comments. In order to make the difference straight-forward, different background colors are used to mark different scores. In this case, the scores given by this student is quite close to expert scores and the maximum gap is only one point.
Figure 2.1 Calibration report for student. For criterion question with score range from 1 to 5, the background color of each score is different. The score in dark green cell is 5; the scores in green cell is 4; the score in yellow cell is 3; the score in orange cell is 2; the score in red cell is 1.

Figure 2.2 The inheritance diagram of the types of peer-review rubric questions in Expertiza.

There are three main types of peer-review rubric questions (shown in Figure 2.2), that is choice question, text response and upload file. Choice question has two subtypes, scored question and unscored question. Criterion and scale are scored questions; conversely, dropdown, multiple-choice and checkbox are unscored questions. It is worth noting that in Expertiza, only scored questions are included in peer-review grades. As a scored question type, criterion is the combination of dropdown and text area. It means that peer reviewers can not only give a score to certain questions, but also write some text comments. And criterion is one of the most frequently used question types in Expertiza.
The calibration report for the instructor is shown in Figure 2.3. In contrast to the student’s report, the instructor’s report shows information for the whole class. Each cell with colored background shows the percentage of students who chose certain option of one question. It uses a similar color blending as the student report. Bolded percentages indicate the most frequent answer for each question. Figure 2.3 shows that for question 1 and 2, most students gave the same score as the instructor and for question 3 most students gave 3, but the instructor gave 4.

![Calibration report for instructor](image)

**Figure 2.3** Calibration report for instructor. The bolded percentages are the most frequent ones for each question. Scores in green cells are the expert scores given by the instructor; scores in light green cells are 1 point away from the expert review score; scores in yellow cells are 2 points away from the expert review; scores in orange cells are 3 points away from the expert review; scores in red cells are 4 points or more away from the expert review.

### 2.4 Reputation

Reputation is a quantization measurement to judge which peer reviewer are more reliable. Several algorithms have already been created to calculate reputation values from peer-review grades [5, 6]. Basically, each algorithm will consider one or more measurements, such as validity, reliability and spread. And “reputation values can be used to either give credit to students for careful reviewing or to weight peer-assigned grades” [3].
Chapter 3  Reputation Systems

In this chapter, five reputation algorithms are introduced in detail, including Hamer’s algorithm and Lauw’s algorithm. Reputation algorithms are the main part of reputation web service. As mentioned above, one advantage of using a web service is that peer-review systems can access more than one algorithm and choose the most suitable one for further grade aggregation. Then several suggestions for improving the accuracy of these algorithms are presented.

3.1   Reputation Algorithms

3.1.1   Hamer’s Algorithm

Hamer’s algorithm was published in 2005. This algorithm generates two quantities, namely a grade for each artifact and a weight for each peer reviewer. The weight for each peer reviewer can be regarded as the reputation. And the artifact grades are actually aggregated grades with each reviewer's reputation value as weight. [5] The input of Hamer’s algorithm is an adjacency matrix. Rows of this adjacency matrix represents different artifacts and columns presents different peer reviewers. And each item of adjacency matrix is the grade that certain peer reviewer gave to specific artifact.

To compute the reputation of each peer reviewer, the algorithm first assigns each reviewer the same reputation, which is one. Each artifact’s first-iteration aggregated grade is the sum of each peer-review grade divided by the number of peer-reviews. Then the algorithm calculates the difference between aggregated grade of each artifact and each peer-review grade. The larger this difference, the more inconsistent this peer reviewer is compared to other peer reviewers. The algorithm then updates the reputation of each peer reviewer accordingly. After that, the newly updated reputation values are used to revise the aggregated grade of each artifact.
Steps mentioned above are called one round of iteration. During each iteration, both of peer reviewers’ reputation values and artifacts’ aggregated grades are updated once. In most case, after several rounds of iterations, a convergence will occur. And the criterion of convergence can be set differently. In this algorithm, convergence is defined that previous-round reputation values and current-round reputation values are identical on four-digit decimal precision.

A more accurate definition of Hamer’s algorithm is mentioned below. Let

- $a$ be an artifact;
- $A$ be the set of all artifacts;
- $r$ be a peer reviewer;
- $R$ be the set of all peer reviewers;
- $g^r_a$ be the grade that peer reviewer $r$ assigned to $a$;
- $R_r$ be the set of artifacts reviewed by $r$;
- $A_a$ be the set of peer reviewers who have reviewed artifact $a$;
- $W_r$ be the reputation value of peer reviewer $r$;
- $G^a_{expert}$ be the expert grade of artifact $a$;
- $G^a_{predict}$ be the aggregated grade of artifact $a$ based on current-round reputation values and peer-review grades.

Hamer’s algorithm is suitable to a situation in which ground truth is not available. In this way, aggregated grades have to take the place of ground truth temporarily and be updated during each iteration. $G^a_{predict}$ represents the weighted average of peer-review grades assigned to artifact $a$:

$$G^a_{predict} = \frac{\sum_{r \in A_a} g^r_a \cdot W_r}{\sum_{r \in A_a} W_r}$$ (1)
After obtaining the aggregated grades $G^a_{predict}$ for each artifacts, Hamer’s algorithm can use them to update the reputation value of each peer reviewer. It can be derived by comparing peer-review grades and aggregated grades:

$$\Delta_r = \frac{\sum_{a \in R_r} (G^a_{predict} - g^a)^2}{|R_r|}$$ (2)

The higher $\Delta_r$ value, the lower consistency between grades assigned by this peer reviewer and others. It also indicates that this peer reviewer has low reliability, which leads to low reputation. So reputation value can be assigned in inverse proportion:

$$W_r \propto -1 \Delta_r$$ (3)

In Hamer’s algorithm, $W_r$ is not simply equal to $1/\Delta_r$. Instead, the equation multiplies the average of $\Delta_r$ ($mean\Delta_r$) as a constant:

$$W'_r = \frac{mean\Delta_r}{\Delta_r}$$ (4)

The reason why is that if one peer reviewer’s assigned grades are quite close to the aggregated grades, his/her $\Delta_r$ value can be very small. Then the reputation value can be so large that his/her peer-review grades can not only dominate the final aggregated grades this time, but also in the future. In order to reduce this kind of impact and scale reputation values, Hamer’s algorithm multiplies $mean\Delta_r$ to original value.

Hamer also used a “log-damping” approach to further scale the reputation values to a controlled range. [5]

$$W_r = \begin{cases} 2 + \log(W'_r - 1) & \text{if } W'_r > 2 \\ W'_r & \text{if } W'_r \leq 2 \end{cases}$$ (5)
\[ G_{predict}^a \text{ and } W_r \text{ keep updating until convergence occurs. And the reputation range of Hamer’s algorithm is } (0, \infty) \]

### 3.1.2 Hamer-expert Algorithm

If the ground truth (expert grade) of each artifact is available, the algorithm only need to iterate once and get the reputation values of peer reviewers easily. So Hamer’s algorithm with expert grade is called Hamer-expert Algorithm [4]. The input of Hamer-expert algorithm is also an adjacency matrix, the same as Hamer’s algorithm.

To compute the reputation of each peer reviewer, the algorithm will calculate the difference between the expert grade and each peer-review grade. The larger this difference, the more inconsistent this peer reviewer compares to teaching staff. In this case, Hamer-expert algorithm will update the reputation of each peer reviewer only once. Since the algorithm use expert grades instead of aggregated grades and expert grades do not need to be updated. So the reputation values after the first round iteration are the final outputs of Hamer-expert algorithm.

Just like Hamer’s algorithm, Hamer-expert algorithm also used a “log-damping” approach to further scale the reputation values to a controlled range. The formula can be the same as formula (5). The range of Hamer-expert algorithm is also \((0, \infty)\).

### 3.1.3 Lauw’s Algorithm

Lauw’s algorithm focuses on two quantities, namely, leniency of each peer reviewer and quality of each artifact. Peer reviewers who tend to give higher grades will have leniency bigger than 0, while those who tend to give lower grades will have leniency smaller than 0 [6]. And the absolute value of leniency depends on how far one grade is away from the expert grades or aggregated grades. The input to Lauw’s algorithm is also an adjacency matrix. Each row and column represents different artifacts and peer reviewers separately.
And each item of adjacency matrix is the grade that certain peer reviewer gave to a specific artifact.

Lauw’s algorithm first assigns each peer reviewer the same leniency 0, which assumes all peer reviewers are unbiased at the beginning. The first-round aggregated grade of each artifact is the naive average of corresponding peer-review grades. In each iteration, the algorithm calculates the difference between the aggregated grade of each artifact and each peer-review grade, then updates the leniency of each peer reviewer accordingly. Then newly updated leniencies can be used to tune the aggregated grade of each artifact. Ideally, after several-round iterations, a convergence will occur.

A more accurate definition of Lauw’s algorithm is mentioned below. Let

- $\alpha$ be the user-determined compensation scaling factor;
- $a$ be an artifact;
- $A$ be the set of all artifacts;
- $r$ be a peer reviewer;
- $R$ be the set of all peer reviewers;
- $g^r_a$ be the grade that peer reviewer $r$ assigned to $a$;
- $R^r$ be the set of artifacts reviewed by $r$;
- $A_a$ be the set of peer reviewers who have reviewed artifact $a$;
- $l_r$ be the leniency of peer reviewer $r$;
- $W_r$ be the reputation value of peer reviewer $r$;
- $G_{expert}^a$ be the expert grade of artifact $a$;
- $G_{predict}^a$ be the aggregated grade of artifact $a$ based on current-round reputation values and peer-review grades.

In Lauw’s algorithm, the initial leniency of each peer reviewer is 0. $G_{predict}^a$ is used to represent the aggregated grade and will be updated during each iteration:
\[ G_{\text{predict}}^a = \frac{\sum_{r \in A_a} g_i^a*(1-\alpha*l_r)}{|A_a|} \] (6)

Sign \( \alpha \) is the compensation scaling factor. It “controls the extent to which the scores may be adjusted to compensate for leniency” [6]. The range of \( \alpha \) can be from 0 to 1 (include 0 and 1). Here I defined the value of \( \alpha \) as 0.5.

After obtaining the aggregated grade \( G_{\text{predict}}^a \) for each artifact, Lauw’s algorithm can use them to update the leniency of each peer reviewer. The update can be achieved by comparing peer reviewers’ assigned grades and aggregated grades:

\[ l_r = \frac{\sum_{a \in R_r} (g_i^a - G_{\text{predict}}^a) / g_i^a}{|R_r|} \] (7)

During each iteration, the algorithm checks the range of each peer reviewer’s leniency and restricts the range of leniency between –1 and 1 (including –1 and 1), which means each leniency smaller than –1 will rescale to –1 and each leniency bigger than 1 will rescale to 1. In most cases, the difference between the peer-review grade and aggregated grade will not exceed the value of peer-review grade itself. This restriction is to avoid some extreme situations that cause some quite high (or low) leniencies to dominate the values of aggregated grades. During the calculation, \( G_{\text{predict}}^a \) and \( l_r \) keep being updated until convergence occurs. Then the reputation of each peer reviewer can be calculated like this:

\[ W_r = 1 - |l_r| \] (8)

So the reputation range of Lauw’s algorithm is [0,1].
3.1.4 Lauw-expert Algorithm

If the expert grade of each artifact is available, they can be used instead of the aggregated grades generated by each iteration. And I call Lauw’s algorithm with expert grade Lauw-expert Algorithm [4].

The input of the Lauw-expert algorithm is the same as Lauw’s algorithm. Since now the expert grades are available, there is no need to calculate the aggregated grade of each artifact during each iteration. Actually only one iteration is required and formulas (7) and (8) can help obtain the reputation of each peer reviewer. And there is one difference between Lauw’s algorithm and Lauw-expert algorithm in formula (7), that is, in the Lauw-expert algorithm, $G^{a}_{\text{predict}}$ are replaced by $G^{a}_{\text{expert}}$. The reputation range of the Lauw-expert algorithm is also [0,1].

3.1.5 Quiz-based Algorithm

Student-generated questions have been considered as a helpful process to motivate critical thinking for a long time [8, 9]. The process of creating a quiz is an instance of self-directed learning. At very beginning, the authors have to think about which parts of the artifacts are suitable for creating quiz questions. And then they have to create several potential options for each quiz question with one or more correct answers. Quiz scores can generate several interesting metrics. First of all, teaching staff can have a general idea on whether peer reviewers read others’ artifacts carefully or not. Secondly, the author can check whether peer reviewers understand the key points of each artifact. Thirdly, the quiz score can be regard as the reputation value to compute the aggregated grade of each artifact [9]. It is reasonable that in most cases when one student gets a higher quiz score, it means he/understand one artifact better than others and his/her peer-review to this artifact can also be more reliable than others.
In Expertiza, there is a student-generated quiz feature. It supports multiple-choice questions and checkbox questions, both of which have four options. And the number of quiz questions is decided by instructor.

![New Quiz](image)

**Figure 3.1** New quiz creating page.

Figure 3.1 shows the UI for creating quiz questions. In this case the number of quiz question is 3, and there is no restriction on the number of certain kind of questions. It means that author can create three multiple-choice questions or three checkbox questions in this case, although the complexities of these two kinds of quizzes will be different.

After typing in the title of each question, author can choose either multiple-choice question or checkbox question, fill in the four options and then choose the correct answer(s). Figure 3.2 presents the UI of two created questions. The first one is a checkbox question with all four options being correct answers and the second one is a multiple-choice question with only the last option being the correct answer.
After the end of the second round of peer-review, reviewers can take the quizzes. During the submission stage and first-round peer-review stage, authors may modify the contents of their artifacts according to suggestions from peer reviewers. But after the second-round peer-review, contents of all artifacts are unchangeable. So it is the best time for taking quizzes.

After peer reviewers submit their quizzes, they can see the result report immediately. At the top of the report, the accuracy rate is displayed. Peer reviewers need to get 80% or higher percentage to be considered as passing the quiz [9]. Part of the report is presented in Figure 3.3. This peer reviewer’s accuracy score is 40%, which means this peer reviewer did not pass this quiz. And each peer reviewer can only take each quiz once.
Figure 3.3 Quiz report with total score in percentage, correct answer(s) for each question and quiz takers’ answers.

The input of the Quiz-based algorithm is also an adjacent matrix. Each row stands for an artifact and each column stands for a peer reviewer. And each item of adjacency matrix is the quiz score. One important thing is that one peer reviewer can only take a quiz if s/he reviewed the artifact created by the same author [9]. As mentioned above, quiz scores can also be regarded as reputation values. So I can calculate the aggregated grades based on peer-review grades and corresponding quiz scores.

A more precise definition of Quiz-based algorithm is given below. Let

- $a$ be an artifact;
- $A$ be the set of all artifacts;
- $r$ be a peer reviewer;
- $g^r_a$ be the grade that peer reviewer $r$ assigned to $a$;
- $A_a$ be the set of peer reviewers who have reviewed artifact $a$;
- $N^r_a$ be the number of correct answers that peer reviewer $r$ gave;
\begin{itemize}
    \item $N^\text{total}_a$ be the total number of quiz questions in the quiz of artifact $a$;
    \item $Q^r_a$ be the quiz score that peer reviewer $r$ obtained on quiz of artifact $a$;
    \item $G\text{\textsubscript{predict}}^a$ be the aggregated grade of artifact $a$ based on current-round reputation values and peer-review grades.
\end{itemize}

In the Quiz-based algorithm, the weight of each quiz question is one. So the quiz score that peer reviewer $r$ obtained is calculated by dividing the number of correct quiz answers by total number of questions in this quiz:

$$Q^r_a = \frac{N^r_a}{N^\text{total}_a} \quad (9)$$

So the reputation range of Quiz-based algorithm is $[0,1]$. Then the aggregated grade of each artifact can be computed by using quiz scores as weights:

$$G\text{\textsubscript{predict}}^a = \frac{\sum_{r \in A_a} g^r_a \cdot Q^r_a}{|A_a|} \quad (10)$$

### 3.2 Suggestions for Improving the Accuracy of Reputation Algorithms

Hamer’s algorithm and Lauw’s algorithm are easy to implement and can obtain the results quickly. However, it is possible that starting out with the simple average scores as aggregated grades will not always lead to convergence (also known as a fixed point) [5]. So in the modified method, both Hamer’s algorithm and Lauw’s algorithm will iterate 10,000 times first. If reputation values have still not reached a fixed point, the method will reduce the digits of decimal precision, and trying another 10,000 rounds of iterations until reaching the fixed point.

However, even the reputation values eventually reach a fixed point it can be a locally optimal solution, instead of the globally optimal solution. It is because the original Hamer’s
algorithm and Lauw’s algorithm assume that the initial reputation values assigned to each peer reviewer is always equal to one, which means that each peer reviewer’s abilities are the same. One alternative way is to use multiple sets of inputs to calculate the results several times and pick up the best one. And instead of assigning a random reputation value to each student as initial condition, considering other metrics, such as calibration results or reputation values from former assignments can be a better way to get the globally optimal solution.

Research shows that the performance of Hamer’s algorithm and Lauw’s algorithm varies a lot on different kinds of assignments [4]. For instance, on writing assignments Hamer’s algorithm has a quite low overall bias. But for coding assignments, such as CSC 517 OSS project and assignments combine with writing and coding, such as CSC 517 final project, their performance is not satisfactory. So the type of assignment should also be taken into consideration.
Chapter 4  Reputation Web Service

This chapter first gives the reason why a reputation web service is necessary, then presents the structure of web service in detail. It includes the design of server side and client side, how to embed the client side into the peer-review system, and the standard data transmission format and security methods to encrypt the sensitive data.

4.1 Why Web Service

Making each peer-review system implement different reputation algorithms is time consuming, or even unrealistic. What’s more, the quantity of reputation algorithms on each client system is very limited. So it is quite difficult for each client system to compare different algorithms and try to select the best one.

A reputation web service can be a good way to solve these problems. Firstly, on the server side several reputation algorithms have already been implemented. And client systems do not need to deploy these algorithms themselves. All they need to do is wrap the raw data into a standard transmission format and send it to web service. This is easy to manage and saves the developer a lot of time. If someone figures out a new way to calculate the reputation values, I can implement it in the web service and let different client system try this new method. It makes the reputation web service “pluggable” with any system [4]. And all client systems can evenly share these resources. Secondly, since different reputation algorithms have not previously been compared with each other, the reputation web service can be used to compare different algorithms and pick the most suitable one. And after gathering enough information, reputation web service can recommend several algorithms based on prior knowledge for some kinds of assignments [6]. One criterion to decide the most suitable reputation algorithm is that aggregated grades calculated based on its results are closest to expert grades.
4.2 Web Service Structure

The reputation web service consists of three main parts, that is client side, server side and the standard transmission format. Figure 4.1 shows the structure of the web service in general. Several client systems are communicating with reputation web service. Since each system has its unique DB schema, different data wrappers are needed to convert raw data into standard transmission format. And on the server side, another data wrapper is used to parse the request, generate adjacency matrices, which are the inputs to reputation algorithms. Finally, the reputation web service sends the results back to client systems.

Figure 4.1 The structure of the reputation web service.

4.3 Peer-Review Markup Language

Peer-Review Markup Language (PRML) is a generic schema for encapsulating the raw data into standard data transmission format [4]. In this way, different client systems can communicate with the reputation web service without changing their database schemas.
This language defines some entities commonly used in different peer-review systems. The entities used in the reputation web service are a subset of the data defined in PRML, including clients, assignments, tasks, reviewers, reviewed entities and peer-review grade [4]. Table 4.1 explains each entity in detail.

Table 4.1 Entities needed in reputation web service

<table>
<thead>
<tr>
<th>Entity name</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>A peer-review system that communicates with the reputation web service.</td>
</tr>
<tr>
<td>Assignment</td>
<td>A unit of work that the instructor assign to students.</td>
</tr>
<tr>
<td>Task</td>
<td>A task is the smallest unit in each assignment and each task includes one deadline.</td>
</tr>
<tr>
<td>Actor</td>
<td>A user who did one or more peer-reviews.</td>
</tr>
<tr>
<td>Artifact</td>
<td>The submissions for the current assignment. They can be links, files, etc. The reputation web service does not consider the content of each artifact; only the identifier of each artifact matters [4].</td>
</tr>
<tr>
<td>Answer</td>
<td>The number of points each reviewer gives to each artifact. Each peer-review record contains identifier of artifact, identifier of reviewer and peer-review grade.</td>
</tr>
</tbody>
</table>

Standard data transmission format for reputation web service is a JSON-based format with compact structure, which has three parts. The first part is the information related to assignment(s) and task(s). Figure 4.2 shows the first part, assignment information, with sample data. It allows data coming from multiple assignments and appointing one task for each assignment. According to the sample data, two assignments’ second-round peer-review records will be sent to reputation web service. And maximum and minimum grades of each assignment are also mentioned to help calculate the reputation values.
The second part of standard JSON format is the additional information. They can be initial Hamer reputation values, initial Lauw reputation values, expert grades or quiz scores. Data presented in Figure 4.3 is used for different reputation algorithms. For instance, expert grades are extra inputs of Hamer-expert algorithm and Lauw-expert algorithm; quiz scores are additional inputs of Quiz-based algorithm. Details of each algorithm will be stated in next chapter.

The last part is the review records. It is the most important part because each line records how many points each peer reviewer giving to certain artifact. Figure 4.4 presents the sample peer-review records.
4.4 Server Side Design

The server side uses Ruby on Rails framework and follows the MVC design pattern. Each algorithm was implemented in a model file. And the controller focuses on parsing JSON request to adjacency matrices, building data structure, calling different algorithms and sending results back to client system. In reputation web service there is no need to create views because all messages will be transmitted via JSON format. In Figure 4.1, there is only one data wrapper needed for server side. It is a big advantage of reputation web service, that is using standard JSON transmission format can not only unify the interface, but also satisfy the needs of different client systems.

4.5 Client Side Design

Figure 4.1 also presents that each peer-review system needs one specific data wrapper. It is because database structure of each system is different. However, the data wrapper is the only thing each client system need to build. So comparing with understanding the logic of reputation algorithms and implementing them, just building a data wrapper can save lots of time and effort. Currently, one data wrapper has already been built and been embedded into Expertiza with a user interface.
The basic user interface of the client side is presented in Figure 4.5. The instructor follows four simple steps to send the standard JSON request. The first step is to type in identifier(s) of assignment(s). These text fields only accept numerical values in order to avoid mistyping. The second assignment identifier text field is optional, which is designed for writing assignments (writing assignment 1a and 1b). Normally, in CSC 517 course, there are two writing assignments. Since they are similar to each other, I tend to merge these two assignments into one sometimes. The last text field is used to specify round number of assignment. The default round number is 2, which means to use the second-round peer-review records as inputs. This bases the reviewer’s reputation on that reviewer’s second-round reviews only.

The second step is to choose different kinds of reputation algorithms. They are Hamer’s algorithm, Lauw’s algorithm, Hamer-expert algorithm, Lauw-expert algorithm and Quiz-based algorithm. Thirdly, instructor needs to choose some additional information. It can be expert grades, initial reputation values or quiz scores. For initial reputation values, the instructor can choose either from Hamer-expert algorithm or Lauw-expert algorithm and type.
in corresponding assignment identifier(s). And the final step is to click the “Send request” button.

Figure 4.6 shows the results of writing assignment 1a using Hamer’s algorithm with second-round peer-review records. The table in Figure 4.6 presents the request and response information with color blending.

![Image](image.png)

**Figure 4.6** Client side UI in Expertiza with partial results from Hamer’s algorithm acting on second-round peer-review records from writing assignment 1a (724).

The results of writing assignments using Hamer-expert algorithm with expert grades are shown in Figure 4.7. The checkbox before “Add expert grades” is gray, which means it is disabled, cannot be unchecked. The reason is that when instructor chose the Hamer-expert algorithm, the data wrapper needed to add expert grades into request information by default. If instructor unchecks the “Add expert grades” for some reason, it will lead to a conflict. So in order to avoid it, some constraints have been added to this user interface. When the instructor chooses Hamer-expert algorithm or Lauw-expert algorithm, the “Add expert
grades” checkbox will be checked and disabled; when the instructor chooses the Quiz-based algorithm, the “Add quiz scores” will be checked and disabled and so on.

Figure 4.7 Client side UI in Expertiza with partial results from Hamer-expert algorithm acting on second-round peer-review records from writing assignments (724 and 733).

Figure 4.8 shows the results from another more complex situation. In this case, initial reputation values for Lauw’s algorithm will not be one (the default value) any more. Instead, these values came from writing assignments. This feature is designed for the third experiment, which will be explained in detail in chapter 5.
Figure 4.8 Client side UI in Expertiza with partial results from Lauw’s algorithm acting on peer-review records from OSS project (736) and initial reputation values from Lauw-expert algorithm acting on second-round peer-review records from writing assignments (724 and 733).

4.6 Security of Web Service

Security is also an important issue, since expert grades and peer-review grades are sensitive data and should not be revealed to unauthorized people. However, according to the design above, the reputation web service sends the data in plaintext. In order to protect these sensitive data, encryption algorithms are needed.

The first solution is to use public-key cryptography. It is an asymmetric key encryption algorithm and cryptographic keys are paired. One is public key, which is disseminated widely and anyone with public key can encrypt messages. The other one is the private key, which can only be used by the keyholder to decrypt private messages [10]. By implementing this solution, client sides can use public key to encrypt the JSON data and server side can use corresponding private key to decrypt the encrypted data. However, there is a maximum message length restriction for public-key cryptography. Since it is possible that request data exceeds the maximum length restriction, another method is needed to apply to all situations.
The second solution is the combination of asymmetric key encryption algorithm and symmetric key encryption algorithm, which does not have restriction mentioned above. Procedure of sending encrypted request is presented in Figure 4.9. The client side uses a newly generated symmetric key to encrypt the JSON data and then encrypts the symmetric key with the public key from the asymmetric key encryption algorithm. After that, it sends the encrypted request to the server side. Then the server side decrypts the symmetric key with the private key. Secondly, it obtains the JSON data by using the symmetric key. And sending the response back to client side is the reverse process. In practice, AES is chosen as the symmetric encryption algorithm, and RSA is chosen as the asymmetric encryption algorithm.

Figure 4.9 Procedure of sending encrypted request.

In summary, the “pluggable” reputation web service can make peer review systems access to multiple reputation algorithms and compare with each other. So there is no need to implement reputation algorithms locally. But each client system needs a specific data
wrapper. The data wrapper can convert client system’s database schema into a standard JSON transmission format, which is the subset of PRML. After reputation web service receives the JSON request, it will do calculation and send the JSON response back to client system. What’s more, the reputation web service also uses cryptography to protect the sensitive data.
Chapter 5  Experiments

In this chapter, five reputation algorithm experiments are presented. The purpose of these experiments is try to improve the performance of these reputation algorithms by using different inputs. For the basic Hamer’s and Lauw’s algorithm, in order to improve the accuracy of these algorithms, experiments are designed to use the different initial reputation values instead of one. For first one or two assignments in one course, initial reputation values can come from calibration assignment because there is not former regular assignment to obtain the peer-review records. For later assignments, initial reputation values can come from the results of former assignments. And for vary-rubric-by-round assignments, it is interesting to compare the aggregated grades based on formative rubric with those based on summative rubric.

Although these experiments are based on data from Expertiza, the reputation web service provides a new way for teaching staff to design their own experiments in order to find useful patterns to calculate the aggregated grades on their own courses. What’s more, it is not necessary for them to have any programming skills. They only need to make their systems connect to reputation web service.

5.1  Experiment 1 – How to calculate the aggregated grades based on reputation values?

5.1.1  Design

The purpose of this experiment is to introduce the process of aggregated grades calculation. It includes three main steps. The first one is calculating different sets of reputation values by using reputation web service. The second step is to use these reputation values as weights to compute the aggregated grades (weighted average grades) for each artifact. Finally, we compare different sets of aggregated grades with expert grades by using several metrics.
Writing assignments in CSC 517 fall 2015 course and CSC 517 spring 2016 course are selected to do this experiment because all three reputation algorithms are available to them.

After finishing this experiment, three sets of aggregated grades can be compared with expert grades. They are listed below:

1. Aggregated grades calculated by Hamer’s algorithm;
2. Aggregated grades calculated by Lauw’s algorithm;
3. Aggregated grades calculated by Quiz-based algorithm;

In this experiment, all available records (second-round peer-review records and quiz-related records) are used to calculate the aggregated grades.

### 5.1.2 Result analysis

Table 5.1 presents the results from the three algorithms acting on peer-review records from writing assignments fall 2015. All available peer-review records are used as inputs to reputation algorithms in this experiment. Conversely, later experiments may only use records from peer reviewers who met the experiment requirements.

Bias range, average absolute bias and root mean square error (RMSE) are used as metrics to evaluate the performance of difference algorithms. According to the results from Table 5.1, Lauw’s algorithm has the lowest average absolute bias and RMSE.

**Table 5.1** Experiment 1 results from Hamer’s algorithm, Lauw’s algorithm and Quiz-based algorithm acting on all available second-round peer-review records from writing assignments fall 2015.

<table>
<thead>
<tr>
<th>Writing assgts, fall 2015</th>
<th>Hamer’s alg.</th>
<th>Lauw’s alg.</th>
<th>Quiz-based alg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. abs. bias</td>
<td>4.810</td>
<td><strong>3.578</strong></td>
<td>3.816</td>
</tr>
<tr>
<td>RMSE</td>
<td>6.559</td>
<td><strong>4.902</strong></td>
<td>5.021</td>
</tr>
</tbody>
</table>
Table 5.2 shows the results from writing assignments spring 2016. It is obvious that these results are better than those shown in Table 5.1. It may be the reason of calibration, since students finished calibration process at the beginning of spring 2016 semester. According to Table 5.1, Hamer’s algorithm has the lowest average absolute bias and RMSE. And it also has the smallest bias range.

Table 5.2 Experiment 1 results from Hamer’s algorithm, Lauw’s algorithm and Quiz-based algorithm acting on all available second-round peer-review records from writing assignments spring 2016.

<table>
<thead>
<tr>
<th></th>
<th>Hamer’s alg.</th>
<th>Lauw’s alg.</th>
<th>Quiz-based alg.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Avg. abs. bias</td>
<td>2.977</td>
<td>3.170</td>
</tr>
<tr>
<td></td>
<td>RMSE</td>
<td>3.671</td>
<td>3.932</td>
</tr>
</tbody>
</table>

5.2 Experiment 2 – How much do reputations change by rounds within one assignment?

5.2.1 Design

This experiment is about comparing the performance of three algorithms (Quiz-based algorithm, Hamer’s algorithm and Lauw’s algorithm) acting on peer-review records from writing assignment fall 2015 and spring 2016 separately. Since the Quiz-based algorithm does not rely on expert grades, Hamer’s and Lauw’s algorithm are chosen to do this experiment instead of Hamer-expert algorithm and Lauw-expert algorithm.

In last experiment, the input data of different algorithms are second-round peer-review grades. The reason is that second-round peer-review uses summative rubric which can represent the overall quality of each artifact and artifacts during second-round peer-review are unchangeable. But second-round peer-review records may not be a better indication of how good a peer reviewer the student is, because the first-round rubric is more detailed. In
order to verify which round of rubric can get the more accurate aggregated grades, peer-review records of this experiment will not only from second-round peer-review, but also from first-round peer-review.

In this experiment, students who finished both rounds of peer-reviews and took relative quizzes are considered. And four sets of aggregated grades are generated to compare with expert grades. All sets of grades are listed below:

1. Aggregated grades calculated by Quiz-based algorithm;
2. Aggregated grades calculated by Hamer’s algorithm;
3. Aggregated grades calculated by Lauw’s algorithm;
4. Naive averages calculated from peer-review grades;
5. Expert grades.

5.2.2 Results analysis

For writing assignment fall 2015, there are 38 out of 95 students who met all requirements. Based on the fact that doing second-round peer-reviews and taking quizzes are voluntary, this proportion (40%) is quite reasonable. Hence the final results can also be reliable.

Figure 5.1 shows the aggregated grades calculated from second-round peer-review records. Artifacts are sorted by expert grades in ascending order. The orange line, blue line and green line represent the Quiz-based algorithm, Hamer’s algorithm and Lauw’s algorithm respectively. For most artifacts, the orange line and green line overlap with each other, which indicates the results computed by Quiz-based algorithm and Lauw’s algorithm are quite close. Conversely, the blue line has larger fluctuations when expert grades are lower than 91. In general, there still exists big gaps (gap larger than 5 points) between all sets of aggregated grades and expert grades.
Figure 5.1 Reputation algorithm experiment with three sets of aggregated grades calculated based on second-round peer-review records and expert grades sorted ascendingly. The x-axis represents identifiers of different artifacts; y-axis represents the grades in the range between 75 and 100.

According to Table 5.3, the bias range of third set (aggregated grades calculated by Lauw’s algorithm) is the smallest, and the average absolute bias and RMSE of third set are also the lowest among all three sets of aggregated grades. Surprisingly, naive averages from peer-review grades performs better that Quiz-based algorithm and Hamer’s algorithm. Its performance is close to Lauw’s algorithm, which is the best one among all aggregated grades. Further experiments are needed to check out the reason of this phenomenon.

Table 5.3 Experiment 2 results from Quiz-based algorithm, Hamer’s algorithm and Lauw’s algorithm acting on second-round peer-review records from writing assignments fall 2015.

<table>
<thead>
<tr>
<th></th>
<th>Quiz-based alg.</th>
<th>Hamer’s alg.</th>
<th>Lauw’s alg.</th>
<th>Naive averages</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Avg. abs. bias</td>
<td>3.782</td>
<td>4.362</td>
<td><strong>3.646</strong></td>
</tr>
<tr>
<td></td>
<td>RMSE</td>
<td>5.068</td>
<td>5.991</td>
<td><strong>4.805</strong></td>
</tr>
</tbody>
</table>
Aggregated grades calculated from first-round peer-review records are presented in Figure 5.2. Comparing with Figure 5.1, it is obvious that aggregated grades from first-round peer-review records have larger variation than those from second-round peer-review records, especially for last eight artifacts.

![Figure 5.2 Reputation algorithm experiment with three sets of aggregated grades calculated based on first-round peer-review records and expert grades sorted ascendingly. The x-axis represents identifiers of different artifacts; y-axis represents the grades in the range between 75 and 100.](image)

Table 5.4 shows the results in detail, which are based on the three algorithms excluding the Quiz-based algorithm. Quiz-based algorithm is available only on second-round peer-review records because peer reviewers can only access quizzes after second-round peer-review finished. And the lower bounds of bias range are quite low for all three algorithms.

According to the original peer-review records, two rounds of peer-reviews on one artifact differ greatly. The average grade of first-round peer-review is almost 40 points lower than that of second-round peer-review. And comments from first-round peer-review concentrate on “no citation” and “poor organization”, which did not occur again in second-round peer-review. So it is highly possible that this artifact was not finished during the first-round peer-review, and these problems were solved before the second-round peer-review.
Table 5.4 Experiment 2 results from Quiz-based algorithm, Hamer’s algorithm and Lauw’s algorithm acting on first-round peer-review records from writing assignments fall 2015.

<table>
<thead>
<tr>
<th>writing assgts, fall 2015</th>
<th>Hamer’s alg.</th>
<th>Lauw’s alg.</th>
<th>Naive averages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bias range</td>
<td>[-32.107, 8.696]</td>
<td>[-34.426, 9.048]</td>
<td>[-34.556, 9]</td>
</tr>
<tr>
<td>Avg. abs. bias</td>
<td>5.624</td>
<td>7.993</td>
<td>8.089</td>
</tr>
<tr>
<td>RMSE</td>
<td>7.696</td>
<td>10.260</td>
<td>10.394</td>
</tr>
</tbody>
</table>

According to Table 5.4, Hamer’s algorithm performs best among all three sets. However, the best results based on first-round peer-review records are worse than any results calculated from second-round peer-review records. There is a reason to believe that using second-round peer-review records can obtain aggregated grades with higher accuracy than using first-round peer-review records.

For writing assignment spring 2016, there are 48 out of 57 students who finished both rounds of peer-reviews and took corresponding quizzes. Almost 85% of all students enrolled in this course met the requirements. The results should be persuasive. According to Table 5.5, Hamer’s algorithm performs the best on both rounds. And using second-round peer-review records can obtain aggregated grades with higher accuracy for this assignment.

Table 5.5 Experiment 2 results from Quiz-based algorithm, Hamer’s algorithm and Lauw’s algorithm acting on first and second round of peer-review records from writing assignment spring 2016.

<table>
<thead>
<tr>
<th>1st round peer-</th>
<th>Quiz-based alg.</th>
<th>Hamer’s alg.</th>
<th>Lauw’s alg.</th>
<th>Naive averages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. abs. bias</td>
<td>5.771</td>
<td>6.479</td>
<td>6.705</td>
<td></td>
</tr>
<tr>
<td>review</td>
<td>RMSE</td>
<td>6.778</td>
<td>7.602</td>
<td>7.854</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td></td>
<td>Avg. abs. bias</td>
<td>2.971</td>
<td>2.961</td>
<td>3.189</td>
</tr>
<tr>
<td></td>
<td>RMSE</td>
<td>4.042</td>
<td>3.820</td>
<td>4.131</td>
</tr>
</tbody>
</table>

### 5.2.3 Conclusion

It is certain that using second-round peer-review records will get aggregated grades with higher accuracy than using first-round peer-review records. Since during second-round peer-review, almost all artifacts are final versions and for vary-rubric-by-round assignment, summative rubric can represent the overall quality of each artifact better. The performance of three reputation algorithms is uncertain. But in most cases, Hamer’s and Lauw’s algorithm perform better than the Quiz-based algorithm and all reputation algorithms perform better than naive averages of peer-review grades.

### 5.3 Experiment 3 – Can reputation be taken from one assignment to a similar one?

#### 5.3.1 Design

The Hamer-expert algorithm can be used only when expert grades are available. But using Hamer-expert algorithm can be a very good practice when acting on writing assignment 1a and 1b in CSC 517 fall 2015 course. In total, the two assignments are almost identical. Both of them are team-based assignments and each team can have at most two team members. Students need to write for Wikipedia on object-oriented related topics. In most of these cases, Wikipedia has topic-related pages with limited content or even does not have those pages. And all the work is done in the Wikipedia sandbox. The review rubrics used in the two
writing assignments are same. But there are some differences between writing assignment 1a and 1b. One is that topics in these two assignments are different, and the other one is that writing assignment 1b will not start till writing assignment 1a is finished. In this case, it is possible that students can do peer-reviews for both assignments.

Since writing assignment 1a and 1b are close enough, there is a reason to believe that the performance of peer reviewers on these two assignments are interlinked. Hence an experiment is designed to check whether the aggregated grades with initial reputation values calculated from writing assignment 1a can do a better job than aggregated grades with initial reputation values equal to 1.

![Diagram]

*Figure 5.3 Aggregated grades calculation procedure of experiment 3.*

The procedure for this experiment is described in Figure 5.3. Firstly, collect expert grade of each artifact from writing assignment 1a, then use the Hamer-expert algorithm to obtain the initial reputation value for each peer reviewer. Secondly, run Hamer’s algorithm with initial reputation values coming from writing assignment 1a to update reputation values for each peer reviewer of writing assignment 1b. After that, calculating the aggregated grade of each artifact based on the peer-review records and newly updated reputation values. At the same time, running Hamer’s algorithm with all initial reputation values equal to 1, and following the same procedure to compute another set of aggregated grade. Finally, comparing these two sets of aggregated grades with expert grades to check which one is more accurate.
There is one potential problem in this experiment. That is, only second-round peer-review records were chosen. Nevertheless, not all students did both second-round peer-reviews in writing assignment 1a and 1b showing in Figure 5.4, so initial reputation values of some students are unavailable. There are two potential solutions to handle this problem. The first way is to assign initial reputation values equal to 1 for those students whose initial reputation values are not available. It is because they did not finish second-round peer-review in writing assignment 1a and only finished that in writing assignment 1b (Part 2, excluding Part 3 in Figure 5.4). The second one is only to consider students who did both second-round peer-reviews in writing assignments (Part 3 in Figure 5.4). It is because different sets of aggregated grades are comparable only if peer reviewers come from the same set of objects [6]. And if the proportion of students who finished second-round peer-reviews for both assignments is large enough, the final results are considered to be reliable.

![Venn diagram of students’ peer-review relationship on writing assignments.](image)

After doing this experiment, three sets of grades can be compared with each other. They are listed below (two sets of aggregated grades are based on Hamer’s algorithm):

1. Aggregated grades with initial reputation values equal to 1;
2. Aggregated grades with initial reputation values calculated from writing assignment 1a;
5.3.2 Results analysis

Only 12 out of 95 students finished second-round peer-reviews for both writing assignments. After implementing the first solution, the final reputation values are the same as those calculated when initial reputation values are equal to 1. The reason why is that the number of students who completed second-round peer-reviews for both writing assignments is quite limited. And after assigning initial reputation values of the other 83 students to one, Hamer’s algorithm will reach the same fixed point as when all initial reputation values are equal to 1, because Hamer’s algorithm will iterate many times until reaching a convergence stage. If two sets of inputs are similar enough, the results will stand a good chance of reaching the same fixed point [5].

Table 5.6 Experiment 3 results from Hamer’s algorithm with different initial reputation values acting on second-round peer-review records from writing assignment 1b fall 2015.

<table>
<thead>
<tr>
<th>Writing assignment 1b, fall 2015</th>
<th>Aggregated grades with init. repu. equal to 1</th>
<th>Aggregated grades with init. repu. from writing assgt. 1a</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bias range</td>
<td>[-8.464, 11.665]</td>
<td>[-7.775, 11.769]</td>
</tr>
<tr>
<td>Avg. absolute bias</td>
<td>4.520</td>
<td>6.043</td>
</tr>
<tr>
<td>RMSE</td>
<td>6.047</td>
<td>6.919</td>
</tr>
</tbody>
</table>

The result of implementing the second way (only to consider students who did both second-round peer-reviews in writing assignments) is shown in Table 5.6. The bias ranges of these two sets are close to each other. However, average absolute bias and RMSE of the first set (aggregated grades with initial reputation values equal to 1) is smaller than those of the second one, which is contrary to expectation. The reason behind this can be that aggregated grades only come from 12 peer reviewers’ records, which are much less than the whole number of students enrolled in this course. Since students can decide to do second-round peer-review or not by themselves, these students are randomly selected. And it is obvious
that they cannot represent the review status of whole class very well. Since Lauw’s algorithm has similar mechanism as Hamer’s, it also cannot obtain reliable results in this situation.

5.3.3 Conclusion

The precondition for using initial reputation values is that there are enough peer reviewers who finished the peer-review of each assignment. There are two solutions to deal with those peer reviewers who did not meet the requirement. The first one is to set unavailable initial reputation values to ones. This solution is suitable in the situation when only a few students’ initial reputation values are unavailable. Otherwise the reputation values will be the same as those calculated based on all initial reputation values equal to 1 because Hamer’s and Lauw’s algorithm will keep iterating until a convergence occurs. The second solution is only to consider those peer reviewers who met the requirement. Since it is voluntary for peer reviewers to do second-round peer-review, the number of those peer reviewers who met the requirement should be large enough to make the them representative.

5.4 Experiment 4 – Can reputation be taken from one assignment to a different one?

5.4.1 Design

The second experiment focuses on making reputation values calculated by writing assignments and program 1 as initial inputs to obtain the aggregated grades of OSS (Open Source Software) project. All three assignments come from CSC 517 fall 2015 course. In order to avoid the problem occurred in the first experiment, merging writing assignment 1a and 1b can be a good choice because their settings are close enough. But differ from writing assignments, program 1 and OSS project are both coding assignments. Hence writing assignment 1a and 1b are considered as same kind of assignments (writing assignment); program 1 and OSS project are regard as same kind of assignments (coding assignment).
There are three hypotheses for this experiment:

1. Aggregated grades calculated by reputation algorithms are more accurate than naive averages of peer-review grades;
2. The initial reputation values from the same kind of assignments can improve the accuracy of aggregated grades and improve on setting initial reputation values to 1;
3. Using initial reputation values from a different kind of assignment will decrease the accuracy of aggregated grades and perform worse than setting initial reputation values to 1.

Figure 5.5 Aggregated grades calculation procedure of experiment 4.

The procedure of this experiment is described in Figure 5.5. At first, we collect expert grades for all artifacts from writing assignments and project 1. Next, using the Hamer-expert algorithm to obtain the initial reputation value for each peer reviewer separately. Then implementing Hamer’s algorithm with initial reputation values obtaining from writing assignments and program 1 to update reputation value of each peer reviewer from OSS project respectively. After that, calculating the aggregated grades for each artifact based on the peer-review records and newly updated reputation values. At the same time, using Hamer’s algorithm to act on peer-review records coming from OSS project with all initial reputation values equal to 1, and following the same procedure to compute another set of aggregated grades. In order to verify the first hypothesis, naive average grade of each artifact
is also calculated. Finally, we compare these four sets of aggregated grades with expert grades to determine which one is more accurate.

The same problem occurs; that is, not all students finished second-round peer-reviews for the three assignments. The first workaround is to assign initial reputation values from writing assignments to 1 for those students who did not finish second-round peer-review in writing assignments but finished them for the OSS project (Part 3, excluding Part 5, 7 in Figure 5.6); and to assign initial reputation values from program 1 to ones for those students who did not finish program 1 reviews, but finished them for the OSS project (Part 3, excluding part 6 and 7 in Figure 5.6). The second solution is only to consider students who finished all 3 second-round peer-reviews. (Part 7 in Figure 5.6).

Both writing assignments and OSS project are vary-rubric-by-round assignments. As mentioned above, second-round peer-review using summative rubric can represent the overall quality of each artifact better. For program 1, it had two rounds of peer-review with the same rubric. So program 1 is not a vary-rubric-by-round assignment. During the class, the instructor encouraged students to do second-round peer-review. The reason why is that after first-round peer-review, authors may modify their artifacts according to the comments they received. So doing second-round peer-review can represent the latest quality of each artifact. And in this experiment, only second-round peer-reviews of all three assignments counted.
After doing this experiment, four sets of aggregated grades are generated based on Hamer’s and Lauw’s algorithms and compared with expert grades. All sets of grades are listed below:

1. Aggregated grades with initial reputation values equal to 1;
2. Aggregated grades with initial reputation values calculated from writing assignments;
3. Aggregated grades with initial reputation values calculated from program 1;
4. Naive averages calculated from peer-review grades;
5. Expert grades.

**5.4.2 Results analysis**

There are 49 out of 95 students who completed all three second-round peer-reviews. At this time, the first solution is still not available since only half of students have the initial reputation values.

If only considering Part 7 in Figure 5.6, the proportion of students doing peer-review is reasonable and the final results can also be reliable. The results second method is presented in Figure 5.7.
Figure 5.7 Reputation algorithm experiment with three sets of aggregated grades and expert grades sorted ascendingly. The x-axis represents identifiers of different artifacts; y-axis represents the grades in the range between 70 and 100.

In Figure 5.7, the red line represents the expert grades in ascending order. The orange, blue and green line represents three sets of aggregated grades separately. In most cases, these three lines overlap a lot, which means these three sets of aggregated grades are quite close to each other. However, for a few artifacts, such as artifact 24393, aggregated grades with initial reputation values calculated from program 1 are closer to expert grades than other two sets.

Table 5.7 Experiment 4 results from Hamer’s algorithm with different initial reputation values acting on second-round peer-review records from OSS project fall 2015.

<table>
<thead>
<tr>
<th>OSS project, fall 2015</th>
<th>Hamer’s alg. with init. repu. equal to 1</th>
<th>Hamer’s alg. with init. repu from writing assgts.</th>
<th>Hamer’s alg. with init. repu from program 1</th>
<th>Naive averages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. abs. bias</td>
<td>6.057</td>
<td>6.374</td>
<td>5.957</td>
<td>8.245</td>
</tr>
<tr>
<td>RMSE</td>
<td>7.528</td>
<td>8.053</td>
<td>7.456</td>
<td>9.240</td>
</tr>
</tbody>
</table>
In Table 5.7, the bias range of the third set (aggregated grades with initial reputation values calculated from program 1) is the smallest among all four sets and the average absolute bias of third set is also the best one. By contrast, the second set (aggregated grades with initial reputation values calculated from writing assignments) of data is the worst-performing among all three sets of aggregated grades calculated by reputation algorithm. And naive averages perform worse than any other aggregated grades. The data shown above can verify all three hypotheses.

The results using Lauw’s algorithm with same data are presented in Table 5.8, which can also verify three hypotheses mentioned above. And according to these three metrics, Hamer’s algorithm performs better than Lauw’s algorithm in this case.

**Table 5.8** Experiment 4 results from Lauw’s algorithm with different initial reputation values acting on second-round peer-review records from OSS project fall 2015.

<table>
<thead>
<tr>
<th>OSS project, fall 2015</th>
<th>Lauw’s alg. with init. repu equal to 1</th>
<th>Lauw’s alg. with init. repu from writing assgts</th>
<th>Lauw’s alg. with init. repu from program 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. abs. bias</td>
<td>7.495</td>
<td>7.596</td>
<td><strong>7.411</strong></td>
</tr>
<tr>
<td>RMSE</td>
<td>8.653</td>
<td>8.707</td>
<td><strong>8.598</strong></td>
</tr>
</tbody>
</table>

However, the max absolute biases of all six sets of aggregated grades (each algorithm has three sets of aggregated grades) are all quite high (more than 15 points), which means there still exist some obvious differences between expert grades and aggregated grades. Hence there must be some other issues also influencing the aggregated grades and not considered into algorithms, such as poorly designed rubrics, insufficient peer-review training, etc. Then I figured out that different ways of grading can help to explain these results. Since I was one of the teaching assistant of CSC 517 fall 2015 course, I found that teaching staff
would use different ways to grade program 1 and OSS project. Program 1 has two rounds of peer-reviews with the same rubric. The rubric has 14 questions (see in Appendix A), which means that each question will affect approximately 7% of the final grade because each question in this rubric has the same weight and Expertiza uses naive average value to compute the final grade. And the most important thing is that teaching staff followed this rubric strictly while grading. OSS project is an assignment with vary-rubric-by-around feature. Its second-round summative rubric has 7 questions (see in Appendix B), which means each question will affect more than 14% of final grade. And teaching staff did not follow the summative rubric very well while grading.

For example, one OSS artifact got 91 for expert grade but only got approximately 75 for aggregated grade. The final comments given by teaching staff is

“Well, from the video they did the thing we expect them to do, but their tests are failing, and they should have fixed them.”

and that team got 91 for final grade, which is the sixth lowest score out of 31 artifacts. In summative rubric, there is a test-related question

“If it is an Expertiza project, check the pull request. Did the build pass in Travis CI? Was there any conflict that must be resolved? You can check those on the pull request on GitHub. Ignore this question if it is not an Expertiza project.”

According to 13 valid peer-review records, most peer reviewers were able to figure out this problem. And the average score of this question is 2.16 out of 5, which means that 8.11 points will be deducted from total score since the code did not pass the TravisCI.

And during grading, teaching staff almost did not consider another question in this rubric. That is

“Check the commits. Was new code committed during the 2nd round?”
Since this team did not commit new code or did not committed promptly, the average of this question is 3.58 out of 5, which means 4.06 points will be deducted from total score. So only these two questions have already deducted more than 12 points from total score. And two peer reviewers who have relatively high initial reputation values gave quite low scores to this artifact, 71 and 69 separately. (The initial reputation values of these two peer reviewers are 2.96 and 1.95, which were calculated from program 1 using Hamer-expert algorithm. The average of all initial reputation values is 2.26.)

What’s more, only 3 out of 31 artifacts got the grades lower than 90 and this one got 91. It is obvious that teaching staff also considered it is not a quite successful artifact. However, a relatively tolerant grades still be assigned to this team. So it can be the reason why there is a large difference between expert grades and aggregated grades and a new grading method specific to peer-review grades may help to solve this problem.

5.4.3 Conclusion

Aggregated grades calculated by reputation algorithms are more accurate than naive averages of peer-review grades; the initial reputation values from the same kind of assignments can improve the accuracy of aggregated grades and better than making ones as initial reputation values; conversely, if the initial reputation values are taken from a different kind of assignments, the accuracy of aggregated grades will decrease and be even worse than making initial reputation values all 1s.

At the time of grading, it is possible that teaching staff did not follow the rubric strictly. There are several potential solutions. The first one is changing the rubric and make it closer to the criteria of teaching staff. Secondly, using a new formula to calculate the peer-review grades can be another approach to solve this problem. The third way is letting teaching staff make adjustment on their scoring approach, although it may be hard to implement.
5.5 Experiment 5 – Can reputation be taken from calibration to real assignments?

5.5.1 Design

In normal CSC 517 course, there are five assignments, that is writing assignment 1a and 1b, program 1, OSS project and final project. According to the first two experiments, initial reputation values calculated from former assignments can lead to more accurate aggregated grades than initial reputation values equal to 1. The precondition of this conclusion is that enough peer reviewers should complete certain round of peer-review.

Nevertheless, for writing assignments there is no way to calculate the initial reputation values. This is because they are the first two assignments in this course. In this case, using peer-review records from calibration assignment to calculate the initial reputation values can be a good workaround. And this method bases on the hypothesis that peer reviewer who has a good reputation value in calibration assignment also does a good job in real assignments.

This experiment only focuses on students who finished the calibration process, second-round peer-review and took related quizzes. In order to check the effect of calibration, a new
experiment is designed to make different algorithms act on peer-review records from writing assignments in CSC 517 Spring 2016 course. The procedure of this experiment is presented in Figure 5.8. The first two sets of aggregated grades are calculated by the Quiz-based algorithm and Hamer’s algorithm. The third set is computed by Hamer’s algorithm with initial reputation values not equal to 1. These initial reputation values are the results of running the Hamer-expert algorithm on expert grades and peer-review records of calibration assignments. Finally, we compare different sets of aggregated grades with expert grades. After doing this experiment, four sets of aggregated grades are generated to compare with expert grades. They are listed below:

1. Aggregated grades calculated by Quiz-based algorithm;
2. Aggregated grades with initial reputation values equal to 1;
3. Aggregated grades with initial reputation values calculated from calibration assignment;
4. Naive averages calculated from peer-review grades;
5. Expert grades.

### 5.5.2 Results analysis

There are 45 out of 57 students who completed calibration process, second-round peer-review and took relative quizzes, which is almost 80% of all students enrolled in this class.

Results of experiment 5 are shown in Figure 5.9. The red line represents the expert grades in ascending order. The green line, blue line and orange represent different sets of aggregated grades respectively. Aggregated grades calculated from the Quiz-based algorithm tend to have less accuracy than other two sets of aggregated grades. And aggregated grades calculated from initial reputation values equal to 1 and those calculated from initial reputation values from calibration assignment are quite close to each other. However, for some artifacts the orange line is slightly closer to red line than blue line.
Figure 5.9 Reputation algorithm experiment with three sets of aggregated grades and expert grades sorted ascendingly. The x-axis represents identifiers of different artifacts; y-axis represents the grades in the range between 85 and 100.

Table 5.9 presents the results of experiment 5 in detail. For bias range, three sets of aggregated grades perform close to each other. Nevertheless, with regard to average absolute bias and RMSE, the third set (aggregated grades with initial reputation values calculated from calibration assignment) performs best and the first set (aggregated grades calculated by Quiz-based algorithm) performs worst among three sets of aggregated grades. And naive averages perform worse than any set of aggregated grades on all three metrics. What’s more, aggregated grades with initial reputation values calculated from calibration assignment just perform slightly better than those with initial reputation values equal to 1. One potential reason is that the differences between aggregated grades and expert grades are too small to improve greatly.
Table 5.9 Experiment 5 results from Quiz-based algorithm, Hamer’s algorithm with different initial reputation values and naïve averages acting on second-round peer-review records from writing assignment spring 2016.

<table>
<thead>
<tr>
<th></th>
<th>Quiz-based algorithm</th>
<th>Hamer’s alg. with initial repu equal to 1</th>
<th>Hamer’s alg. with init. repu from calibration</th>
<th>Naïve averages</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bias range</strong></td>
<td>[-7.557, 9.960]</td>
<td>[-6.745, 10.244]</td>
<td>[-6.775, 10.185]</td>
<td>[-8.505, 10.128]</td>
</tr>
<tr>
<td><strong>Avg. abs. bias</strong></td>
<td>2.971</td>
<td>2.832</td>
<td>2.754</td>
<td>3.258</td>
</tr>
<tr>
<td><strong>RMSE</strong></td>
<td>4.042</td>
<td>3.716</td>
<td>3.646</td>
<td>4.269</td>
</tr>
</tbody>
</table>

5.5.3 Conclusion

Calibration assignment can offer initial reputation values for those assignments which are at the very beginning of one course. Aggregated grades with initial reputation values calculated from calibration assignment performs the best. And no matter which kinds of initial reputation values, Hamer’s algorithm performs better than Quiz-based algorithm. However, initial reputation values from calibration assignment did not help to improve the accuracy of aggregated grades very much. So further experiments are needed to check the effect of calibration.
Chapter 6  Future Work

Despite the success of reputation web service’s deployment, there are still aspects that can be improved.

Firstly, more reputation algorithms are need. For now, there are only five algorithms implemented in web service. And since both Hamer’s algorithm and Lauw’s algorithm need iterate several times until convergence occurs, it can be possible that the results are just locally optimal solution, instead of the globally optimal solution. For Hamer-expert algorithm and Lauw-expert algorithm, they can be use only expert grades are available. And the scope of Quiz-based algorithm is also quite limited. Therefore, more algorithms with different calculation methods are needed. I should consider to use some data mining methods, such as K-nearest neighbor, decision tree and Artificial Neural Network, to build some models. Since Expertiza and other peer-review systems have already existed for several years, we have enough data to do model training and then use these models to do grades prediction.

What’s more, all algorithms mentioned above are all rating-based. Some other peer-review systems, such as Critviz\(^2\) and Mobius SLIP\(^3\), measure the qualities of peer-reviews based on ranking. I can also build some ranking-based algorithms to increase the compatibility of reputation web service. Then more peer-review systems can be connected to our reputation web service.

Secondly, more experiments are required to further inspect the performance of different reputation algorithms and try to find the reason why there still exist large gaps (differences, larger than 10 points) between expert grades and aggregated grades even in best sets of aggregated grades. And for calibration assignment, I just did one experiment and its effect still unclear.

\(^2\) https://critviz.com/
\(^3\) http://www.mobiuslip.com/
Moreover, teaching staff can design their own experiments no matter what kinds of courses they teach. Since it is easy to make one peer-review system communicate to reputation web service (only need to build a data wrapper), different teaching staff can make use of web service without knowledge of programming. Maybe after several experiments, they can find one or two useful patterns to calculate the accurate aggregated grades. These patterns can be various, such as using students’ GPA to become initial reputation values or using reputation values of some related courses to become inputs of algorithms, etc. Once a reliable pattern is found, teaching staff do not need to check all peer-review records every time, instead they only need to check subset of them and give marks according to aggregated grades.

Thirdly, reputation web service needs more security measures. Although I have already encrypted the sensitive data communicating between client side and server side. Hackers can still keep sending requests to web service and make a Denial-of-service attack. Therefore, a token-based authentication is needed. Basically, only authorized client systems can get a token and embed it in request message. And web service will execute operations only it recognizes a correct token. Also, the token should also be changed after some time period.
REFERENCES


## Table 0.1 Formative rubric of writing assignments in CSC 517 fall 2015 and spring 2016

<table>
<thead>
<tr>
<th>Question</th>
<th>Type</th>
<th>Weight</th>
<th>Score Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>List the unfamiliar terms used in this wiki. Are those unfamiliar terms well defined or linked to proper references?</td>
<td>Criterion</td>
<td>1</td>
<td>(neither defined nor linked)</td>
</tr>
<tr>
<td>Rate the overall readability of the article. Explain why you give this score.</td>
<td>Criterion</td>
<td>1</td>
<td>(not readable and confusing)</td>
</tr>
<tr>
<td>Rate the English usage. Give a list of spelling, grammar, punctuation mistakes or language usage mistakes you can find in this wiki (e.g. ruby on rails -&gt; Ruby on Rails).</td>
<td>Criterion</td>
<td>1</td>
<td>(terrible English usage)</td>
</tr>
<tr>
<td>List any related terms or concepts for which the writer failed to give adequate citations and links. Rate the helpfulness of the citations.</td>
<td>Criterion</td>
<td>1</td>
<td>(more citations are needed)</td>
</tr>
<tr>
<td>Rate how logical and clear the organization is. Point out any places where you think that the organization of this article needs to be improved.</td>
<td>Criterion</td>
<td>1</td>
<td>(terrible organization)</td>
</tr>
<tr>
<td>Randomly pick some sentences or paragraphs and search for it with a search engine. List any sources that may infringe copyrights.</td>
<td>Criterion</td>
<td>1</td>
<td>(many copied content spotted)</td>
</tr>
<tr>
<td>Checklist</td>
<td>SectionHeader</td>
<td></td>
<td>—</td>
</tr>
<tr>
<td>The discussion is appropriate and reasonable, i.e., not too easy or too difficult for your peers to follow.</td>
<td>Checkbox</td>
<td></td>
<td>Checked/Unchecked</td>
</tr>
<tr>
<td>Most of the sources are current (less than 5 years old).</td>
<td>Checkbox</td>
<td></td>
<td>Checked/Unchecked</td>
</tr>
<tr>
<td>This article is up to date.</td>
<td>Checkbox</td>
<td></td>
<td>Checked/Unchecked</td>
</tr>
<tr>
<td>Taken together the sources represent a good balance of potential references for this topic</td>
<td>Checkbox</td>
<td></td>
<td>Checked/Unchecked</td>
</tr>
<tr>
<td>Short Response</td>
<td>SectionHeader</td>
<td></td>
<td>—</td>
</tr>
<tr>
<td>Please make a comment about the sources. Explain how the author can improve the use of sources in the article.</td>
<td>TextArea</td>
<td></td>
<td>—</td>
</tr>
<tr>
<td>What other sources or perspectives might the author want to consider?</td>
<td>TextArea</td>
<td></td>
<td>—</td>
</tr>
<tr>
<td>Give compliments for the article. Separate them with line breaks.</td>
<td>TextArea</td>
<td></td>
<td>—</td>
</tr>
<tr>
<td>Give suggestions for the article. Separate them with line breaks.</td>
<td>TextArea</td>
<td></td>
<td>—</td>
</tr>
</tbody>
</table>
## Appendix B

**Table 0.2** Summative rubric of writing assignments in CSC 517 fall 2015 and spring 2016

<table>
<thead>
<tr>
<th>Question</th>
<th>Type</th>
<th>Weight</th>
<th>Score Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organization: how logical and clear is the organization?</td>
<td>Criterion</td>
<td>2</td>
<td>(Terrible organization) 0 to 5 (Very logical and clear)</td>
</tr>
<tr>
<td>Originality: If you found any plagiarism in round 1, has it been removed? Then, randomly pick some sentences or paragraphs and search for them with a search engine. Describe any text that may infringe copyrights.</td>
<td>Criterion</td>
<td>2</td>
<td>(Several places of plagiarism spotted) 0 to 5 (No plagiarism spotted)</td>
</tr>
<tr>
<td>Clarity: Are the sentences clear, and non-duplicative? Does the language used in this artifact simple and basic to be understood?</td>
<td>Criterion</td>
<td>2</td>
<td>(Terrible English usage) 0 to 5 (Good English usage)</td>
</tr>
<tr>
<td>Coverage: does the artifact cover all the important aspects that readers need to know about this topic? Are all the aspects discussed at about the same level of detail?</td>
<td>Criterion</td>
<td>2</td>
<td>(Not agree) 0 to 5 (Strong agree)</td>
</tr>
<tr>
<td>Definitions: are the definitions of unfamiliar terms clear and concise? Are the definitions adequately supported by explanations or examples?</td>
<td>Criterion</td>
<td>2</td>
<td>(Several definitions are missing or incomplete) 0 to 5 (Strong agree)</td>
</tr>
<tr>
<td>References: do the major concepts have citations to more detailed treatments? Are there any unavailable links?</td>
<td>Criterion</td>
<td>2</td>
<td>(Many more references should be added) 0 to 5 (Strong agree)</td>
</tr>
<tr>
<td>Did the authors revise their work in accordance with your suggestions?</td>
<td>Criterion</td>
<td>1</td>
<td>(Not agree) 0 to 5 (Strong agree)</td>
</tr>
</tbody>
</table>
## Appendix C

### Table 0.3 Peer review rubric of program 1 in CSC 517 fall 2015

<table>
<thead>
<tr>
<th>Question</th>
<th>Type</th>
<th>Weight</th>
<th>Score Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Please review the code on Git. How well does the code follow &quot;good Ruby and Rails coding practices&quot;?</td>
<td>Criterion</td>
<td>1</td>
<td>0 to 5</td>
</tr>
<tr>
<td>Is the user interface intuitive and easy to use? If not, is it well described in the README file?</td>
<td>Criterion</td>
<td>1</td>
<td>(UI is hard to use and there is no instruction) 0 to 5</td>
</tr>
<tr>
<td>Check the basic functionalities for admins below and see if they all work: 1) Can an admin log in? 2) Can (s)he edit his profile? 3) Can (s)he log out?</td>
<td>Criterion</td>
<td>1</td>
<td>(None of them work or are testable) 0 to 5 (All of them work)</td>
</tr>
<tr>
<td>Admins should have higher authorities over ordinary library members. Check the functionalities for admin below and see if they all work: 1) Can an admin create other admins? 2) Can an admin view the list of other admins? 3) Can an admin view the details (except password) of another admin? 4) Can an admin delete other admins other than himself and the preconfigured admin?</td>
<td>Criterion</td>
<td>1</td>
<td>(None of them work or are testable) 0 to 5 (All of them work)</td>
</tr>
<tr>
<td>Admin should be able to view and edit the profiles of books. Check the functionalities for admin below and see if they all work: 1) Can an admin add books? 2) Can an admin view the list of all the books? 3) Can an admin view the details of a book? 4) Can an admin edit the details of a book? 5) Can an admin change its status i.e., check out or return a book on behalf of a library member?</td>
<td>Criterion</td>
<td>1</td>
<td>(None of them work or are testable) 0 to 5</td>
</tr>
<tr>
<td>Admin should be able to view the history of both books and library members. Check the functionalities for admin below and see if they all work: 1) Can an admin view the checkout history of a book? 2) Can (s)he view the checkout history of a library member?</td>
<td>Criterion</td>
<td>1</td>
<td>(Neither of them work or testable) 0 to 5 (Both of them work)</td>
</tr>
<tr>
<td>This project should support the basic functionalities for library members. Check the functionalities for library members below and see if they all work: 1) Can a user sign up to become a library member? 2) Can (s)he log in after sign up? 3) Can (s)he edit his profile? 4) Can (s)he log out?</td>
<td>Criterion</td>
<td>1</td>
<td>(None of them work or are testable) 0 to 5 (All of them work)</td>
</tr>
<tr>
<td>The library members should be able to perform basic actions on the books. Check the functionalities for library members below and see if they all work: 1) Can a library member search for books (with ISBN, title, author, description or status)? 2) Can a library member view the details of a book? 3) Can a library member check out an available book? 4) Can (s)he return it? 5) Can a library member view his/her own checkout history?</td>
<td>Criterion</td>
<td>1</td>
<td>(None of them work or are testable) 0 to 5 (All of them work)</td>
</tr>
<tr>
<td>The admin should be able to delete an unchecked book. Check the functionalities for admin below: 1) Can an admin delete an unchecked book? 2) See what happens to the checkout history of the users who once checkout that book. Is it intuitive or described in the README file?</td>
<td>Criterion</td>
<td>1</td>
<td>0 to 5</td>
</tr>
<tr>
<td>The admin should be able to delete an user who does not have any book checkout currently. Check the functionalities for admin below: 1) Can an admin delete a library member who is not holding any book? 2) See what happens to the checkout history of the books that were checked out by him/her. Is it intuitive or described in the README file?</td>
<td>Criterion</td>
<td>1</td>
<td>0 to 5</td>
</tr>
<tr>
<td>Special test case A: The system should handle the scenario that admins delete a book which is currently checked out. Please check: 1) Is this case considered and handled properly in this project? 2) Is this scenario described in the README file?</td>
<td>Criterion</td>
<td>1</td>
<td>(Neither of them work in intuitive way or are testable) 0 to 5 (Both of them work in intuitive ways)</td>
</tr>
<tr>
<td>Special test case B: The system should handle the scenario that admins delete users who are currently holding books. Please check: 1) Is this case considered and handled properly in this project? 2) Is this scenario described in the README file?</td>
<td>Criterion</td>
<td>1</td>
<td>(Neither of them work in intuitive way or are testable) 0 to 5 (Both of them work in intuitive ways)</td>
</tr>
</tbody>
</table>
Are the extra credit features implemented? Please check the functionalities below: 1) Can a library member register to receive a notification email when a book becomes available? 2) Can a library member submit suggestions for new books? 3) Can admins view and approve the suggestions? 4) Can Admins edit book details during the approval process? 5) Can the suggested books be seen after approval?

<table>
<thead>
<tr>
<th>Criterion</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>None of those are implemented.</td>
<td>0 to 5 (All of those are implemented.)</td>
</tr>
</tbody>
</table>

Has this team thoroughly tested at least one model and one controller?

<table>
<thead>
<tr>
<th>Criterion</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not tested at all</td>
<td>0 to 5 (100% test coverage)</td>
</tr>
</tbody>
</table>

Did the team kept on submitting their changes throughout the project?

<table>
<thead>
<tr>
<th>Criterion</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Just a few commits towards the end</td>
<td>0 to 5 (Version control is used continuously)</td>
</tr>
</tbody>
</table>
# Appendix D

## Table 0.4 Formative rubric of OSS project in CSC 517 fall 2015 and spring 2016

<table>
<thead>
<tr>
<th>Question</th>
<th>Type</th>
<th>Weight</th>
<th>Score Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code</td>
<td>SectionHeader</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Check the pull request; does the build pass?</td>
<td>Checkbox</td>
<td>Checked/Unchecked</td>
<td>—</td>
</tr>
<tr>
<td>(1) Have tests been added to the project? (2) Do the new tests pass? Leave a comment if any of them fail or are missing. (3) Has the test coverage increased?</td>
<td>Criterion</td>
<td>1</td>
<td>(No tests added; existing tests fail) 0 to 5 (Tests added; all tests pass)</td>
</tr>
<tr>
<td>Look at the newly-added code in the pull request. Check the variable, method, and class names. List any name(s) that are not reasonable or suggestive of the functionality.</td>
<td>Criterion</td>
<td>1</td>
<td>(Major changes on variable names needed) 0 to 5 (Names are intuitive)</td>
</tr>
<tr>
<td>Check the code contributed by the author. Comment if (1) some of the functions are too long; (2) some of the code should be extracted into separate methods; (3) more comments are needed, because you have trouble following the code; (4) the code does not follow the Ruby Style Guide, or (5) you find any other code that “smells” or is not DRY.</td>
<td>Criterion</td>
<td>1</td>
<td>(Smelly: long methods/ limited comments/ bad coding style) 0 to 5 (Nice code)</td>
</tr>
<tr>
<td>Manually test the author's work. Keep in mind of the edge cases if you see any. Do they work as intended? If you find any case for which the code does not work, please describe in the comment box in enough detail.</td>
<td>Criterion</td>
<td>1</td>
<td>(Cannot figure out how to test/the code doesn't work) 0 to 5 (Works well even in edge cases)</td>
</tr>
<tr>
<td>Writeup</td>
<td>SectionHeader</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Read the writeup; check whether it clearly and adequately indicates what functionality the work is related to.</td>
<td>Criterion</td>
<td>1</td>
<td>(Missing or doesn't explained well) 0 to 5 (Explained with enough details)</td>
</tr>
<tr>
<td>Does the writeup indicate how to test the new code from UI?</td>
<td>Criterion</td>
<td>1</td>
<td>(Missing or not able to follow) 0 to 5 (Explained with enough details)</td>
</tr>
<tr>
<td>Does the writeup explain how and why the authors did the work the way they did? If they should have used certain design principles or patterns, did they use them correctly? Comment on anything that is missing or hard to follow.</td>
<td>Criterion</td>
<td>1</td>
<td>(Missing or doesn't explained well) 0 to 5 (Explained with enough details)</td>
</tr>
<tr>
<td>If you are reviewing a testing team’s work (<a href="https://example.com">E1572. Feature Test for Assignment Creation via instructor; E1573. Unit Test for Assignment model; E1574. Feature Test for Assignment Submission</a>). Please git clone their repo and run the test on your local machine, you should upload a screenshot of the running result.</td>
<td>UploadFile</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>
### Appendix E

**Table 0.5** Summative rubric of OSS project in CSC 517 fall 2015 and spring 2016

<table>
<thead>
<tr>
<th>Question</th>
<th>Type</th>
<th>Weight</th>
<th>Score Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall, how well was the code written. If you found problems in the code in the first round (bad names, long or complicated functions, lack of comments, bad code style, DRY problems...), have the authors improved the code accordingly?</td>
<td>Criterion</td>
<td>1</td>
<td>(Major improvements needed) 1 to 5 (Well-written)</td>
</tr>
<tr>
<td>Check the commits. Was new code committed during the 2nd round?</td>
<td>Criterion</td>
<td>1</td>
<td>(No new commits) 1 to 5 (Many commits)</td>
</tr>
<tr>
<td>IF it is an Expertiza project, check the pull request. Did the build pass in Travis CI? Was there any conflict that must be resolved? You can check those on the pull request on GitHub. Ignore this question if it is not an Expertiza project.</td>
<td>Criterion</td>
<td>1</td>
<td>(Strong disagree) 1 to 5 (Strong agree)</td>
</tr>
<tr>
<td>Did the team add test cases? Did the coverage increase? How well do the newly added tests cover the range of this project?</td>
<td>Criterion</td>
<td>1</td>
<td>(No new test or tests fail) 1 to 5 (Passing tests added, coverage increases)</td>
</tr>
<tr>
<td>Verify the system operation from the UI. Does it work as intended? If this project is to refactor or fix code, do the features work as they are supposed to? If this project is a testing project, do the tests cover all the scenarios?</td>
<td>Criterion</td>
<td>1</td>
<td>(Major part missing or not work) 1 to 5 (Everything works)</td>
</tr>
<tr>
<td>Overall, how good is the write up? If you found problems with the write-up (lack of explanation of the functionality, lack of explanation of how to check the work...), have the authors improved the write-up accordingly?</td>
<td>Criterion</td>
<td>1</td>
<td>(Major improvements needed) 1 to 5 (Well-written)</td>
</tr>
<tr>
<td>Do you think that this code is ready to be deployed onto the production server (for the corresponding OSS project)? If not, what is your biggest concern? Should the project be redone from scratch using a different approach, or is this a good starting place for a future team to pick up?</td>
<td>Criterion</td>
<td>1</td>
<td>(Should be redone from scratch) 1 to 5 (Ready to be merged)</td>
</tr>
</tbody>
</table>