Task Assignment of Peer Grading in MOOCs

Yong Han(✉), Wenjun Wu, and Yanjun Pu

State Key Laboratory of Software Development Environment, School of Computer Science, Beihang University, Beijing, China
{hanyong,wwj,puyanjun}@nlsde.buaa.edu.cn

Abstract. In a massive online course with hundreds of thousands of students, it is unfeasible to provide an accurate and fast evaluation for each submission. Currently the researchers have proposed the algorithms called peer grading for the richly-structured assignments. These algorithms can deliver fairly accurate evaluations through aggregation of peer grading results, but not improve the effectiveness of allocating submissions. Allocating submissions to peers is an important step before the process of peer grading. In this paper, being inspired from the Longest Processing Time (LPT) algorithm that is often used in the parallel system, we propose a Modified Longest Processing Time (MLPT), which can improve the allocation efficiency. The dataset used in this paper consists of two parts, one part is collected from our MOOCs platform, and the other one is manually generated as the simulation dataset. We have shown the experimental results to validate the effectiveness of MLPT based on the two type datasets.

Keywords: Task assignment · Crowdsourcing · LPT · Peer grading

1 Introduction

Currently MOOCs learning is becoming a popular way to acquire knowledge besides the traditional classrooms. A typical MOOC course contains videos, exercises, online test, peer grading and forums. Such an online course often has thousands of active learners in every study session [9]. When students submit their homework to the course, there are a lot of submissions of each assignment for course staffs to grade. Given the scale of the submissions, if the staffs have to examine all the submissions, they will be overwhelmed by the grading workload so that they can't have sufficient time and energy to focus on other things that will be helpful for students.

Researchers have proposed the methods of peer grading which are similar to crowd sourcing. In the process of peer grading, students are asked to grade their submissions. Therefore, the students are both submitters and the graders, which allows us to exploit this unique feature to solve this peer grading problem. In general, there are two basic problems in peer grading: (1) one is how to assign the submissions to the graders considering the difference of knowledge level between graders, (2) the other is how to aggregate to peer grades to generate a final fair score for each student. In this paper we mainly focus on the first problem.

Z. Bao et al. (Eds.): DASFAA 2017 Workshops, LNCS 10179, pp. 352–363, 2017.
DOI: 10.1007/978-3-319-55705-2_28

In most MOOCs platforms, the systems adopt a simple random assignment approach. Their peer grading servers record the submitted time of an incoming submission and store it in queue. Then the distribution mechanism chooses an idle grader randomly and assign it to the grader without considering the grader's knowledge level. Such a random assignment approach often results in biased grading scores for students because the assignment plan may put graders with a high level of knowledge mastery in the same group to evaluate the same homework assignment. Previous research papers have shown that the reliability of every grader is related to his knowledge skills. The students with good knowledge mastery tend to deliver reliable and accurate grading scores. Therefore, in the uneven distribution plan, some submissions receive very accurate evaluations while the others receive with very inaccurate evaluations. The problem can be avoided if the peer grading server makes grading task allocation with the awareness of the knowledge level of each participating student. The task allocation algorithm should be to distribute students with different knowledge levels evenly to grading groups so that the difference among the average knowledge level in each group could be minimized.

Considering the unique characteristic of peer grading, we choose the method Longest Process Time (LPT) [5] as a heuristic algorithm for grading task assignment. The algorithm is used in parallel systems to deal with task scheduling. It is a static task allocation strategy, so it requires all the tasks are assigned before the processors begin to deal with the tasks. Assigning tasks in peer grading has its unique characteristics compared to other task scheduling problems. A homework submission in the submission set may be assigned to its author if the distribution mechanism is not restricted and this is not allowed in practice. Furthermore, one submission should be graded by several graders and conversely one grader should also grade the same number of submissions.

Based on the algorithm LPT, we propose our algorithm Modified LPT (MLPT) to solve the allocating submissions in peer grading. Most researchers on peer grading focus on developing statistical models to infer accurate scores based on peer grading results. Few work consider the impact of the peer grading task allocation. Our main contribution is the introduction of student knowledge mastery into the design of task allocation algorithm. By developing the student performance evaluation model, our peer grading system can accurately estimate the student knowledge level. And based on the, we redefine the LPT algorithm to adapt the peer grading process to the distribution of student knowledge level.

The rest of the paper is organized as follows: Sect. 2 compares our work with other related efforts. Section 3 presents the overview of our peer grading framework and describes the MLPT algorithm in detail. Sections 4 and 5 present our experimental datasets and analysis results. Discussion and Future work is given in Sect. 6.

2 Related Work

In this paper, before allocating tasks, we detailed analysis the student online performance of learning for accurately estimating the knowledge level of student. We have referred to the thought task assignment in Crowdsourcing and proposed an effective method by compare the previous models. We first review some existing researches for allocating

tasks. Allocating tasks is a long historical issue and appears in many fields such as distributing conference papers or in the parallel systems and so on. It is also one of the core problems of the crowdsourcing.

In the mode of Crowdsourcing, a task is decomposed into multiple small tasks [8] which is assigned to massive online workers. Because of the existing of human error and fraudulent workers, the confident of the collection results which typically contains a lot of noise is usually very low [2, 3]. In order to ensure the reliability of the results, the measure of redundant allocation is typically taken, namely, assigning the same task to multiple workers, and aggregating the multiple results to generate a final one. Here the task allocation is to find a balance between the reliability of the results and the redundancy of the task allocation.

Ul Hassan and Curry [15] proposed a task allocation solution based on the human ability. This method first models the abilities of a worker according to historical data. Similarly, it models the types of workers' abilities that the task required. Then through adaptation, the algorithm allocates the task to the most suitable workers thus obtaining the optimal results with the minimal redundancy. Finally, they adjust the worker model by comparing the completing results to the aggregating result.

Jung and Lease [16] had solved the problem of data sparsity by decomposing matrix based on the research of Umair and Edward, increasing the universality of the method. In addition to model the abilities of the workers based on the historical data, other researchers also considered the demographic factors, education background, and especially the social information to determine the level of the workers' abilities which can allocate the task to the worker accurately.

Furthermore, because the most Crowdsourcing tasks are paid tasks, the capital budget is constraint [4], Karger et al. [17] inspired by the algorithm of propagation of confidence and the algorithm of low rank matrix approximation designed the minimum cost allocation strategy. The results in the strategy make the cost of the spending the least if it meets certain prerequisites of reliability.

Yan et al. [18] proposed an online adaptive task assignment algorithm by ensuring the confidence of the results. The algorithm allocates the subsequent redundancy and the workers according to the required of the task by analyzing the completion of the task time to time and the confidence of the results to make the final results highly reliability.

Piech et al. [6] exploited the confidence estimated to determine what point in the grading process is confident about each submission's score. They simulated grading taking place in rounds. For each round they ran model using the corresponding subset of grades and counted the number of submissions for which they were over 90% confident that they predicted grades were within 10pp of the students' true grade.

Besides, in the area of spatial crowdsourcing, tasks with spatiotemporal constraints are allocated [19–23].

In our work, there is no need to classify the students according to the types. The primary information about students is their historical learning data, thus we exploit some models compute the student knowledge level based on this data. By extracting student features, we using the method of logistic regression to fit the features and this part accounts for the proportion of our work. We also define a hyper-parameter as the redundancy value, which requires us to constantly experiment.

We have done experiments over different redundancy values and different number of students. The results can be seen in Sect. 5. Our another attempt is using the examination scores as the knowledge level, but it does not generate a good effect, the reasons may be that the score is a monotonous factor (Fig. 1).

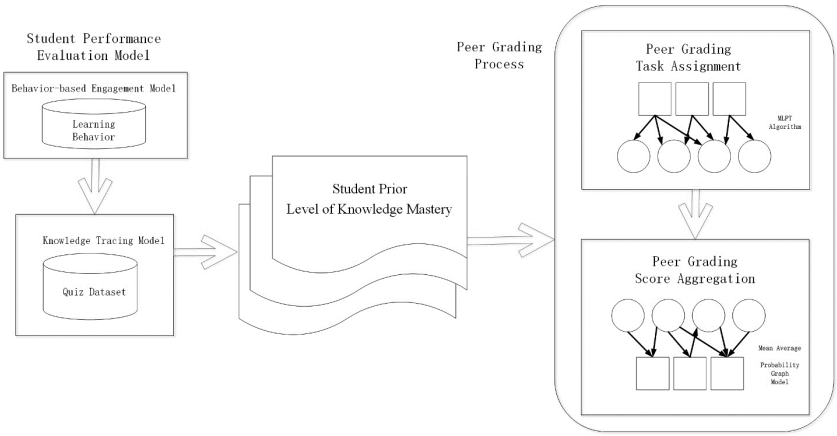

Fig. 1. Peer grading framework.

3 Methods

In this section, we present the overview of the entire peer grading framework and introduce the design of grading task assignment in detail. Figure 2 illustrates the basic framework of our peer grading process. It consists of three major components: the student performance evaluation model, the peer grading task allocator and the score aggregation model. The student performance evaluation model is responsible for computing each student's knowledge level by analyzing each student's behavior in the watching course videos and the performance exercises after the videos. The outcome from the model presents the prior knowledge level of students for the peer grading process to determine the arrangement of peer grading groups. The peer grading process includes two steps: grading task allocation and score aggregation model. In this paper, we introduce the MPLT algorithm to perform task allocation in order to minimize the difference among the students in each grading group. Based on the task assignment plan, students can evaluate other peers' submissions and give scores using their own judgement. As each submission in general receives multiple scores from its graders, the score aggregation model needs to infer the accurate and unbiased score for the submission from the scores recommended by peer graders.

At the beginning of the section, we describe the problem of assignment allocation in peer grading. And then we present the algorithm of LPT that is used in the field of task scheduling, and how we can extend the idea LPT for the purpose of peer grading and introduce the new algorithm MLPT. At the end of the section, we briefly discuss our research work on student performance evaluation model.

3.1 The Problem of Submission Allocation in Peer Grading

In the process of peer grading, each student plays two roles: a grader and a submitter. Suppose that there are $\mathbb{N}$ students participating peer grading, and there are also $\mathbb{N}$ submissions from these students. For convenience, we suppose that each student needs to grade $\mathcal{M}$ submissions and each submission is graded by $\mathcal{M}$ graders, typically the value of $\mathcal{M}$ is between 3 and 6. The description of assignments allocation in peer grading is shown in Fig. 2. Specifically, the students are divided into $\mathbb{N}$ groups: each group contains $\mathcal{M}$ students, and each student is included in one group, so each group corresponds to one submission [1, 8].

The process of improving the allocating submissions can be expressed as the following problem:

Define the student collection $\mathrm{V} = \{v_1, v_2, \ldots, v_n\}$ and the submission collection $\mathrm{U} = \{u_1, u_2, \ldots, u_n\}$.

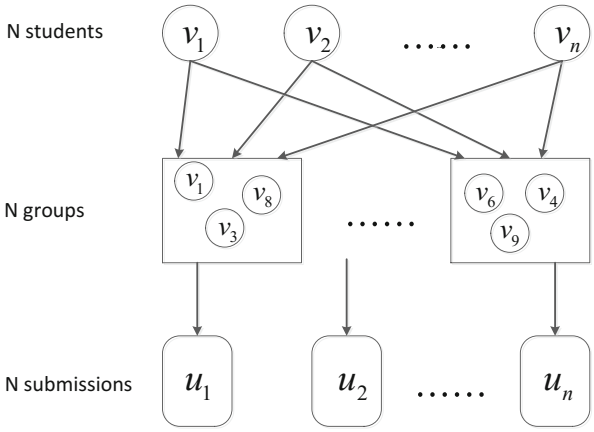

Fig. 2. The process of task assignment in peer grading.

The degree of a student v_i mastering the knowledge is denoted by $\mathrm{w} = \mathrm{w}(v_i)$ where $\mathrm{w}(v_i)$ lies in the interval (0, 1) and the parameter i falls in the range (0, n) if student v_i submits the submission u_i. $\mathrm{w}(v_i)$ that is generated by the student evaluation model, is regarded as the prior level of student knowledge mastery.

The problem is that given the parameter m how to divide the student collection $G_1, G_2, \ldots, G_n$, and make it satisfy the following conditions:

1. The grader collection $G_i = \{v_{i1}, v_{i2}, \ldots, v_{im}\}$ only contains m students.
2. For any student v_i, there are m groups including her.
3. $v_i \in G_i$, it means that all of the students grade the submission u_i, so the submitter v_i could be contained in the grading group G_i.

4. The variance of the students' comprehensive knowledge level must be the minimal, namely $\min\{var(w(G_1), w(G_2), \ldots, w(G_n))\}$, for the purpose that the grading level of each group is as close as possible in where $w(G_i) = \sum_{v_j \in G_i} w(v_j)$.

3.2 The Algorithm of LPT

The algorithm LPT is a basic method to solve the task scheduling problem in parallel computing system. The tasks scheduling problem in a parallel system can be described as the optimization process of minimizing the difference of processing duration among all the machines. Given the tasks $J = \{j_1, j_2, \ldots, j_n\}$, each of which has a processing time $t(j_i)$. Define $T = \{t(j_1), t(j_2), \ldots, t(j_n)\}$. Assume the tasks are allocated to the machines $M = \{m_1, m_2, \ldots, m_p\}$ with the same model. We assume that the tasks are independent and their arrivals occur at the same time point (Table 1).

Table 1. The process of the algorithm of LPT.

Algorithm: the LPT algorithm of task scheduling	
1.	Sort the tasks in the descend order by the costing time, make the result as $\mathbf{t}(J_1) \geq \mathbf{t}(J_2) \geq \cdots \geq \mathbf{t}(J_n)$.
2.	for $\mathbf{i} = 1, 2 \ldots, \mathbf{p}$: Assign the load L_i of machine M_i as 0, namely $L_i \leftarrow 0$. Assign the task collection $\mathbf{J(i)}$ of the machine M_i as empty, namely $\mathbf{J(i)} \leftarrow \boldsymbol{\phi}$.
3.	for $\mathbf{k} = 1, 2 \ldots, \mathbf{n}$: Select the current machine with the smallest load, recorded as M_i. Add the current task J_k to the task collection of the machine $\mathbf{J(i)}$, namely $\mathbf{J(i)} \leftarrow \mathbf{J(i)} \cup J_k$. Update the load of machine M_i, namely $L_i \leftarrow L_i + t(J_k)$.

The task scheduling has been proved to be a NP-complete problem [7, 14], namely the complexity of finding the optimal solution increases as the number of tasks and the number of machines in the exponential way. To find the acceptable solution in practice, it is necessary to employ the heuristic algorithm to compute an approximate solution.

The LPT algorithm adopts the greedy strategy for exploring approximate solutions according to the static task assignment policy. At the initial moment, all of the tasks are allocated to the machines, and during the executions of the tasks, the task schedule must remain until the end. The LPT algorithm arranges the pending tasks in a descending order and always assigns the tasks to the machines as soon as they become available. The detail process of the algorithm is described as follows.

The process of the algorithm LPT is simple and clear, and the time complexity can reach $O(n \log n)$ if the step of finding the earliest free machine adopts the minimum heap algorithm [10, 12]. Furthermore, the algorithm also has the optimal approximate result.

It can be proved that if $F^*(J)$ denotes the optimal costing time when assigning the task collection J to machine collection M, F(J) denotes the costing time adopting the algorithm LPT, so we can obtain:

$$\frac{F(J)}{F^*(J)} \leq \frac{4}{3} - \frac{1}{3m}$$

From the approximation relation we can draw that LPT is an efficient algorithm. The detailed process of proving the approximation relation can be found in references [9].

3.3 The Algorithm of MLPT

Similarly, there are many similarities between assigning submissions and task scheduling in parallel systems. We introduce a MLPT algorithm by redefining the process of LPT towards the needs of grading task assignment. Because for the problem of assigning submissions, assigning submissions to the graders can be viewed in another angle as assigning the graders to the submissions, so the student collection $V = \{v_1, v_2, \ldots, v_n\}$ is considered as the task collection $J = \{j_1, j_2, \ldots, j_n\}$ and the knowledge level $w(v_i)$ of each student is the processing time $t(j_i)$, then the submission collection $U = \{u_1, u_2, \ldots, u_n\}$ is considered as the machine collection $M = \{m_1, m_2, \ldots, m_p\}$. According to these above assumptions, assigning the graders to submissions in peer grading is equivalent to allocating the tasks to machines in the scheduling process, the MLPT algorithm can ensure the sum of the knowledge levels of the graders in each group is close to any of the others (Table 2).

Table 2. The algorithm of MLPT.

Algorithm: the algorithm of MLPT allocating submissions
1. Sort the graders by the level of knowledge in descending order, namely $w(v_1) \geq w(v_2) \geq \cdots \geq w(v_n)$
2. For $i = 1, 2, \ldots, n$:
The submission collection U(i) graded by grader v_i is assigned as empty, Namely, $U(i) \leftarrow \phi$.
3. For $j = 1, 2, \ldots, n$:
The comprehensive grading level L_j corresponding to the submission u_j is assigned to 0, namely $L_j \leftarrow 0$.
The student group V(j) corresponding to the submission u_j is assigned to 0, namely $V(j) \leftarrow 0$.
4. For $i = 1, 2, \ldots, n$:
For $k = 1, 2, \ldots, m$:
Select the submission with the smallest comprehensive grading level from submission collection $\{u_j \mid j \neq i\}$, recorded as u_j.
Insert the current grader v_i to the submissions collection u_j, namely $V(j) \leftarrow V(j) \cup v_i$.
Update the comprehensive grading level L_j of the submission u_j, namely $L_j \leftarrow L_j + w(v_i)$.
Insert the submission u_j to the collection U(i) of graded by grader v_i, namely $U(i) \leftarrow U(i) \cup u_j$.

Moreover, there are two conditions in assigning submissions compared to task scheduling. One condition is that the redundant grading parameter m guarantees that the number of different submissions for each grader is the same [13]. The other is that the grader could not evaluate his own submission.

3.4 Student Performance Evaluation Model

We establish a two-stage model to assess student mastery level of each knowledge skill, in other words, to estimate the probability that a student has learned the specific knowledge of a chapter. The first stage is the behavior-based student engagement model, which first analyze student video-watching activities within a chapter and extract interpretive quantities to predict the probability that a student has mastered the knowledge of that certain chapter. This model extract eight features including the total video playing time, the number of video played, the number of pauses, the number of sessions, the number of requests, the number of rewinds, the number of slow play rate setting, the number of fast forwards. The logistic regression method is used to fit these features and predict the engagement level of every student.

The second stage is the knowledge tracing model to infer the level of student knowledge mastery. The knowledge tracing model, popularized in Intelligent Tutoring System (ITS), leverages the quiz response sequence of every student to assess their learning outcomes. Our work adopts the knowledge tracing model and ameliorates it by combining the prediction results obtained in the behavior-based student engagement model. Essentially, it is based on a 2-state dynamic Bayesian network where student responses are the observed variable and student knowledge is the latent one. The model takes student response sequences and uses them to estimate the student's level of knowledge. The initial state of the model can be trained by the results obtained in the behavior-based student engagement model.

4 Datasets

The dataset using in our paper is collective from the course of computer network experiment abbreviated as CNE) of our MOOCs platform of Beihang Xuetang (http://mooc.buaa.edu.cn). The MOOC course CNE includes three assignments for the Sophomore undergraduate who had not studied this course before and we only use assignment 1 and assignments 2. The course required that the students had to upload the submissions before the end of each week.

Specially, the session course was open in the year of 2015 and active for a 11-week semester. The course contains 10 chapters where there are several video units per chapter and a final examination. After each video unit, there are multiple choice problems as quiz for students. The number of the problems ranges from 5 to 13 in each unit. Also, the course contains 4 open-ended design problems that demand the students to design and implement the experimental programs independently. The score of these open-ended problems are generated through the peer grading mechanism in our MOOC platform. There are approximately 600 active students each semester since the course CNE went online and roughly 500

students to take part in the final exams. The number of the records collected by the system is 2 million, which logs every user interaction with the courseware including scanning and clicking, video interaction, posting questions and comments in the course forum.

After removing the incomplete items in the original dataset, we choose 210 students including 210 items, and in our setting each student grade 4 submissions and each submission must be graded by 4 graders. In order to prove the advantage of the MLPT algorithm, we also produced two simulation datasets imitating the real data. In the simulation datasets, the number of items is 200 and we valued the redundancy peer grading number as 3, 4 and 5 in our experiments. The major difference between the real data and the simulation data is that the prior is generated from the normal distribution and evenly distribution, respectively.

5 Experimental Results

This section mainly verifies the algorithm of MLPT from both the real dataset and simulation dataset. For each dataset, we compute the variance of all the groups using the sum of each group as one item and we have run 10 times for the computing an average value. In both Tables 3 and 4, the simulation dataset 1 is generated from an evenly distribution and the simulation dataset 2 is generated from a random normal assignment distribution.

Table 3. The comparison between random assignments and MLPT based on the simulation datasets.

	Random assignment	MLPT
Simulation dataset 1	0.381	0.009
Simulation dataset 2	0.266	0.017

Table 4. The comparison between random assignments and MLPT based on the real datasets.

	Random assignment	MLPT
Assignment 1	0.355	0.053
Assignment 2	0.376	0.046

Table 3 shows the results from the simulation datasets, the dataset 1 and dataset 2, which are generated from the evenly distribution and normal distribution, respectively. The MLPT algorithm reduces the variances of the knowledge master level among the groups from 0.381 to the 0.009 on the simulation dataset 1 and from 0.266 to 0.017 on the simulation dataset 2.

Similarly, Table 4 is the results generated from the real datasets. The value of the variance is 0.053 after using MLPT in Assignment 1, and the optimization percentage is 85.0%. The optimization percentage is 87.7% and the MLPT value is decreased from 0.376 down to 0.046 in Assignment 2. Note that the MLPT optimization results on the simulation datasets seem better than the results on the real dataset. The reason may be that the prior inferred from the performance of quiz doesn't completely associated with the actual knowledge level of students in the open-ended problems.

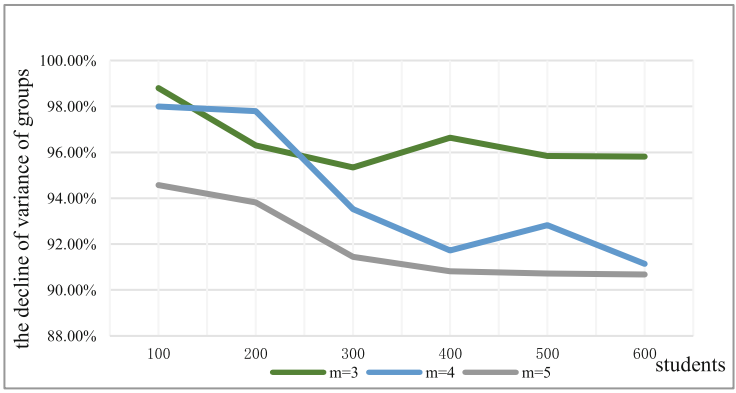

Fig. 3. The degree of declining of variance from MLPT algorithm to the random algorithm.

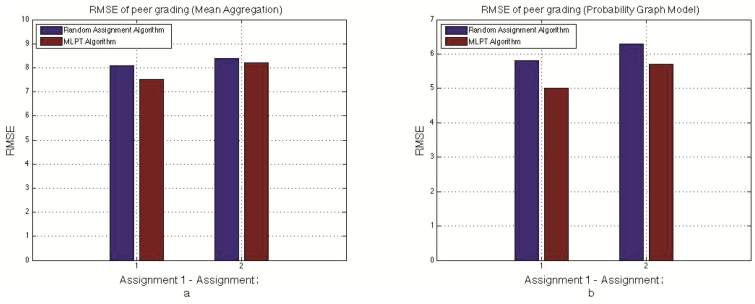

Fig. 4. The effectiveness of task assignment on RMSE of peer grading.

Through simulation, we can explore different settings such as the various numbers of students and the different values of the parameter m to compare the performance between the MLPT algorithm and the random algorithm. In the simulation dataset, we have verified a series of datasets with the number of students varying from 100 to 600 and the value of parameter m ranging from 3 from 5. And in each simulation, we compute the decline in the group variance caused by the MLPT algorithm over the Random algorithm. Figure 3 plots the results generated from the Random and MLPT algorithms. It demonstrates that the MLPT algorithm can reduce the variance of all the groups significantly in comparison with the Random algorithm, especially when there are not many students and the parameter m is relatively small. Figure 4 shows the comparison results generated from the MLPT and Random algorithm based on the real datasets in the process of peer grading. The vertical axis indicates the Root Mean Square Error (RMSE), which is used to compute the accurate of grading scores. In the research of Mi and Yeung [11], researchers proposed new score aggregation methods based on probabilistic graph models to infer accurate scores of student submissions. These methods outperform the method using the mean aggregation strategy. It is necessary to investigate whether the MLPT performance has any impact on different aggregation methods. The result in Fig. 4(a) shows the difference of RMSE between the MLPT algorithm and the RMSE algorithm based on the mean aggregation strategy and

Fig. 4(b) shows the results based on the probabilistic graph model. No matter which aggregation strategy is adopted, the MLPT algorithm can effectively improve the overall accuracy of peer grading scores.

6 Discussion and Future Work

Peer grading in MOOC platforms need an effective framework for grading tasks allocation and score aggregation in order to deliver accurate feedbacks of homework evaluation to online students. Our paper presents a complete peer grading framework and introduces a new task scheduling algorithm MLPT for the task assignment problem of peer grading. This framework can infer the knowledge mastery level of every student using a two-stage data analysis pipeline with the behavior-based engagement model and knowledge tracing-based student performance evaluation model Our task allocation algorithm regards the assessment outcomes of each student in the quizzes of the MOOC courses as the priors and utilizes the priors to build peer grading groups. It aims at minimizing the capability difference among grading groups and generating a fair and accurate evaluation for each student submission. The experimental results from both simulation datasets and real course datasets demonstrates that the MLPT algorithm can outperform the conventional random strategy.

There are a number of issues to be addressed in future work. In this paper, we assume that the redundancy number of peer grading tasks for every submission remains constant. However, in practice, students may fail to complete their grading tasks in time, thus resulting in less redundant grading results for some submissions. It would be interesting to investigate whether this problem affects the performance of the schedule algorithm. Another issue is whether the errors in the student evaluation model has any impact on the MLPT algorithm. Given the probability nature of the knowledge tracing model, it is unavoidable that it may give a biased estimation of knowledge levels of some students when there are noises in the data of user interactions and question answering. One of the feasible solutions is to have staffs to cross-examine some peer grading results and correct the potential errors.

Acknowledgments. This work was supported in part by grant from State Key Laboratory of Software Development Environment (Funding No. SKLSDE-2015ZX-03) and NSFC (Grant No. 61532004).

References

1. Fonteles, A.S., Bouveret, S., Gensel, J.: Heuristics for task recommendation in spatiotemporal crowdsourcing systems. In: Proceedings of the 13th International Conference on Advances in Mobile Computing and Multimedia, pp. 1–5. ACM (2015)
2. Cheng, J., Teevan, J., Bernstein, M.S.: Measuring crowdsourcing effort with error-time curves. In: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, pp. 1365–1374. ACM (2015)
3. Qiu, C., Squicciarini, A.C., Carminati, B., et al.: CrowdSelect: increasing accuracy of crowdsourcing tasks through behavior prediction and user selection. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, pp. 539–548. ACM (2016)

4. Howe, J.: The rise of crowdsourcing. Wired Mag. **14**(6), 1–4 (2006)
5. Wiki for Multiprocessor Scheduling Information. https://en.wikipedia.org/wiki/Multiprocessor_scheduling
6. Piech, C., Huang, J., Chen, Z., et al.: Tuned models of peer assessment in MOOCs. arXiv preprint arXiv:1307.2579 (2013)
7. Coffman, Jr. E.G., Sethi, R.: A generalized bound on LPT sequencing. In: Proceedings of the 1976 ACM SIGMETRICS Conference on Computer Performance Modeling Measurement and Evaluation, pp. 306–310. ACM (1976)
8. Alfarrarjeh, A., Emrich, T., Shahabi, C.: Scalable spatial crowdsourcing: a study of distributed algorithms. In: 2015 16th IEEE International Conference on Mobile Data Management, vol. 1, pp. 134–144. IEEE (2015)
9. Baneres, D., Caballé, S., Clarisó, R.: Towards a learning analytics support for intelligent tutoring systems on MOOC platforms. In: 2016 10th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS), pp. 103–110. IEEE (2016)
10. Gonzalez, T., Ibarra, O.H., Sahni, S.: Bounds for LPT schedules on uniform processors. SIAM J. Comput. **6**(1), 155–166 (1977)
11. Mi, F., Yeung, D.Y.: Probabilistic graphical models for boosting cardinal and ordinal peer grading in MOOCs. In: AAAI, pp. 454–460 (2015)
12. Massabò, I., Paletta, G., Ruiz-Torres, A.J.: A note on longest processing time algorithms for the two uniform parallel machine makespan minimization problem. J. Sched. **19**(2), 207–211 (2016)
13. Gardner, K., Zbarsky, S., Harchol-Balter, M., et al.: The power of d choices for redundancy. ACM SIGMETRICS Perform. Eval. Rev. **44**(1), 409–410 (2016)
14. Feier, M.C., Lemnaru, C., Potolea, R.: Solving NP-complete problems on the CUDA architecture using genetic algorithms. In: International Symposium on Parallel and Distributed Computing, ISPDC 2011, Cluj-Napoca, Romania, pp. 278–281. DBLP, July 2011
15. Ul, Hassan U., Curry, E.: Efficient task assignment for spatial crowdsourcing. Expert Syst. Appl: Int. J. **58**(C), 36–56 (2016)
16. Jung, H.J., Lease, M.: Crowdsourced task routing via matrix factorization. Eprint Arxiv (2013)
17. Karger, D.R., Oh, S., Shah, D.: Budget-optimal crowdsourcing using low-rank matrix approximations. In: 2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton), pp. 284–291. IEEE (2011)
18. Yan, Y., Fung, G.M., Rosales, R., et al.: Active learning from crowds. In: Proceedings of the 28th International Conference on Machine Learning (ICML 2011), pp. 1161–1168 (2011)
19. Tong, Y., She, J., Ding, B., et al.: Online minimum matching in real-time spatial data: experiments and analysis. Proc. VLDB Endow. (PVLDB) **9**(12), 1053–1064 (2016)
20. Tong, Y., She, J., Ding, B., et al.: Online mobile micro-task allocation in spatial crowdsourcing. In: Proceedings of the 32nd International Conference on Data Engineering (ICDE 2016), pp. 49–60 (2016)
21. Tong, Y., She, J., Meng, R.: Bottleneck-aware arrangement over event-based social networks: the max-min approach. World Wide Web J. **19**(6), 1151–1177 (2016)
22. She, J., Tong, Y., Chen, L., et al.: Conflict-aware event-participant arrangement and its variant for online setting. IEEE Trans. Knowl. Data Eng. (TKDE) **28**(9), 2281–2295 (2016)
23. She, J., Tong, Y., Chen, L., et al.: Conflict-aware event-participant arrangement. In: Proceedings of the 31st International Conference on Data Engineering (ICDE 2015), pp. 735–746 (2015)