A COMPETITION-BASED STUDENT ASSIGNMENT MANAGEMENT SYSTEM

Xuliang Duan, Chao Wang
Department of Computer Science
Sichuan Agricultural University,
Ya'an, China

Abstract

This paper introduces the theories and methods of a competition-based student assignment management system (CSAMS) which is used to motivate students in their completion of assignments. The approach uses a time-competition model, presents a peer-evaluation method using a chess rating system, and also includes winnowing-based plagiarism (similarity) detection to build a competitive working environment to increase students' motivation in their coursework. Comparison analysis shows that the application of the time-competition model effectively eliminates procrastination during students' completing their assignments, and the plagiarism detection in CSAMS makes the similarities of course work controllable over an acceptable range. In short, practice feedback in several courses suggest that the CSAMS exerts a significant positive influence on students' completion of assignments and yields favorable results.

Key Words

Competition; Teaching; Peer-evaluation; Rating system; Winnowing

Introduction

Exercises and course assignments are very important components of a student's course of study. However, students' attitude towards these tasks is not always as serious as expected, in that a considerable portion of students are dilatory and indolent [1] and some even plagiarize [2-4]. The Competition-based Students Assignments Management System (CSAMS) has been developed for such cases to motivate students and to help them perform more effectively in their studies.

In modern e-learning, engaging students and improving their motivation with social learning technologies have become an important trend.[5]. Social learning is becoming an increasingly important role in providing motivation or incentives for learning in universities. Vassileva summarized some different ways to make learning or work more gratifying, such as: 1) make it game-like, a combination of challenge and fun, 2) boost the feeling of achievement by providing constant feedback on performance, 3) relate performance to status in peer group (social reward), and 4) relate performance to marks or credentials[6]. These methods are very instructive for the designing of incentive mechanism in CSAMS.

In the pursuit of motivating students in learning, new areas of science such as social psychology, economic/game theory, and peer learning become relevant as a source of methods and techniques. Peer to peer learning such as peer reviews are very common in college classrooms. Assessment can foster peer learning, Wolfe used the peer review method in student assignment scoring and described a system where each week students accessed and scored each other's assignments on a course Web site. The peer review process supported the teacher in the roles of "coach" and "resource" as opposed to "lecturer" and "enforcer" and worked exceedingly well. In that process, as Wolfe concludes, students learned from their peers, received quick and plentiful feedback and felt comfortable in the role of critical reviewer. While, there were a few problems such as in most classes it was too much work for the students to do a meaningful review of every other student each week[7].

Plagiarism is an unavoidable topic in the digital age. "In surveys, nearly 70% of college students admit to having taken material from the Internet without properly crediting its source"[8]. Not only from the Internet, but also the works of others are involved, and it is common practice that a considerable portion of students at some time directly copy all or part of another's work as their own. To deal with such cases, local or Web-based plagiarism detection has been introduced to review programming assignments and to stimulate students' performance [4,9,10]. Plagiarism detection is included as an important function of most of the online judging systems, and similarity is an important factor in automatic grading[11,12].

Competition-based learning has been shown to be an effective methodology for stimulating motivation[13,14]. Students have a more positive and more diligent attitude towards their assignments in competitive situations[15]. The study presented in this paper attempts to combine time competition, chess-like peer-evaluation, and plagiarism detection into the CSAMS to provide a competition-based assignment work environment. In the CSAMS, students compete to do assignments in less time and compete with others in peer-evaluation under the common stress of plagiarism detection. In particular, similarity detection works not only for programming codes and other plain text, but also for multimedia documents with images, tables, graphics, and objects that most types of digital assignment involve.

Competitions in Assignment Management

Time Competition

In general, most students wait until the deadline to submit their reports because of the absence of time competition. As a result, finally, many of them have no time to process the mountain of assignments and find that plagiarizing is the only option. In 2005-2006, Vassileva et al. proposed an adaptive reward mechanism to reward newly contributed resources in social learning technologies designing. Therefore, time was introduced as one of the reward factors that FC (community reward factor) has its maximum value when a new topic is introduced, and then more and more faster decreases with the time. Their results proved that this adaptive reward mechanism was effective in motivating users to share resources early[6,16]. In CSAMS, there is no other social learning factor such as resource quality involved; for simplicity, such cases are handled by an automatic grading method that gives a higher score for assignments submitted earlier within the specified period. Therefore, TopCoder's online computer-programming time-competition model [17] was used in this work. For each course assignment, a separate task is assigned to everyone in the class, with each task having its own start and end time in the system. The start time is just the starting point of automatic scoring; early submissions gain high scores, while those who submit their jobs near the end time will obtain a much lower score.

The automatic time score can be calculated as follows[17]:

$$AutoScore = 100 * \left(Floor + \frac{(1 - Floor) * TT^2}{10 * PT^2 + TT^2} \right), \quad (1)$$

where *Floor* represents the lower bound of a "very low score", TT is the maximum allowed time, and *PT= submittime - starttime* is the actual time consumed in doing a task. Instead of a linear function, a power function is used to create a greater grade gap during the period when most of the students have probably just finished the job. For example, assuming that the total time is 300 minutes and *Floor* is 0.3, the curve of score versus time is as shown in Figure 1.

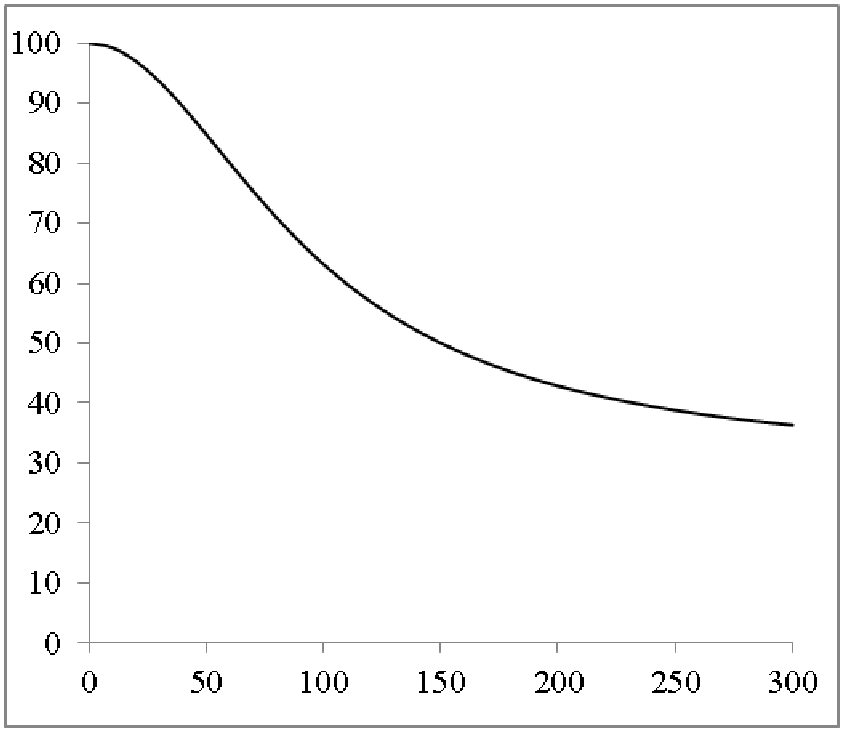

Figure 1. Automatic score as a function of time.

In CSAMS, it is recommended to set the start time a little later than the actual time that the task is assigned (e.g., the estimated shortest time to complete) to avoid high-scoring premature job submissions. However, the autoscore is only a part of the final score; the other part is set by the professor's judgment. These two score components are prorated as follows:

$$Score = Weight * AutoScore + (1 - Weight) * Grade, \quad (2)$$

where *Weight* is the weight of the autoscore, which should be adjusted according to the task type: if the task is relatively simple, the autoscore can account for a large proportion, while if quality and accuracy are more valued, *Weight* should be a small value, even zero. A zero weight value means that the submission time is no longer considered because the final score relies only on the professor's judgment.

Competition in Rating System-Based Peer-Evaluation

Peer assessment is a special form of collaborative learning, in which peer students learn through assessing others' work; different types of tasks can be performed in different peer assessment methods [18]. The peer-evaluation approach described in this paper reduces the difficulty and increases the interest of judging each other's work based on a paired-comparison approach as used in chess and other games. It borrows a chess rating algorithm to evaluate student work, with better performance leading to higher scores after repeated paired comparisons.

Disinterested evaluation of students' assignments is an exhausting job. Efforts have been made to use Web-based voting for assessing assignments, but the results are not very satisfactory because there is too much work as Wolf points out [7] and too much cheating despite the use of multiple technical methods to prevent it. Inspired by chess competitions, a new peer-evaluation method for students' assignments is presented here based on a chess rating system. In this system, the score is not an absolute measurement, but can only be inferred from wins and losses against other players. A player's rating depends on the ratings of his or her opponents and the results obtained against them. Students achieve their final rating levels through competition with each other.

Initially, every student has the same rating; in competition, two students' submissions are shown together on a screen, and the evaluator (also a student) is called upon to choose the better one as the winner (anyhow he or she is required to choose a winner with a draw not considered; or refresh the page to start a new comparison), thus increasing the winner's score while decreasing the loser's. The list of ratings is updated over time so that every student can find his or her position in the list. As a part of the teaching process, everyone is required to choose at least a given number of times to guarantee that every player has a roughly equal number of rating opportunities. At the end, the final grade is calculated according to the rating results, and the higher the rating, the higher is the final score.

The chess rating based peer-evaluation is game-like, provides immediate feedback by ranking list, and the rating result (students' performance) is related to final marks. Most of these features fit in well with the ways of making learning more gratifying that summarized by Vassileva in the “Introduction” section.

Among all rating-system algorithms, Elo forms the basis of many other subsequent rating algorithms. The Elo rating system is a statistics-based method for calculating the relative skill levels of players in two-player games such as chess and in other multiplayer competitive games. It is named after its creator, Arpad Elo, a Hungarian-born American physics professor and also a master-level chess player in the United States Chess Federation (USCF)[19,20].

The Glicko and Glicko-2 rating systems were invented by Mark Glickman as an improvement of the Elo system. Glickman's principal contribution to measurement is the concept of ratings reliability, called RD for "ratings deviation", which is a measurement of the accuracy of a player's rating. For example, a rating=1500 and an RD=100 mean that the player's real strength is in [1500-2RD, 1500+2RD], or in other words, that the strength is between 1300 and 1700 with 95% confidence. The algorithm considers the fact that an accurate player (with low RD) would not change his rating very much, and as a result, the rating change is smaller after a game (or a game period) when the player's RD is low and also the opponent's is high [21-23].

The Glicko-2 rating system improves upon the Glicko rating system by further introducing the rating volatility σ, which indicates the degree of expected fluctuation in a player's rating. The volatility measure is high when a player has erratic performance, and the volatility measure is low when the player performs at a consistent level[23].

CSAMS uses Glicko-2 as its rating algorithm. The chief functions of CSAMS peer-evaluation are as follows:

(1) Authentication. In the study of impact of peer evaluation confidentiality on student marks, Peterson found that confidential evaluations significantly dropped students' marks while non-confidential evaluations raised them.[24] So in this system, everyone who wants to take the peer-evaluation must first log on with a student ID, both to record everyone's every choice to avoid irresponsible voting, and also because the peer-evaluation is an important part of the teaching process and every student must take part in a certain number of voting rounds.

(2) Paired competition. For example, to evaluate Web page design assignments, the two designs are embedded in a Web page, and students are asked to express their preferences for the winner in their own terms. In this case, it is simple and easy for each student to make his or her alternative choice. The two opponents in each round are randomly selected from students who have an approximately equal rating score.

(3) Ranking list. Pope considered the effect of stress on the students in their self- and peer assessment and found that this assessment stress leads to improved student performance in summative tasks [25]. In CSAMS, the evaluation results are published after a while as a rankings list; everyone can view his or her own rank and also appreciate the degree of others' satisfaction with their own work.

To make the peer-evaluation results more objective and comprehensive, the following technical and control measures are used in the CSAMS assessment process:

(1) Clear criteria. A list of explicit evaluation standards such as originality, practicability and esthetic appearance are provided, and everyone is asked to follow the guidelines in making their own choice.

(2) Avoidance of canvassing. In CSAMS, the two opponents in each round are randomly selected from students who have an approximately equal rating score. In addition, some technical measures are adopted to avoid encountering a voter's own work. In any case, it is practically impossible to canvass for anybody in the whole peer-evaluation process.

(3) Set the ceiling amount of votes allowed. To avoid the rating be swayed by a small group of highly active users and reduce the influence of individual irresponsible voting, CSAMS restricts students to making their choice no more than a limited amount.

(4) Teacher participation. Teachers are also involved in the evaluation process. At the beginning of the evaluation, related teachers are invited to participate in this teaching game. There is no restriction on the number of times a teacher can vote, and they have extra days after the student evaluation period. Above all, the CSAMS aims to make the results more objective and interesting but not to lighten teachers' workload.

(5) Clean up malicious data manually. It is easy to identify malicious votes according to some simple rules. For example, if the time interval between a voter's choices is too short, or his or her choices are regular "left win" or "right win" (the system records every choice made by each voter), these votes can be ruled out and the rating result is rebuilt according to the rating logs.

However, it is inevitable that an extreme individual is probably gaming the system by picking the "winner" randomly, and these malicious data are hard to identify and clean up. As the amount of allowed votes for each is limited, and the choices will never be focused on one person, there wouldn't be much random choices of one's assignment. In reality, a chess player may have erratic performances within some certain time periods, but in the long run, these limited deviations would have very slightly effects on his/her overall ranking. So we can view the few random choices as the player's erratic performances.

Winnowing-Based Plagiarism Detection

Competition in Diligence of Working

In a nontransparent study environment, a large number of students' assignments contain some (or much) plagiarism because students believe that it is hard to detect. Martin's study confirms that students who have a stronger belief that plagiarism will be detected will be less likely to plagiarize, and those who plagiarize less will be more likely to develop better skills, more creativity, and greater self-confidence[26]. CSAMS publishes suspicious documents and their corresponding similarity values online in a timely fashion to make plagiarism public. In addition, the final score is affected by the severity of plagiarism; a very high similarity value will lead to a very low grade or even to disqualification. The system makes students working in an open competitive environment aware that plagiarists will be publicized in full view of everyone. These huge costs of plagiarism will encourage everyone to think seriously about his or her own attitude towards completing assignments.

Although it is mature and prevalent that various plagiarism techniques are applied in the detection of textual-based documents such as written text or program code, however, how to cope with multimedia documents with images, tables, and other objects is seldom described in the literature. CSAMS uses the winnowing document similarity-based detection algorithm [39], which is a well-known fingerprint algorithm. Plain text, program codes, or multimedia documents with images, tables, and other objects, will all be detected as plagiarized or not as they are submitted to prevent plagiarism and thus to increase students' performance.

The Winnowing Algorithm

Winnowing is an efficient k-gram document fingerprint calculation method which was proposed by Saul Schleimer in 2003 [27] . The construction of the fingerprints guarantees a set of theoretical properties in terms of fingerprint density and substring matching detection capability[28]. Winnowing is used by MOSS, a widely used service for detecting plagiarism primarily in programming assignments, and works extremely well in that system[27].

There are three correlative parameters, t, k and w, in the winnowing algorithm. The threshold t guarantees that only matching substrings longer than or equal to t can be detected, and k as in k-grams is the noise threshold: any matches shorter than this noise threshold will not be detected. The values of t and $k \leq t$ are chosen by the user. Given a hash list $h_1, h_2, \ldots, h_n$ (e.g., hashed from the k-grams of a document), if $n > t - k$, then at least one of the h_i must be selected to guarantee detection of all matches of length at least t. Therefore, let the third parameter $w = t - k + 1$ be the window size and $h_1, h_2, \ldots, h_k$ be the entire hash sequence that represents a document. Each position $1 \leq i \leq n - w + 1$ in the sequence defines a window of hash values $h_i \ldots h_{i+w-1}$, which guarantees that at least one hash value must be selected in each window to compose a sufficient fingerprint of the document.

The core of the winnowing algorithm is the hash selection method (winnowing definition 1): *In each window, select the minimum hash value. If there is more than one hash list element with the minimum value, select the rightmost occurrence. Now save all selected hash values as the fingerprints of the document* [27].

Building Fingerprints for Documents

Course assignments and laboratory reports are often of two main types: plain text such as program code, or MS Word documents which contain not only rich text, but also tables, graphs, and images. As long as it is possible to obtain any text or objects from a file, it is easy to build a fingerprint for the document using the winnowing algorithm and to perform detection. Depending on the features of the two document types, CSAMS uses different methods to generate the document fingerprints.

Obtaining Fingerprints from Text

In the application proposed here, the winnowing algorithm is used to extract the fingerprints from document text. The first step is to read and then to clean up the text, for example, using regular expressions to replace irrelevant features such as spaces, control characters (e.g., tab, wrap, return), punctuation, and stop words.

The winnowing parameters must be chosen appropriately for different types of jobs. For character-based text such as Chinese documents, the most common words are often two to four characters, and therefore a k-value should be chosen that is larger than the normal length of a word and smaller than the length of a sentence. A value of k=8 was chosen as the optimum.

For program code and HTML documents, there is more complex work to do. In addition to cleaning up common space and control characters, some scholars suggest that comments and even variables should also be removed. In addition, certain more complex plagiarism detection techniques based on lexical analysis convert code structures regardless of programming language or variables to a special form of coding [29]. However, the authors did not believe it necessary to process the program text excessively in the proposed application because the presence of the same comments provides evidence of plagiarism. In any case, regardless of the form of the processed text, it is possible to winnow it and obtain the fingerprints. As for the value of *k*, the authors suggest assigning a larger value because the length of the language declaration "public" or the basic html tag "<body>" is already 6. A value of *k*=12 was chosen for the proposed similarly detection approach.

Objects in Documents

With document types other than plain text, such Microsoft Word documents which include shapes, images, and tables, it is not easy to run winnowing in the same way as for text. For these objects, a more simple method is used that directly computes the hash value of each object as the fingerprint. Of course, if there are too many hash values in a document, it is still possible to perform winnowing as for *k*-grams on text.

In the hash calculation for objects, the following approach is used:

a) Images: directly compute the hash value for an image; in MS Word documents, the same image, even with a different size in different documents, will retain the same message digest.

b) Shapes: a shape may also contain some text, such as the text box in MS Word, so such shapes are combined, and the hash value=H (width + text + height), where H is the selected hash algorithm.

c) Tables: a table is similar to the shape described above to some degree, but its size is easy to change so that it cannot be considered in the same way. Therefore, the hash value is computed for the text in each table row to obtain a sequence of hash values for the whole table.

Checking Plagiarism at Submission

As a piece of work is submitted, CSAMS performs so-called "submission detection" by comparing this document to all previously submitted ones immediately and returns the maximum similarity value. If the similarity is greater than the preset threshold, a warning will be displayed and a confirmation required for final submission, or else the system may accept the submitted assignment and save its fingerprints into the database.

For purposes of real-time detection, the fingerprints of the document being detected are first computed, then they are compared to each of the stored fingerprints in the database, and finally the maximum similarity value is returned.

There are several ways of comparing similarity in a database. The first is data-based retrieval. For example, the hash values of a document fingerprint can be separated by spaces and stored as a string in a full-text index field. When querying, for example in MySQL, the SQL statement "MATCH AGAINST" can be used to query the detected fingerprint (also a string of separated hash values) from the database and to select the highest similarity value as the result. This is an efficient solution because full-text indexing is a mature technology.

The second solution used by CSAMS is to use a vector space model (VSM) to compute the distance between two vectors [30]. This approach is sufficient, flexible, and fast enough to compare one vector to all the vectors in the database. For example, suppose that the winnowing fingerprint of the document currently being detected is *A*= (003,342,063,219,524,342,031,342) and that there are already three fingerprint strings in the database. Selecting and splitting the three strings into sequences yields: *B*= (782,063,342

,219,003,342), *C*= (229,524,457, 342,452,242, 081), *D*= (123,756,542,031,342, 156,219,031). In a VSM, documents are represented as vectors. When performing a comparison, the CSAMS first generates the VSM from the two sequences being compared and then calculates the cosine of the two vectors as the similarity. For example, Table 1 shows the VSM sample of A and B, where the frequency of items in a document is the value in the corresponding vector.

Table 1. VSM Example: VSM from fingerprints of A, B.

$A \cup B$	003	342	063	219	524	031	782
VectorA	1	3	1	1	1	1	0
VectorB	1	2	1	1	0	0	1

The similarity of A and B is the cosine of the two vectors[31]. For this case, sim(*A*,*B*)= cos(vectorA, vectorB)= 0.850, sim(*A*,*C*)= 0.404, sim(*A*,*D*)= 0.507, so the query result will be the maximum similarity value of 0.850.

All-Pairs Plagiarism Detection

Generally, the requirements of an assignment given to all students are always the same, and therefore it is inevitable that there will be varying degrees of content similarity in most of the documents. In this case, the query method mentioned in last section is not very precise because this common content should not be used to calculate the similarity.

In addition, CSAMS also provides all-pairs detection for all submissions to find the highest-similarity pairs or plagiarism clusters. In this situation, the VSM was generated, not only from the two compared fingerprints, but also from all documents, and the value in the VSM vectors was not each term's frequency, but the *tf-idf* (frequency-inverse document frequency) weight, which is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus[32].

However, this is not a perfect approach because it has two bottlenecks: the first is the number of documents, because the complexity of the all-pairs comparison algorithm is $O(n^2)$, and large numbers of documents would lead to considerable inefficiencies; the second constraint is the very large memory requirement for the enormous VSM arising from hundreds of documents, especially in the case of large documents and high density of winnowing. Moreover, computing *tf-idf* requires large computing resources, both time and space. Fortunately, the CSAMS is not used for mass data retrieval, but for management and evaluation of a limited number of assignments. Generally, the size of a class is between 30 and 300; this means that the number of documents associated with an assignment is also in this range, which is within the capacity of the system. Moreover, a professor could also choose the method described in Section "Checking Plagiarism at Submission", which is very efficient and reliable, for all-pairs comparison if necessary.

Results and Discussion

CSAMS is a Web-based system running under Microsoft Windows Server and was developed in ASP.NET 4.0. Two courses of two semesters were involved in the analysis. One course in different semesters has the same contents and requirements. The courses in the first semester were without the support of CSAMS, while the assignments of the two courses in second semester were all under the control of CSAMS. There were 6 assignments and 1 final practice work for the course "Web Design", and a total of 5 assignments for "Algorithm Design and Analysis".

The rating-based peer-evaluation provides a game-like platform for students to learn more and promote exchanges from each other, so as to achieve better learning. Comparison analysis of two courses within and without the CSAMS shows that the time-competition method effectively eliminates procrastination during students' completing their assignments, and the plagiarism detection effectively controls the coursework similarities in an acceptable range.

Rating System-Based Peer-Evaluation

The final assignment for the "Web Design" course was peer-evaluated using the Glicko rating method which is supported by the system. Other types of assignments such as images or programming codes or Microsoft Word documents could also be involved. However, according to our practice, students do not tend to spend lots of time reading others' essays carefully, so the assignments that are visible in judgment are the most appropriate forms. In this task, everyone was required to complete the designing and implementing working of a web site within the stipulated time as requested. As there is no test paper, generally, the grading of this type of assignment is largely dependent on teachers' subjective judgments. Rating-based peer-evaluation is a very open way of doing assessment. The Glicko-2 rating algorithm as implemented required each student to choose his or her preference in every pair comparison (see Section “Competition in Rating System-Based Peer-Evaluation”). The rating period was 10, which means that a player's rating was updated only every 10 comparisons. The other parameters were set as recommended by Glickman in his instructions [23]: the system constant τ=0.6, and initially r=1500, RD=350, σ=0.06 for each new player. Figure 2 shows the schematic of main rating page which contains two randomly selected webpages for rating.

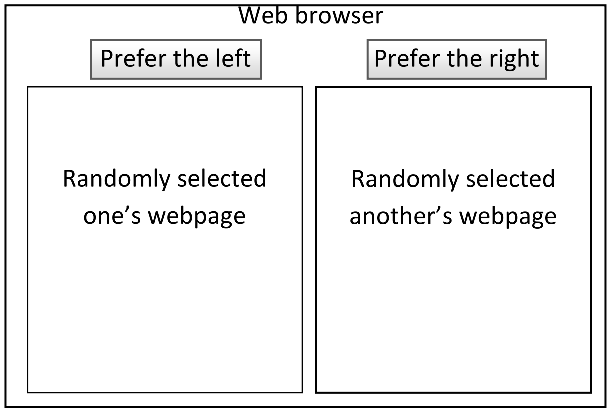
Web browser
Prefer the left
Prefer the right
Randomly selected one’s webpage
Randomly selected another’s webpage

Figure 2. Web-based peer-evaluation interface schematic (the two pages are embedded in the two sides of the main page, students pick the winner by clicking the "Prefer Left" or "Prefer Right" button or pressing the left/right arrow key on keyboard).

There were in total 158 qualified submissions; students were asked to complete their evaluations within 10 days, and as a result of 13309 comparison records (without 2765 of teachers'). Each student is required to take at least 30 times comparing while an average of 78 were performed per student, so it is obvious that most students make their participation in this process not only for accomplishing their obligations but also for interest and enthusiasm; the chess-like method motivate students in their peer-evaluation process. For all the submissions (students' final assignments), the maximum number of comparisons was 173 and the minimum 163, a range of no more than a single Glicko-2 round (10 comparisons). As RD is the measurement of the accuracy of a player's rating, the deviation of RDs reflects that the evaluation is relatively objective because a different person has his/her own inclination. The assignments with high rating score and low deviation value are those well-accepted by majority of all evaluators. The Glicko-2 statistical parameters are given in Table 2, and Figure 3 shows the histogram of all students' ratings.

Table 2. Statistics of r, RD, σ in Glicko-2.

	r	RD	σ
Max	2068.575	183.632	0.06192
Min	763.061	96.918	0.05966
Average	1483.167	140.334	0.06000
SD	252.236	11.536	0.00023

The rating score could be used for computing the final grade of student. The final grade could be gained by linear mapping from the ratings to the score range which we need. For example, after deliberate grading, we regard the highest rating assignment should take 100 points, while the last one only 60, one's final score (total 100) could be calculated as follows:

$$FinalScore = \frac{Rating - Lowest\ Rating}{highstRating - LowestRating} * (100 - 60) + 60 \tag{3}$$

where *Rating* is the student's rating score, 100 and 60 represent the range boundary of the students' final scores.

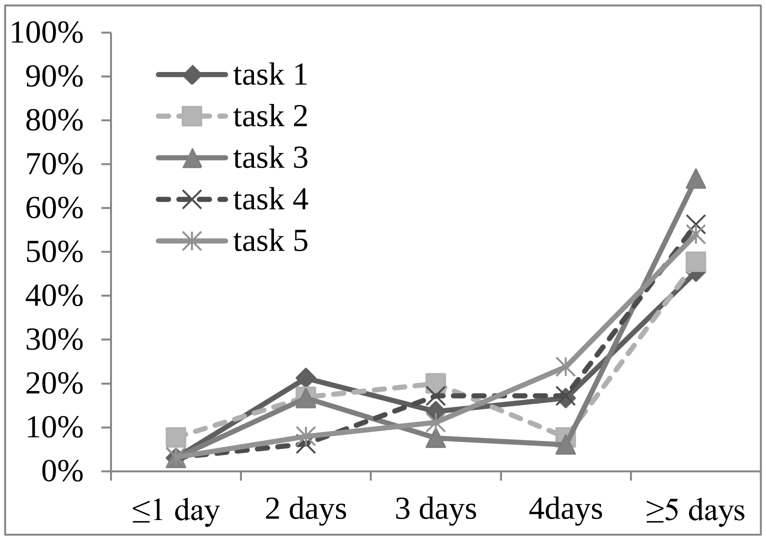

Fig. 3. (a) Completion percentage by number of days, course A without CSAMS.

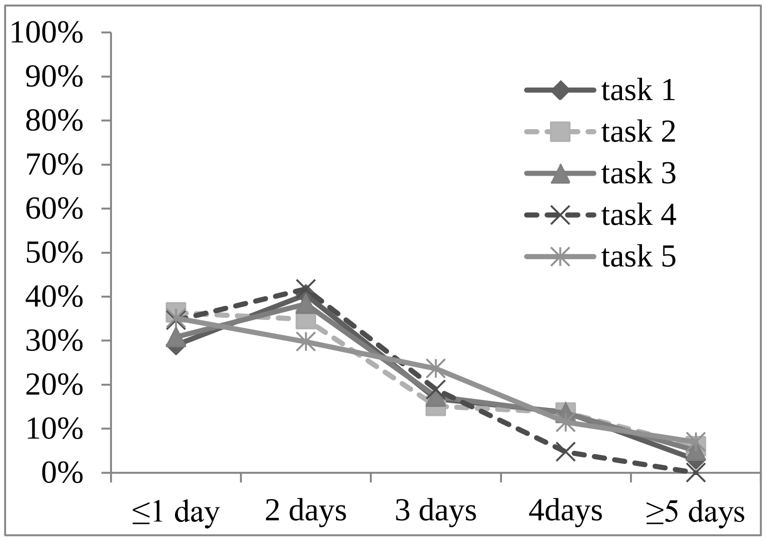

Fig. 3. (b) Completion percentage by number of days, course A supported by CSAMS.

It is hard to say the chess-like peer-evaluation is a certain solution to increase students' performance in quantitative terms, as there is not any absolute standard in measuring these web page designs. However, students' feedback shows when they putting more emphasis on this rating task and care more about the results.

To demonstrate the effectiveness of the Glicko rating results, we divided the students into five equal groups according to rating scores from high to low and then invited a teacher to grade part of students' works. We picked from the first, third, and fifth groups a total of 93 assignments (31 in each group) and scrambles the order. Then the teacher was asked to divide the 93 students into three rating groups according to their marks. Compared with the Glicko rating result, the total error of 9.68% is encouraging. There were five errors between groups 5 and 3, four errors between groups 3 and 1, and no errors between groups 5 and 1. Considering the teacher's personal preferences, we believe that the error is acceptable.

Eliminating Procrastination

Once an assignment had been given out, students could obtain the requirements and submit their work within the specified time through the CSAMS website. The system would automatically score each job as it was submitted, with earlier submissions receiving higher scores. This measure greatly encouraged students to complete their tasks as soon as possible.

Figure 3 shows the completion status of the five tasks for each day after each assignment was given out for course A "Algorithm Design and Analysis" over two semesters. Without CSAMS, the average cumulative completion percentage after the first three days was only 31.7%, and 54% of the students needed five days or more to cope with an exceptionally challenging task. In contrast, Figure 4 (b) shows a different result for the same course, but with support by CSAMS in the later semester. In this case, 70.2% of the students finished their work within two days, and the completion percentage after the first three days reached 89.6%. Moreover, the relatively high completion percentage in the first few days effectively reduced the incidence of plagiarism, which is a significant effect of competition.

Another course B "Web Design" with easier tasks (six tasks without final assignment) provides further evidence for this effect. Before autoscoring, the completion percentage for the first day was 33.8%, and the cumulative percentage was 60.7% after the first three days. By contrast, with support by CSAMS, the completion percentage was 92.7% in the first day. This significant change is shown in Figure 4.

Avoiding Plagiarism

The system runs plagiarism detection on every submission and returns a warning to the student if the maximum similarity is higher than a threshold. Once all assignments have been submitted, all-pairs detection will continue (until the system is idle). The high-similarity pairs which are over tolerance will be published to a list. The publicity is a strong warning to those who want to get something for nothing that they will be identified as a plagiarist by classmates.

The two courses mentioned above (courses A and B) were also involved in examining the effect of opening the plagiarism detection process. Before similarity detection, varying degrees of plagiarism occurred in both courses, and serious plagiarism (defined as similarity > 0.9) was identified for 6% of the students. However, since the students have faced the prospect of being identified publicly for plagiarizing, plagiarism has basically been controlled to an acceptable range. Figure 5 illustrates the change.

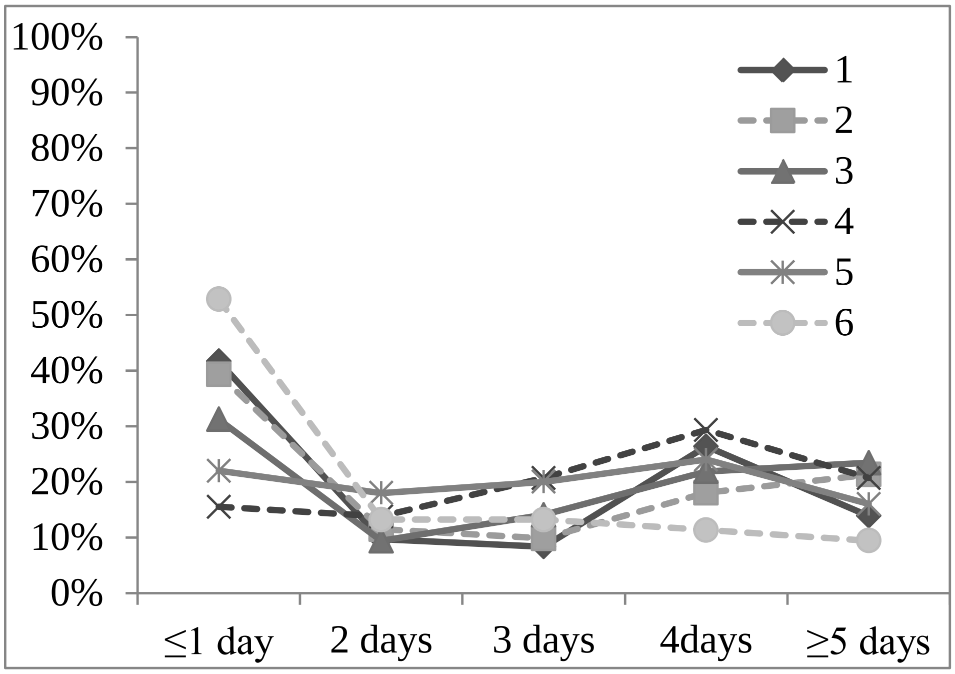

Fig. 4**.** (a) Completion percentage by number of days, course B without CSAMS (with easier tasks);

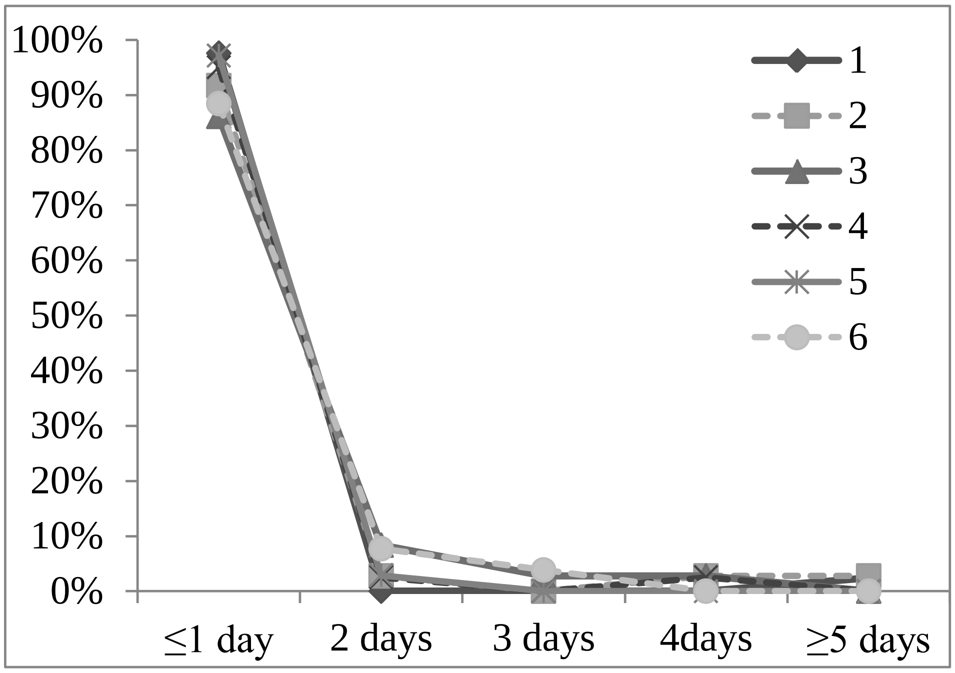

Fig. 4. (b) Completion percentage by number of days, course B supported by CSAMS.

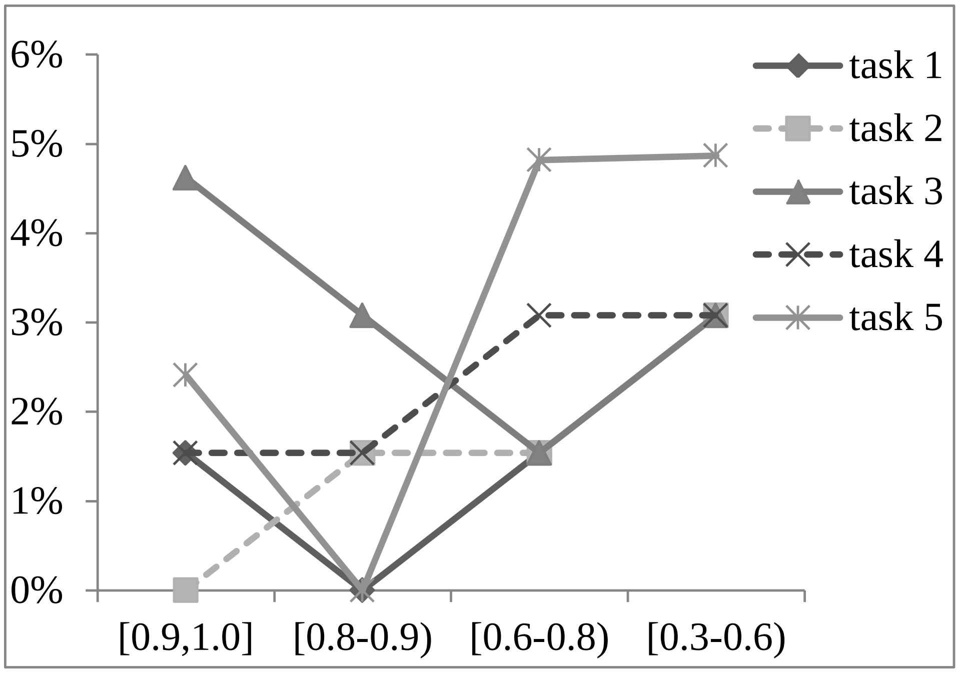

Fig.5. (a) Suspected plagiarism percentages expressed as similarity ranges, course A without CSAMS;

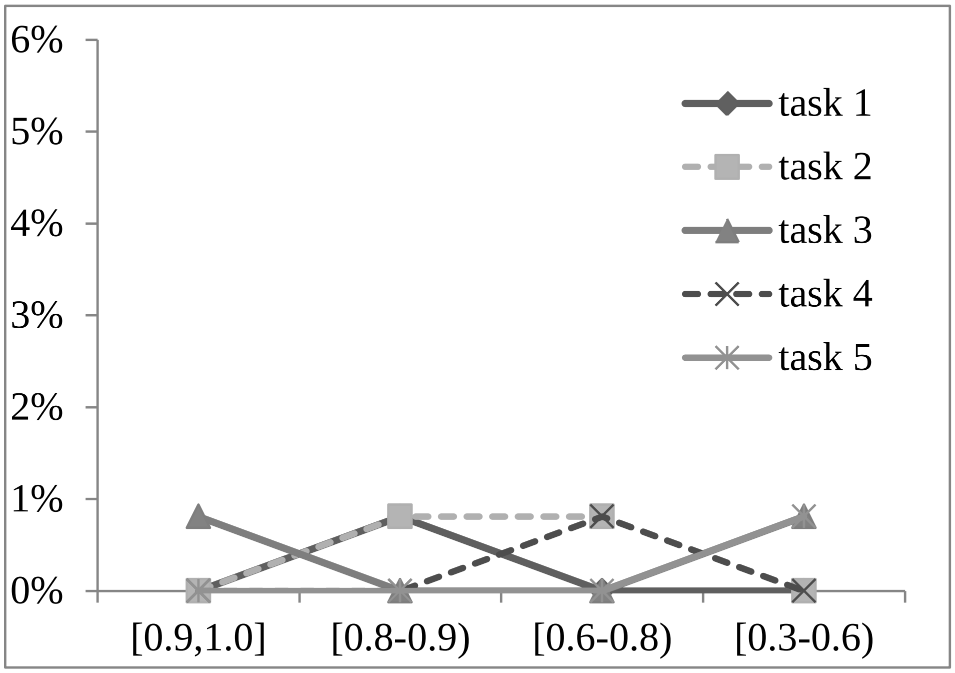

Fig.5. (b) Suspected plagiarism percentages expressed as similarity ranges, course A supported by CSAMS;

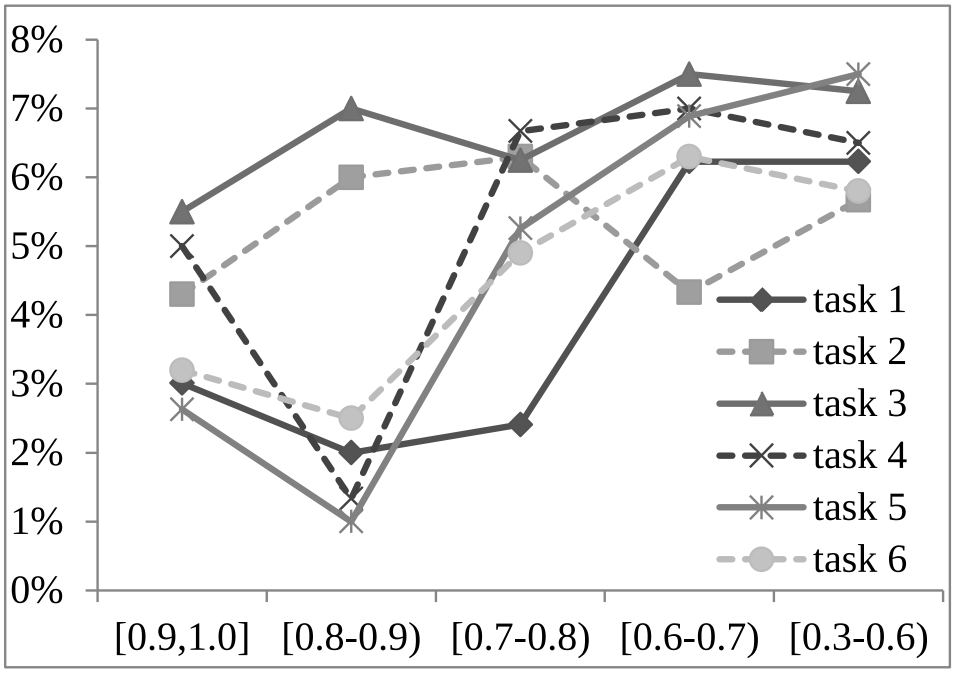

Fig.5. (c) Suspected plagiarism percentages expressed as similarity ranges, course B without CSAMS;

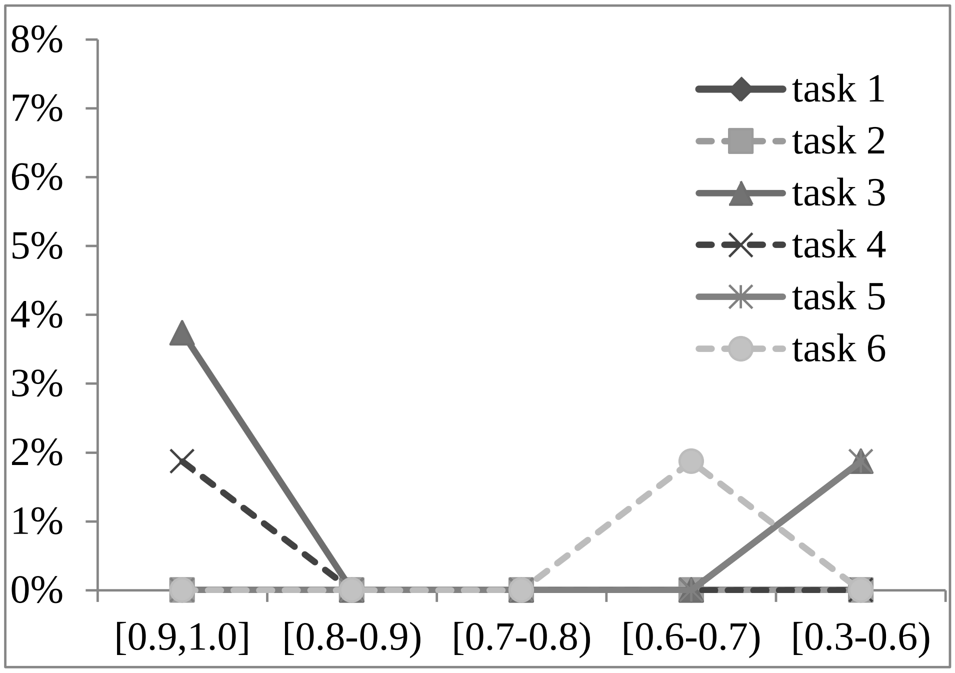

Fig.5. (d) Suspected plagiarism percentages expressed as similarity ranges, course B supported by CSAMS.

It should be noted that the above comparison is not very precise because the numbers of students in a course in different semesters are not exactly the same. For example, in a class of 100 students, a specific plagiarist will contribute only 1% to the plagiarism rate, but if the class size is 30, the rate will increase to 3%. Therefore, these results represent more general trends than accurate values.

Conclusions and Further Work

This research aimed to develop a competition-based system to heighten the sense of urgency for students in finishing their assignments. For this purpose, the TopCoder programming-time competition model was introduced to eliminate procrastination. An innovative method has also been proposed for students to evaluate their own assignments using the Glicko-2 chess rating algorithm. As for plagiarism detection, the well-known winnowing algorithm was used to generate document fingerprints, and as well, similarity measurements and two comparison methods were used as discussed in this paper.

Test results and comparative analysis have shown that CSAMS has showed positive effects on students' learning in the following three ways: 1) encourage students to work more efficiently by time competition, 2) help students work with integrity by plagiarism detection and 3) get students more involved and learn from their peers by an easy and interesting game-like peer-evaluation. Use of the system provides an opportunity for professors to spend more time and effort in preparing fascinating lessons rather than constantly urging students to complete assignments and confronting a large volume of suspected plagiarism without knowing how to make a fair evaluation.

Further research will involve applying the system to more courses and exploring the feasibility of ratings-based peer-evaluation for a greater variety of assignments. On the topic of time competition, the authors are now investigating intelligent automatic scoring of assignments not only according to the time taken, but also according to other given criteria. Some success has been achieved in borrowing ideas from many online judging systems, and in any case, what was intended here was not merely correctness of programs and algorithms, but also the development of intelligent evaluation methods for more universal digital course assignments.

References

1. Beswick, G., E.D. Rothblum, and L. Mann, *Psychological antecedents of student procrastination.* Australian Psychologist, 1988. **23**(2): p. 207-217.

2. Walker, J., *Student Plagiarism in Universities: What are we Doing About it?* Higher Education Research & Development, 1998. **17**(1): p. 89-106.

3. Austin, M.J. and L.D. Brown, *Internet Plagiarism: Developing Strategies to Curb Student Academic Dishonesty.* The Internet and Higher Education, 1999. **2**(1): p. 21-33.

4. Sheard, J., et al., *Cheating and plagiarism: perceptions and practices of first year IT students*, in *Proceedings of the 7th annual conference on Innovation and technology in computer science education*2002, ACM: Aarhus, Denmark. p. 183-187.

5. Hsiao, I.H., et al., *Open Social Student Modeling: Visualizing Student Models with Parallel IntrospectiveViews*, in *User Modeling, Adaption and Personalization*, J. Konstan, et al., Editors. 2011, Springer Berlin Heidelberg. p. 171-182.

6. Vassileva, J., *Toward Social Learning Environments.* IEEE Transactions on Learning Technologies, 2008. **1**(4): p. 199-214.

7. Wolfe, W.J., *Online student peer reviews*, in *Proceedings of the 5th conference on Information technology education*2004, ACM: Salt Lake City, UT, USA. p. 33-37.

8. Blum, S.D., *My Word! Plagiarism and College Culture*. 2009: NY: Cornell University Press.

9. Evans, R., *Evaluating an electronic plagiarism detection service The importance of trust and the difficulty of proving students don't cheat.* Active Learning in Higher Education, 2006. **7**(1).

10. Butakov, S. and V. Scherbinin, *The toolbox for local and global plagiarism detection.* Computers & Education, 2009. **52**.

11. Kurnia, A., A. Lim, and B. Cheang, *Online Judge.* Computers &amp; Education, 2001. **36**(4): p. 299-315.

12. Jones, E.L., *Grading student programs - a software testing approach.* J. Comput. Small Coll., 2000. **16**(2): p. 185-192.

13. Grefenstette John, J.K., Spears William, *Competition-Based Learning Foundations of Knowledge Acquisition*, A.L. Meyrowitz and S. Chipman, Editors. 1993, Springer US. p. 203-225.

14. Gárcia-Mateos, G., *A course on algorithms and data structures using on-line judging.* SIGCSE Bull., 2009. **41**(3): p. 45-49.

15. Lin, K.-C., T.-K. Wu, and Y.-B. Wang, *Developing a Web-based and Competition-based Quiz Game Environment to Improve Student Motivation* Journal of Networks, 2011. **6**(5): p. 736-742.

16. Cheng, R. and J. Vassileva, *Design and evaluation of an adaptive incentive mechanism for sustained educational online communities.* User Modeling and User-Adapted Interaction, 2006. **16**(3-4): p. 321-348.

17. TopCoder. *Competing in a Rated Algorithm Competition*. 2010 2010-12-23 [cited 2012 2012-12-16]; Available from: http://apps.topcoder.com/wiki/display/tc/Competing+in+a+Rated+Algorithm+Competition.

18. Miao, Y. and R. Koper, *An efficient and flexible technical approach to develop and deliver online peer assessment*, in *Proceedings of the 8th iternational conference on Computer supported collaborative learning*2007, International Society of the Learning Sciences: New Brunswick, New Jersey, USA. p. 506-515.

19. Wikipedia. *Elo rating system*. 2013 2013-1-17 [cited 2013 2013-1-18]; Available from: http://en.wikipedia.org/wiki/Elo_rating_system.

20. Glickman, M.E. and A.C. Jones, *Rating the chess rating system.* Chance, 1999. **12**(2): p. 21-28.

21. Glickman, M.E., *Paired comparison models with time-varying parameters*, in *Department of Statistics*,1993, Harvard University.

22. Glickman, M.E., *Parameter estimation in large dynamic paired comparison experiments.* Applied Statistics, 1999. **48**.

23. Glickman, M.E. *Professor Glickman's Glicko-Website*. 2011 [cited 2011 2011-12-16]; Available from: http://www.glicko.net/glicko.html.

24. Peterson, C.H. and N.A. Peterson, *Impact of peer evaluation confidentiality on student marks.* International Journal for the Scholarship of Teaching and Learning, 2011. **5**(2): p. 13.

25. Pope, N.K.L., *The impact of stress in self- and peer assessment.* Assessment & Evaluation in Higher Education, 2005. **30**(1): p. 51-63.

26. Martin, D.F., *Plagiarism and Technology: A Tool for Coping With Plagiarism.* Journal of Education for Business, 2005. **80**(3): p. 149-152.

27. Schleimer, S., D.S. Wilkerson, and A. Aiken, *Winnowing: local algorithms for document fingerprinting*, in *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*2003, ACM: San Diego, California. p. 76-85.

28. Parapar, J. and Á. Barreiro, *Winnowing-based text clustering*, in *Proceedings of the 17th ACM conference on Information and knowledge management*2008, ACM: Napa Valley, California, USA. p. 1353-1354.

29. Roy, C.K., J.R. Cordy, and R. Koschke, *Comparison and evaluation of code clone detection techniques and tools: A qualitative approach.* Science of Computer Programming, 2009. **74**(7): p. 470-495.

30. Wikipedia. *Vector Space Model*. 2013 2013-1-12 [cited 2013 2013-1-16]; Available from: http://en.wikipedia. org/wiki/Vector_Space_Model.

31. Wikipedia. *Cosine Similarity*. 2012 2012-11-4 [cited 2012 2012-12-16]; Available from: http://en.wikipedia.org/ wiki/Cosine_similarity.

32. wikipedia. *Tf-idf*. 2011 2013-1-8 [cited 2013 2013-1-16]; Available from: http://en.wikipedia.org/wiki/Tf-idf.

Biographical Information

Xuliang Duan received the B.S. degree in 2005 and the M.S. degree in 2008 from School of Information Science & Technology of Beijing Forestry University, China. He joined Sichuan Agricultural University in 2008 and currently is a lecturer at the Department of Computer Science. His main research interests include web engineering, natural language processing, information retrieval and learning technologies.

Chao Wang is a professor at the Sichuan Agricultural University, China. He is the department chair of the Computer Science Department as well as the vice-chairman of the Sichuan Basic Computer Education Research Society and the member of Computer Rank Examination Committee of Sichuan province. His research interests include the application of information technologies in education, computer-assisted instruction and mathematics learning.