Ranking students with help of mechanized grading

Christian Queinnec

▶ **To cite this version:**

Christian Queinnec. Ranking students with help of mechanized grading. 2010. hal-00671884

HAL Id: hal-00671884
https://hal.science/hal-00671884

Preprint submitted on 19 Feb 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

RANKING STUDENTS WITH HELP OF MECHANIZED GRADING

Queinnec Christian
LIP6, Universit Pierre et Marie Curie, France
Christian.Queinnec@upmc.fr

Keywords: Mechanized grading, Ranking system

Abstract: Around 2000, we started to propose to students exercises with mechanized grading. Since, we have been accumulating lots of data that confirm the evidence: the more the students practice, the better they perform at examinations!
Therefore, to foster the use of these exercises, we devise a ranking system where every student may compare his skill to the others. As for video games, we expect students to be attracted by this feedback. Computing these skills also have some interesting side-effects mainly on the suggestion of interesting new exercises.

1 INTRODUCTION

We strongly believe in mechanized grading as part of the devices proposed to students to train themselves and improve their knowledge. Mechanized grading also corresponds to the industrial practice known as test driven programming which is worth exposing to students. Mechanized grading easily supports courses with huge number of students and, furthermore, introduce "dynamic annals" (**?**) that is, examinations that may be taken, years after their inception, with the same original conditions.

Around 2000, we started to offer, in various programming development environments (Scheme (**?**), Shell (**?**), etc.), exercises and even examinations with mechanized grading. Our policy remains constant all these years: these exercises are not mandatory nor they count for the final mark. They are only used by volunteering students.

In 2006, we started to build an infrastructure for the deployment of these exercises (**?**) and, since, collect lots of data. When analyzing these data, the main correlation we found was evident: the more the students practice, the better they perform on examinations! Hence our desire to promote the use of these exercises.

To improve our students' motivation, we thought to games and their associated ranking systems: chess (ELO, Glicko (Glickman, 1995)), TrueSkill™ for video games (Graepel et al., 2007), etc. Knowing that you have a rank, that performing exercises evolve your rank and, finally, see where your rank stands among others' ranks is thought to be a clear incentive.

After describing some ranking systems and their appropriateness for our context in Section 2, we describe our own ranking system in Section 3. The experiment we set up for this semester is covered in Section 4. Section 5 analyze related works while the final Section 6 lists some possible outcome of our system.

2 RANKING SYSTEMS

Arpad Elo developed, in 1959, a rating system for chess. The ELO system characterizes every player with a number (the ELO rating). When two players of skill s_1 and s_2 (suppose that $s_1 > s_2$) play together, they exhibit performance p_1 (resp. p_2). These performances are supposed to be normally distributed around s_1 (resp. s_2) with some fixed variance. The first player is expected to win over the second player with a probability that is a function of the difference of skills $(s_1 - s_2)$. After the game, skills are updated to make the outcome of the game more likely. In the ELO system, the sum of skills stay constant and the

winner's skill grows by an amount computed according to the "level of surprise" introduced by the outcome of the game: if the result is surprising given the initial skills, then the transfered amount of skills is bigger than if the result is not surprising.

The ELO system has a number of problems (problems which are addressed, for the chess domain, by additional rules). We only cite those that will be of concern for us: what is the initial skill of a new player ? Why is the variance of performance the same for all players ? How to prevent general deflation or inflation of the skills ?

More elaborated system such as Glicko (Glickman, 1995), addresses these problems while keeping the general ideas the same. However Glicko characterizes players with two numbers: a mean and a standard deviation identifying a Gaussian (a bell curve) along which performance is supposed to be distributed. The sum of skills is no longer a constant.

TrueSkill™, introduced by Microsoft for video games, characterizes players similarly with a mean and a standard deviation. TrueSkill computes skills for every player even if they play in teams where the observed results are "team A beats team B". The main use of players' skills is to propose opponents with equivalent skills in order to let players run uncertain games that is, more fun games. An interesting corollary is that the result of an uncertain game maximizes the knowledge about the skills of the involved players.

In TrueSkill, the initial skill of a new player is $(50, 50/3)$. The mean is 50, half way from 0 and 100 and the standard deviation is $50/3$. A skill (μ, σ) states that the observed performances will be in the range of μ plus or minus 3σ with a level of confidence of 99%. Therefore the initial $(50, 50/3)$ states that the skill of the player is almost surely between 0 and 100. When displayed to the player, a skill is shown as $\mu - 3\sigma$: a conservative estimation.

2.1 Revisiting the past

Our mechanized grading infrastructure collects, in a database, the results of all the performed grading. These records are 4-tuples (student, exercise, date, mark). In order to relate our records to game play, we assume that

> When a student gets a mark on an exercise e, this student beats all the students who attempted the same exercise e and got a smaller mark and, at the same time, is beaten by all the students who attempted the same exercise e and got a bigger mark.

We implemented the Glicko and TrueSkill algorithms and analyzed our collected data. The proposed exercises accompany a course on Unix (shell and make). We considered a first set of 79 students who used the platform 992 times against 39 exercises between September 2008 and January 2009 then a second set of 127 students who used 1242 times the platform against the same set of exercises between September 2009 and January 2010. For these two sets, we also have the marks the students got at the final examination.

Let us give some indications on these data sets. Most of the students use the platform just a few time (1 to 6 attempts on 1 or 2 exercises) and quickly stop after satisfying their initial curiosity while some others succeed to solve up to 35 exercises, see Figure 1 (drawn with Weka (Hall et al., 2009)). Quite often also, we note that a number of students attempt exercises in burst especially in the week that precede examinations.

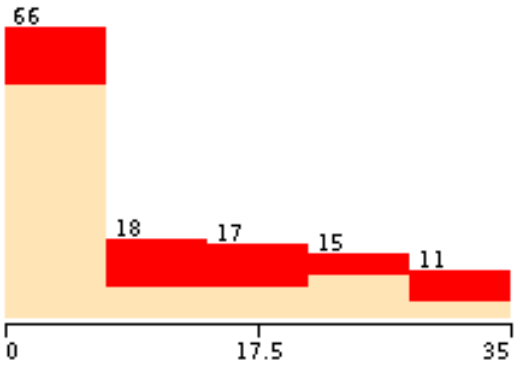

Figure 1: Histogram of the number of students having attempted (but not necessarily solved) a number of exercises. 66 students have attempted between 0 and 7 exercises while 11 have attempted more than 28 exercises. The red color identifies the students that succeed the final examinations.

We first run the Glicko algorithm and compare the obtained skills with the marks of the final examination. The correlation was poor. We then run the TrueSkill algorithm and once again found the correlation to be poor. Even if one attempt of an exercise is worth a number of games against all the students that attempted this exercise before, rare players cannot be given useful estimations thus plaguing other estimations.

We therefore conclude that our estimation of skill, following these lines, was on a wrong path. The only evident correlation was between the number of solved exercises and the final mark. Briefly stated, the more you practice, the better you are!

We also realized that our goal was not to accurately predict success from skill but to provide an in-

centive for practice thus, indirectly, to favor success. We therefore devise the Rango ranking system.

3 RANGO: A RANKING SYSTEM

We compare the results of two students on a given exercise as follows. The student who has a strictly better mark (m) than the other, wins. In case of equal marks, the one that got that mark in the smaller number of attempts (n), wins. Otherwise, equal marks and equal number of attempts qualify as a draw and no one wins.

$$(m,n) < (m',n') \equiv m < m' \vee (m = m' \wedge n < n')$$

Our database is made of 4-tuples (student, exercise, date, mark) that is, (s,e,t,m). We define the "history" of the skill of a student s on an exercise e as the sequence of 4-tuples ordered by increasing dates.

$$\text{history}(s,e) = \{(s,e,t_i,m_i) \mid \forall (i,j), i < j \implies t_i < t_j\}$$

We define the "score" of a student s on an exercise e at time t as the pair formed by the greatest mark he got on that exercise before t and the smallest number of attempts he made before getting that greatest mark. The history makes this computation easy.

$$\text{score}(s,e,t) = (m_i,i) \mid \forall j, \begin{array}{l} t_j < t_i \leq t \implies m_j < m_i \\ t_i < t_j \leq t \implies m_j \leq m_i \end{array}$$

If a student solves an exercise i.e., gets the maximal mark, then the score is unchanged by any further attempts (some students try varying methods to solve the same exercise and they should not be penalized for their curiosity). If the student have not yet fully solved the exercise then the score can only grows up.

When a student s tries, for the 1st time, to solve an exercise e and got mark m then, his score is a pair $(m,1)$. From now on, we normalize marks to always be between 0 and 1, 1 being the maximal possible mark.

To rank students, we may observe that the best possible student (BPS) is simple to define: this student solves any exercise in a single attempt, its score on any exercise is always $(1,1)$. The worst possible student (WPS) may also be defined as the one who systematically fails any exercise in at least one more attempt than any other real student. All real students fit between BPS and WPS.

$$\forall s,e,t, \text{score}(\text{BPS},e,t) \geq \text{score}(s,e,t) > \text{score}(\text{WPS},e,t)$$

For any score on a given exercise e, we define the "rank" of the student for this exercise as the number of students who have a strictly better score on this exercise. The BPS has a rank of 0 for any exercise.

$$\text{rank}(s,e,t) = \text{card}\{s' \mid \text{score}(s',e,t) > \text{score}(s,e,t)\}$$

We then define the "skill" of a student as the percentage of the sum of his ranks on all exercises divided by the sum of the ranks of WPS.

$$\text{skill}(s,t)/100 = \frac{\Sigma_e \text{rank}(s,e,t)}{\Sigma_e \text{rank}(\text{WPS},e,t)}$$

Skills are expressed as a number between 0 and 100. BPS has a skill of 100 while WPS has a skill of 0. Students that never play are assimilated to the WPS. The more exercises you solve, the closer to BPS you are that is, the highest skill you get. Since the number of solved exercises is a predictor of success to the final examination so is your skill.

As a final note, observe that the Rango ranking system needs the set of students and the set of exercises to be defined for the computation. However new exercises and new students may be added dynamically to the system without disturbance.

4 EXPERIMENT

Our platform for mechanically graded exercises is mainly put to work via web applications. Every night, ten seconds are required to recompute all skills. Students may obtain the history of their skill and see where there are comparatively to the other students, see Figure 2.

In the experiment we set-up for this semester we do not reveal the identity of the other students, only the distribution of skills. Identity is not even displayed on the page; students are encouraged to define their own pseudo and to associate a gravatar (Gravatar, 2007) to their email address. We envision to interview the students to see if displaying pseudos and/or gravatars on the skills distribution would foster more competition.

We have several different goals with this experiment:

1. we want to discover if the addition of skills increase the use of mechanically graded exercise,
2. we want to analyze again final skills against final examination marks.

We will report on these goals in the final version of the paper.

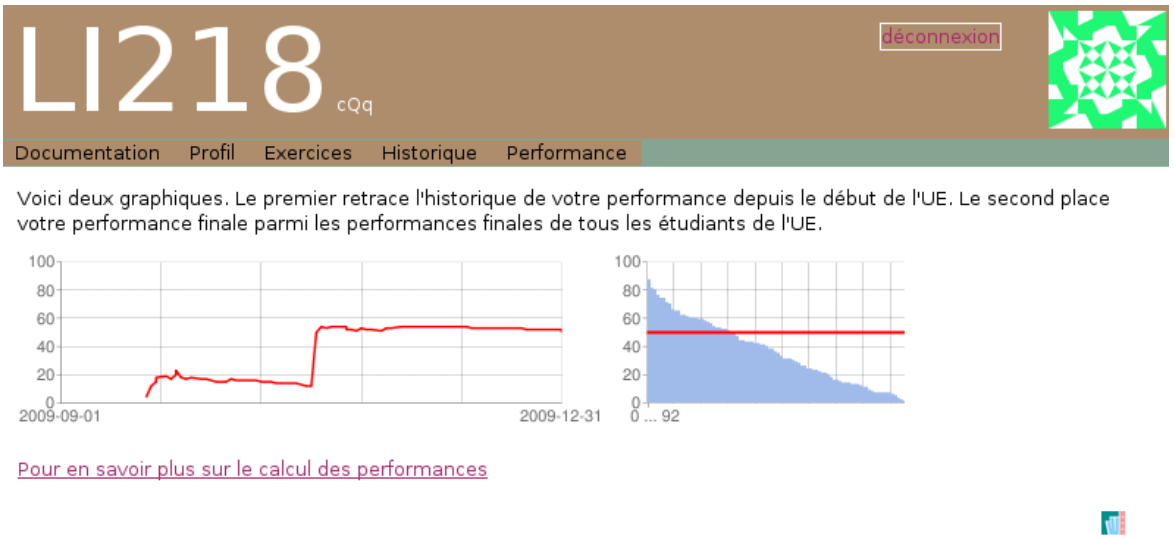

Figure 2: The page of the web application displaying skill. To the left is the history of the skill of the requester. The abrupt skill change is due to the occurrence of a partial examination inducing an incentive to practice. To the right is the final skill of the requester compared to all other final skills. In the upright corner is the gravatar of the student.

5 RELATED WORK

To exploit the logs to infer students' skills is an old idea (Heiner et al., 2004). Many works exist that try to characterize the student's model that is, its shape and its parameters (Jonsson et al., 2005) (Cen et al., 2006). They often start from an analysis relating exercises and the involved primitive skills then, they observe students' progress (mining the logs) in order to determine the parameters that best fit the model mainly with the "expectation maximization" technique (Ferguson, 2005).

The previous studies use far more information than us since they mine the logs of an intelligent tutor system where are recorded which exercise is delivered, how long the student read the stem, what help he requires, etc. By contrast, our grading infrastructure only gives us access to marks. Our set of proposed exercises is not (yet) related to the involved skills nor the set of skills is clearly stated. Therefore we are more interested in fostering the use of exercises with an attractive but rigorous feedback.

6 POSSIBILITIES AND FUTURE WORK

Besides our current experiment described in the previous Section, we envision another use of the skills inspired by video games. Skills may be used to select "interesting" exercises that is, exercises for which the uncertainty of solving them is maximal. Currently, exercises are presented by topic and topics are related, week after week, to the associated lectures. If more and more exercises are added, the selection of new exercises may be suggested instead by the system itself. Given a skill, some exercises may be too easy or too hard. Interesting exercises are the ones in-between.

Conversely exercises may be themselves ranked with respect to the distribution of the skills of the students who solved them (or not). This may be seen as part of the supervision tools watching the progress of the students through the proposed exercises.

Glicko and TrueSkill maintain mean and deviation to characterize players while our system only uses one number. We plan to study if introducing a deviation will improve the prediction of the final mark.

7 CONCLUSIONS

In this paper, we propose an analogy between games play and practice of exercises. After experimenting with well-known ranking systems, we define a new ranking system for students based on the marks they obtain on the exercises they attempt. We then briefly described the experiment we set-up where elaborated skills are fed back to students as an incentive to practice more. Finally, we propose some ideas to further the use of skills.

REFERENCES

Cen, H., Koedinger, K., and Junker, B. (2006). Learning factors analysis - a general method for cognitive model evaluation and improvement. In *Paper presented at the 8th International Conference on Intelligent Tutoring Systems*, pages 164–175.

Ferguson, K. (2005). Improving intelligent tutoring systems: Using expectation maximization to learn student skill levels.

Glickman, M. (1995). The Glicko system. Technical report, Boston University. http://glicko.net/glicko.doc/glicko.html.

Graepel, T., Herbrich, R., and Minka, T. (2007). TrueSkill™: A bayesian skill rating system. Technical report, Microsoft. http://research.microsoft.com/en-us/projects/trueskill/.

Gravatar (2007). A globally recognized avatar. Technical report. http://gravatar.com/.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA data mining software: An update. *SIGKDD Explorations*, 11(1).

Heiner, C., Beck, J., and Mostow, J. (2004). Lessons on using its data to answer educational research questions. In *Workshop Proceedings of ITS-2004*, pages 1–9.

Jonsson, A., Johns, J., Mehranian, H., Arroyo, I., Woolf, B., Barto, A., Fisher, D., and Mahadevan, S. (2005). Evaluating the feasibility of learning student models from data. In *Proceedings of the Workshop on Educational Data Mining at AAAI-2005*, pages 1–6. MIT/AAAI Press.

Kazanidis, I. and maya Satratzemi (2010). Modeling user progress and visualizing feedbak. In *CSEDU 2010 – Proceedings of the second International Conference on Computer Supported Education*, volume 1, pages 46–53, Valencia, Spain.