Integrating IRT analysis into LMS for item pool optimization

Integrating IRT Analysis into LMS for Item Pool Optimization

Panagiotis Fotaris[1], Theodoros Mastoras[2]

[1] Dental Institute, King's College London, London, UK
panagiotis.fotaris@kcl.ac.uk
[2] Department of Applied Informatics, University of Macedonia, Thessaloniki, Greece
mastoras@uom.gr

Abstract. Due to the computerization of assessment tests, the use of Item Response Theory (IRT) has become commonplace for educational assessment development, evaluation, and refinement. When used appropriately by a Learning Management System (LMS), IRT can improve the assessment quality, increase the efficiency of the testing process, and provide in-depth descriptions of item and test properties. This paper introduces a methodological and architectural framework which embeds an IRT analysis tool in an LMS so as to extend its functionality with assessment optimization support. By applying a set of validity rules to the statistical indices produced by the IRT analysis, the enhanced LMS is able to detect several defective items from an item pool which are then reported for reviewing of their content. Assessment refinement is achieved by repeatedly employing this process until all flawed items are eliminated.

Keywords: e-learning, Item Pool Optimization, Computer Aided Assessment, Item Analysis, Massive Open Online Courses, MOOCs, Item Response Theory, IRT, Learning Management System, Technology Enhanced Learning.

1 Introduction

Due to the recent advances in Internet technologies and the booming development of massive open online courses (MOOCs), the use of Computer Aided Assessment (CAA) tools has become a major trend in academic institutions worldwide [1]. Through these systems, tests composed of various question types can be presented to students in order to assess their knowledge [2]. However, there has been considerable criticism of the test quality, with both research and experience showing that many test items are flawed at the initial stage of their development. Test developers can expect about 50% of the items in their item pool to fail to perform as intended, which may eventually lead to unreliable results of examinee performance [3]. Thus a critical challenge lies in how to ensure that the individual test items are of the highest quality possible since an inferior item could have an inordinately large effect on some scores.

The present paper introduces a comprehensible way to present IRT analysis results to test developers without delving into unnecessary details. Instead of memorizing numerous commands and scenarios from technical manuals, test developers can easily

detect problematic questions from the familiar user interface of an LMS. The latter can automatically calculate the limits and rules for the α (discrimination), b (difficulty), and c (guessing) parameters [4] based on the percentage of questions wanted for revision. The examinee's proficiency (θ) is represented on the usual scale (or metric) with values ranging roughly between -3 and 3, but since these scores include negative ability estimates which would undoubtedly confuse many users, they can optionally be normalized to a 0…100 range scale score.

2 Related Work

Students' increasing demand for more flexible learning options during the last decade has led to the widespread use of LMS and CAA tools in education, and, more recently, to the rapid expansion of MOOCs distributed in platforms such as Coursera, Udacity, and EdX. However, there is serious concern around the assessment of student learning due to the fact that only a small fraction of the aforementioned systems supports an assessment quality control process based on the interpretation of item statistic parameters. Popular e-learning platforms such as Moodle and Blackboard have plug-ins or separate modules that provide statistics for test items, but apart from that they offer no suggestions to test developers on how to improve their item pool. Similarly, although new web technologies allow for scalable ways to deliver video lectures, implement social fora, and track student progress in MOOCs [5], there is limited feedback regarding the quality of the test items and the accuracy of the assessment results. Therefore, many researchers have recently endeavored to provide mechanisms for assessment optimization.

Hsieh et al. introduced a model that presents test statistics and collects students' learning behaviors for generating analysis result and feedback to tutors [6]. Hung et al. proposed an analysis model based on Item Analysis (IA) that collects information such as item difficulty and discrimination indices, questionnaire and question style, etc. [7]. These data are combined with a set of rules in order to detect defective items, which are signaled using traffic lights. Costagliola et al.'s eWorkbook system improved this approach by using fuzzy rules to measure item quality, detect anomalies on the items, and suggest improvements [8]. Nevertheless, all of the aforementioned works preferred IA to IRT due to its ease of use without taking into consideration its numerous deficiencies.

On the other hand, IRT has been mainly applied in the Computerized Adaptive Test (CAT) domain for personalized test construction based on individual ability [9]. Despite its high degree of support among theoreticians and some practitioners, IRT's complexity and dependence on unidimensional test data and large samples often relegate its application to experimental purposes only. While a literature review can reveal many different IRT estimation algorithms, they all involve heavy mathematics and are unsuitable for implementation in a scripting language designed for web development (e.g., PHP). As a result, their integration in internet applications such as LMSs is very limited. A way to address this issue is to have a web page call the open-source analysis tool ICL [10] to carry out the estimation process and then import its results for display. The present paper showcases a framework that follows this exact

method in order to extend an LMS with IRT analysis services at no extra programming cost.

3 Open-source IRT Analysis Tool ICL

Several computer programs that provide estimates of IRT parameters are currently available for a variety of computer environments, including Rascal, Ascal, WINSTEPS, BILOG-MG, MULTILOG, PARSCALE, RUMM and WINMIRA to name a few that are easily obtainable [9]. Despite being the de facto standard for dichotomous IRT model estimation, BILOG is a commercial product and limited in other ways. Hanson provided an alternative stand-alone software for estimating the parameters of IRT models called IRT Command Language (ICL) [10]. A recent comparison between BILOG-MG and ICL [11] showed that both programs are equally precise and reliable in their estimations. However, ICL is free, open-source, and licensed in a way that allows it to be modified and extended. In fact, ICL is actually IRT estimation functions embedded into a fully-featured programming language called TCL that supports relatively complex operations. Additionally, ICL's command line nature enables it to run in the background and produce analysis results in the form of text files. Since the proposed framework uses only a three-parameter binary-scoring IRT model (3PL), ICL proves more than sufficient for our purpose and was therefore selected to complement the LMS for item pool optimization.

4 Integrating IRT Analysis in Dokeos

Dokeos is an open-source LMS implemented in PHP that requires Apache acting as a web server and MySQL as a Database Management System. It has been serving the needs of two academic courses at the University of Macedonia for over six years, receiving satisfactory feedback from both instructors and students. In order to extend its functionality with IRT analysis and item pool optimization functions, we had to modify its source code so as to support the following features:

1. After completing a test session, the LMS stores in its database the examinee's response to each test item instead of keeping only a final score by default.
2. Test developers define the acceptable limits for the following IRT analysis parameters: item discrimination (a), item difficulty (b), and guessing (c). The LMS stores these values as validity rules for each assessment. There is an additional choice of having these limits set automatically by the system in order to rule out a specific percentage of questions (Fig. 1.1).
3. Every time the LMS is asked to perform an IRT analysis, it displays a page with the estimated difficulty, discrimination and guessing parameters for each test item. If the latter violates any of the validity rules already defined in the assessment profile, it is flagged for review of its content (Fig. 1.2). Once item responses are evaluated, test developers can discard, revise or retain items for future use.

4. In addition to a total score, the assessment report screen displays the proficiency θ per examinee as derived from the IRT analysis (Fig. 1.3).

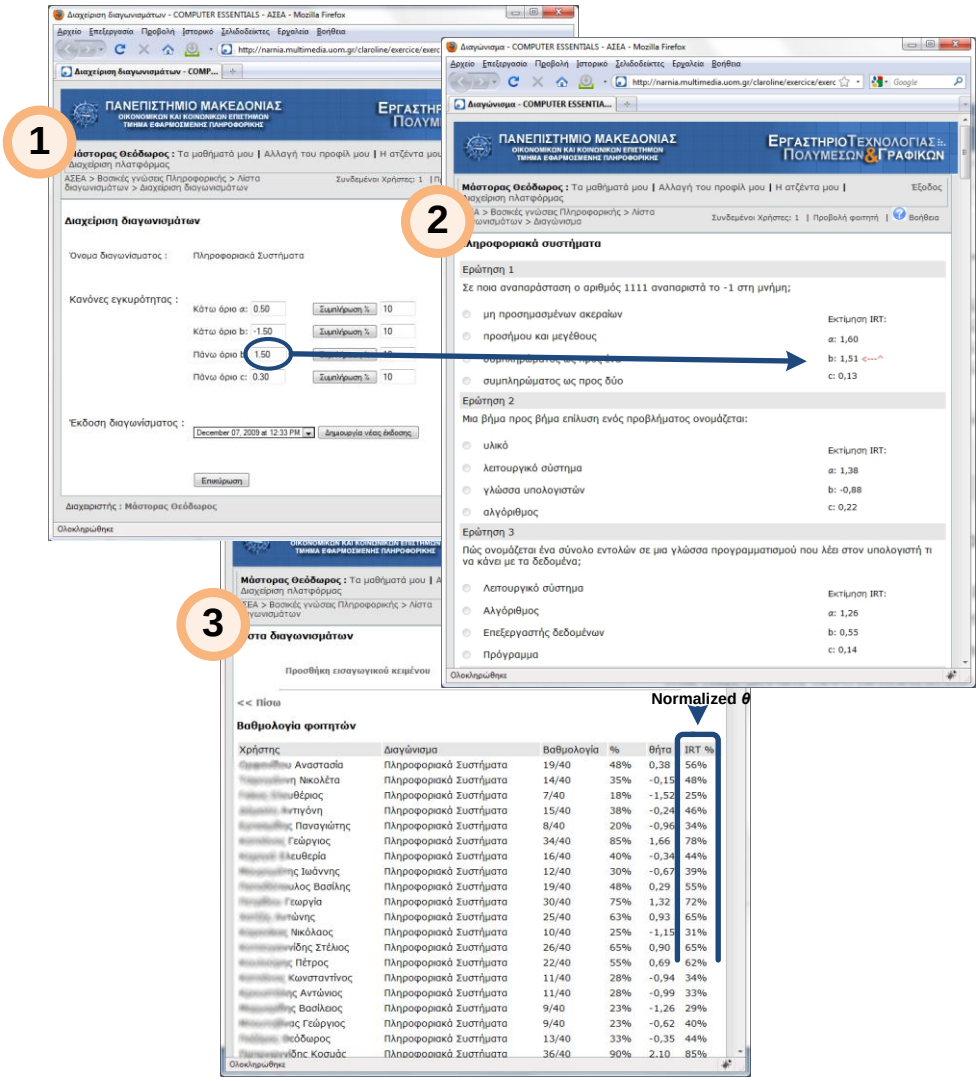

Fig. 1. Functionality features supported in the extended version of Dokeos

The proposed methodology consists of four steps, with each one of them being an action performed by the LMS (Fig. 2). Additionally, the initial database schema has been extended in order to support some extra functions. Once an update of the IRT results is called for, the LMS exports the proper data files and TCL scripts. It then performs a number of calls to the ICL using PHP and after parsing the analysis re-

sults, it imports them to its database. A detailed description of the four methodology steps follows:

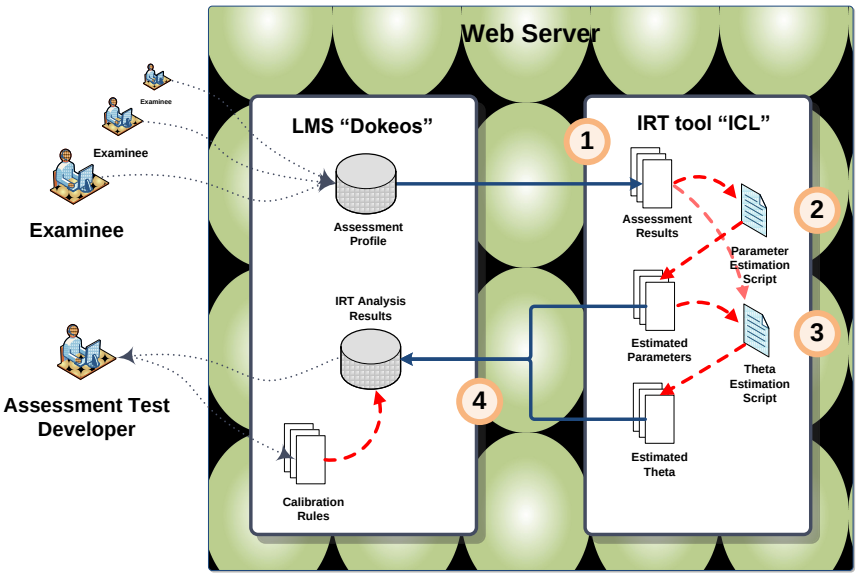

Fig. 2. System architecture

1. The LMS exports the assessment results to a data file and generates a TCL script to process them (parameter estimation script).
2. The LMS then calls up ICL with the parameter estimation script passed as a parameter in order to create a data file containing the α, b, and c values for each test item. At the same time it prepares a second TCL script to process these IRT parameters (θ estimation script).
3. The LMS calls up ICL with the θ estimation script passed as a parameter so as to make a data file with the examinees' θ values.
4. Finally, the LMS imports the two ICL-produced data files (*.par and *.theta) to its database for further processing in the context of the aimed item pool optimization.

As already mentioned, some modifications to the Dokeos database schema had to be performed in order for the system to function properly. More specifically, while the initial schema supported only a total score per examinee ("track_e_exercices" table), the proposed one requires a detailed recording of each examinee's performance per item. The additional functionalities of this new schema are outlined in the following list:

1. Each assessment can have multiple versions based on its revised items. By monitoring the examinees' performance on each item, test developers can determine whether a certain modification of a specific item affected positively its quality. In practice, each version serves as a new test for the LMS.

2. Each examinee's score per item is recorded for every test being administered. These values are held in the assessment results data file (*.DAT) used by ICL.
3. Test developers can establish a new set of rules for each version of the assessment.

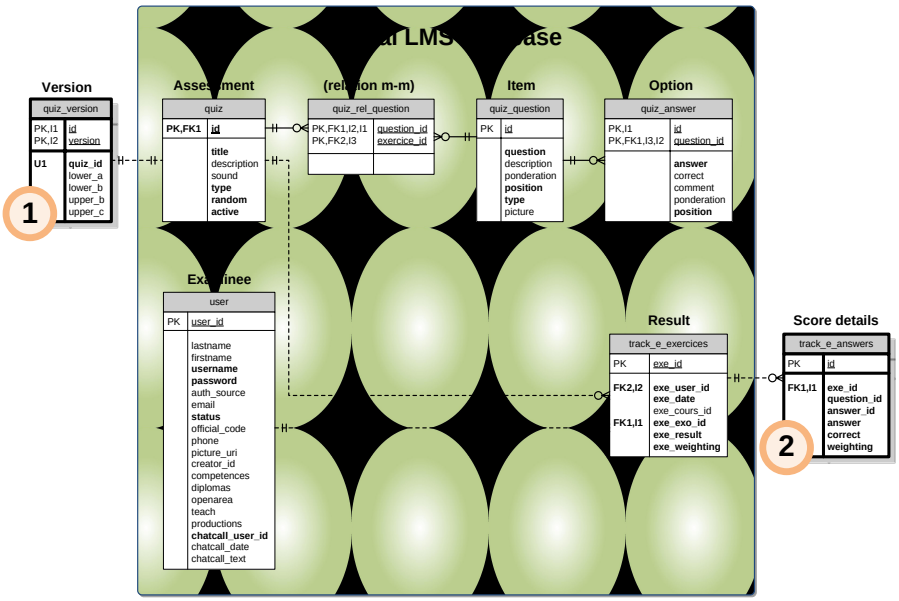

Fig. 3. Entity-Relationship diagram of LMS database extensions

As the main aim of the revised solution is to facilitate further updating processes, the structure and the fields of the initial LMS database have been kept intact, with the only change being the addition of two new tables:

1. Table "*quiz_version*" records each assessment's versions and has a one-to-one relationship to table "*quiz*" (Fig. 3.1).
2. Table "*track_e_answers*" stores the examinee's choice per item (fields "*answer_id*" and "*answer*"), whether this choice was correct (field "*correct*"), and its weight value (field "*weighting*") (Fig. 3.2). Moreover, it supports the recording of multiple responses for future polytomous analyses.

5 Item Pool Optimization Process

The proposed system has been implemented by adding the previous features to an existing version of Dokeos at the Department of Applied Informatics, University of Macedonia. A pilot assessment test containing an item pool of 40 questions on "Fundamentals of Information Systems" was arranged for the experiment. Since it was not connected to an actual university course and contained questions of a general nature, it managed to attract the attention of 113 students who voluntarily participated in the

experiment. Before administering the test, the acceptable limits for the IRT parameters were set to $\alpha \geq 0.5$, $-1.7 \leq b \leq 1.7$, and $c \leq 0.25$ respectively.

Once an initial item pool has been optimized, examinees can be tested routinely. Such a programme of testing is likely to generate a need to retire flawed, obsolete, or frequently used items, and to replace these with new ones. The extended LMS under consideration detects these problem areas, thus making it easier for test developers to improve the quality of their tests provided that they investigate these issues further and focus on addressing the root cause of the problem in each case (e.g., obscure or ambiguous phrases, typographic or logical errors, a lack of essential information, etc.). In addition, the LMS allows them to create a new version of the assessment test effortlessly by copying the previous iteration and either correcting or replacing whichever items have been flagged as defective. Subsequently, once the revised examination cycle is completed, a new analysis report will ascertain whether all items conform to the validity rules. The number of times a specific assessment must be repeated before leading to a final version with all the problematic items eliminated relies on the comprehension of the analysis results. The faster test developers identify the actual cause of each problem and come up with an appropriate solution, the fewer the necessary iterations.

6 Conclusion

The present paper introduced a methodological and architectural framework for extending an LMS with IRT–based assessment optimization. Instead of having web developers implement complex IRT estimation algorithms within the LMS, the proposed methodology uses ICL to obtain reliable IRT analysis results. The latter are then automatically imported into the LMS, thus releasing test developers of this burdensome duty. By applying a set of validity rules, the enhanced LMS is able to detect several defective items which are then reported for review of their content. As a result, the suggested approach is capable of assisting test developers in their continuous effort to optimize their item pools. Moreover, the user-friendly interface allows users with no previous expertise in statistics to comprehend and utilize the IRT analysis results.

According to research focused on IRT sample size effects, a great number of examinees are needed to obtain accurate results [12]. For example, Swaminathan and Gifford concluded that about 1,000 examinees are required when using the 3PL model [13]. Such sample size requirements would normally pose a problem for most test developers due to the fact that the number of examinees in academic courses rarely exceeds 150. However, in cases where instructors are only trying to identify items that are either unrelated to the overall score, too easy, or too difficult, reliable results can be produced even for relatively small classrooms [14]. MOOCs, on the other hand, enroll tens of thousands of students which are more than enough to obtain accurate estimates with any IRT model. As a result, the proposed system would be ideally suited for a MOOC environment; optimizing its extensive item pools will improve the quality of assessment of student learning and could possibly drive more institutions to

offer course credit for MOOC completion, thus further expanding the influence of these courses on higher education throughout the world [9].

This initial experiment produced encouraging results, showing that the system can effectively evaluate item performance and therefore increase the overall validity of the assessment process. The fact that the proposed methodology is not limited to Dokeos but can be adopted by different e-learning environments (e.g., Moodle, MOOC platforms etc.) makes it especially suitable for academic use.

References

1. Virtanen, S. (2009) 'Increasing the self-study effort of higher education engineering students with an online learning platform', International Journal of Knowledge and *Learning*, vol. 4, no. 6, pp. 527-538.
2. Hindi, N.M., Najdawi, M.K. & Jolo, H.A.M. (2008) 'An Examination of Assessment Practices in Colleges of Business at Various Middle East Countries', *International Journal of Teaching and Case Studies*, vol. 1, no. 4, pp. 319-332.
3. Haladyna, T.M. (1999) *Developing and Validating Multiple-Choice Test Items (2nd edition)*, Lawrence Erlbaum Associates, Mahwah, New Jersey.
4. Lord, F.M. (1980) *Applications of Item Response Theory to Practical Testing Problems*, Lawrence Erlbaum Associates, Hillsdale, New Jersey.
5. Piech, C., Huang, J., Chen, Z., Do, C., Ng A., & Koller, D. (2013). 'Tuned Models of Peer Assessment', in S. D'Mello, S., Calvo, R. & Olney, A. (eds.), Proceedings of the 6th International Conference on Educational Data Mining (EDM 2013), July 6-9, Memphis, TN, USA.
6. Hsieh, C., Shih, T.K., Chang, W. & Ko, W. (2003) 'Feedback and Analysis from Assessment Metadata in E-learning', in *17th International Conference on Advanced Information Networking and Applications (AINA '03)*, pp. 155-158.
7. Hung, J.C., Lin, L.J., Chang, W., Shih, T.K., Hsu, H., Chang, H.B., Chang, H.P. & Huang, K. (2004) 'A Cognition Assessment Authoring System for E-Learning', in *24th International Conference on Distributed Computing Systems Workshops (ICDCS 2004 Workshops)*, pp. 262-267.
8. Costagliola, G., Ferrucci, F. & Fuccella, V. (2008) 'A Web-Based E-Testing System Supporting Test Quality Improvement', paper presented to Advances in Web Based Learning – ICWL 2007.
9. Meyer, J. P., & Zhu, S. (2013) 'Fair and equitable measurement of student learning in MOOCs: An introduction to item response theory, scale linking, and score equating', *Research & Practice in Assessment*, vol. 8, no. 1, pp. 26-39.
10. Hanson, B.A. (2002) *IRT Command Language (ICL)*. Obtained through the Internet: http://www.b-a-h.com/software/irt/icl/index.html, [accessed 26/6/2013].
11. Mead, A.D., Morris, S.B. & Blitz, D.L. (2007) *Open-source IRT: A Comparison of BILOG-MG and ICL Features and Item Parameter Recovery*, Illinois Institute of Technology, Institute of Psychology, Chicago, Unpublished manuscript. Obtained through the Internet: http://mypages.iit.edu/~mead/MeadMorrisBlitz2007.pdf, [accessed 1/7/2013].

12. Bunderson, C.V., Inouye, D.K. & Olsen, J.B. (1989) 'The Four Generations of Computerized Educational Measurement', in Linn, R.L. (ed.), *Educational Measurement*, Collier Macmillan Publishers, London.
13. Swaminathan, H. & Gifford, J.A. (1983) 'Estimation of Parameters in the Three-parameter Latent Trait Model', in Weiss, D.J. (ed.), *New Horizons in Testing*, Academic Press, New York.
14. Fotaris, P., Mastoras, T., Mavridis, I., & Manitsaris, A. (2011) 'Identifying Potentially Flawed Items in the Context of Small Sample IRT Analysis', *International Journal On Advances In Intelligent Systems*, vol. 4, no. 1&2, pp. 31-42.