Set-theoretical and Combinatorial Instruments for Problem Space Analysis in Adaptive Serious Games

Enkhbold Nyamsuren[1], Han L.J. van der Maas[2], Matthias Maurer[3]

[1First and corresponding Author] *Welten Institute, Open University of the Netherlands, Netherland, Enkhbold.Nyamsuren@ou.nl*

[2] *Department of Psychology, University of Amsterdam, Netherlands, h.l.j.vandermaas@uva.nl*

[3] *Knowledge Technologies Institute, Graz University of Technology, Austria, mmaurer@tugraz.at*

Abstract

The Computerized Adaptive Practice (CAP) system describes a set of algorithms for assessing player's expertise and difficulties of in-game problems and for adapting the latter to the former. However, an effective use of CAP requires that in-game problems are designed carefully and refined over time to avoid possible barriers to learning. This study proposes a methodology and three different instruments for analyzing the problem set in CAP-enabled games. The instruments include the Guttman scale, a ranked order, and a Hasse diagram that offer analysis at different levels of granularity and complexity. The methodology proposes to use quantified difficulty measures to infer topology of the problem set. It is well-suited for serious games that emphasize practice and repetitive play. The emphasis is put on the simplicity of use and visualization of the problem space to maximally support teachers and game developers in designing and refining CAP-enabled games. Two case studies demonstrate practical applications of the proposed instruments on empirical data. Future research directions are proposed to address potential drawbacks.

Keywords: *Computerized adaptive practice, Game difficulty adaptation, Problem space analysis, Ranked order, Hasse diagram;*

1. Introduction

An effective design pattern for serious games is to have a set of re-playable problems of varying difficulty [1]. Such design promotes the acquisition of new knowledge via a gradual increase in difficulty and knowledge consolidation via practice. Furthermore, the design is easy to integrate with activities both inside and outside of classrooms.

However, the design imposes two major challenges during its development and use. First, to improve learning efficiency, an adaptive mechanism is necessary to detect player's skill level and administer a problem of matching difficulty. For this purpose, [2] proposed the Computerized Adaptive Practice (CAP) algorithm that assesses problem difficulty and player skill to match the former to the latter. Compared to other methods [3], CAP is simpler and easier to use by game developers.

Second, to offer a coherent and smooth learning experience to players, the problem set should be carefully designed and refined during the game's lifetime. However, teachers and developers may lack access to domain experts who can design and maintain a proper set of problems for the game. Therefore, there is a need for a formal method that can facilitate analysis and refinement of the problem set. Such method should infer implicit dependencies among problems composing the set, organize them into a structure coherently presenting the dependencies, and offer insight into how these dependencies may define player's learning experience. Importantly, the method should be accessible to teachers and game developers who may lack technical knowledge for complex analytics.

To address the second issue, we propose a methodology to automatically infer problem dependencies from the CAP algorithm's psychometric measures of difficulty. The proposed methodology is further elaborated into a set of analytical instruments incorporating the Guttman scale [4], a ranked order, and a Hasse diagram [3,5-6]. These instruments offer both well-defined formalization and a graph-based user-friendly visualization of the problem space.

Section 2 briefly introduces the state of the art on adaptation algorithms. Section 3 provides details of the CAP algorithm and discusses how an ill-designed problem set can results in a sub-optimal learning experience in games using CAP. Section 4 introduces the three instruments for analysis of the problem set. Section 5 provides a practical demonstration of applying the instruments on real problem sets from the online serious gaming platform Math Garden [7]. Section 6 discusses the methodology in comparison with existing practices. Section 7 touches on future research including possible connections to the Knowledge Space Theory [8-9]. Concluding remarks are provided in Section 8.

2. State of the art

Adapting game difficulty to the player's expertise level [10-11] can foster more effective learning. In the Zone of Proximal Development [12] theory, Vygotsky argued that balancing task challenge and learner's expertise promotes more optimized and paced learning. Flow theory [13] argues that such balance maintains learner's motivation and helps to focus more cognitive resources on the task. The 4C/ID [14] instructional model also argues for scaffolded learning of complex skills via tasks gradually increasing in difficulty.

A well-validated method for assessment and adaptation is Knowledge Space Theory (KST, [8-9]). KST expresses a player's knowledge state as a set of problems the player can solve at any time during learning. All feasible knowledge states form a knowledge structure where edges represent prerequisite dependencies among knowledge states. Knowledge spaces, more constrained forms of knowledge structures, can be used for assessment and for personalizing learning paths [3]. Building knowledge spaces requires identification of prerequisite dependencies among the problems. This step often requires a consultation with domain experts [15-17] or complex analysis of response patterns [18-20]. This complexity is a major barrier to KST's wider adoption.

Alternatively, the Computerized Adaptive Practice (CAP, [2]) algorithm was specifically designed for serious games to automatically and in real time adapt game difficulty to player's expertise. CAP is an extensively modified variation of the Elo rating system [21] that was originally developed to assess and match chess players. While many variations of the Elo system exist to match human players in competitive online games [22], CAP was designed to match a human player against the game rather than another human player. The effectiveness of CAP was extensively validated [2,23-25] with an online platform Math Garden [7] that offers serious games as subscription services. Currently, Math Garden is used by more than 1500 Dutch schools offering an opportunity to test CAP using big data.

CAP's adaptation is simpler to use than KST. CAP adopts a psychometric approach to estimate problems' difficulties given players' responses. The process is automatic and produces quantitative and intuitive measures of difficulty that are easy to understand for teachers and developers. However, as discussed in section 3, a poorly designed problem set can diminish learning efficiency. The issue can be mitigated if the problem set is refined over time, but CAP lacks a formal method for analyzing and describing the problem set.

3. Computer Adaptive Practice system

3.1 Adaptation and assessment algorithms in CAP

The section offers a short overview of the Computerized Adaptive Practice (CAP) algorithm. A more detailed discussion is available in [2,26]. CAP assumes that the game contains a set of playable single-player problems. Each problem has a difficulty rating β that increases with problem's difficulty. Similarly, each player has an expertise rating θ that increases with the player's expertise. Given above notions, CAP provides two main functions.

First, given a player m with the expertise rating θ_m, CAP selects a problem with the difficulty β that the player m can solve with a target probability of success P_t. The process involves three steps: (a) defining P_t, (b) estimating a target difficulty rating β_t, and (c) selecting a problem that closely matches the β_t. Equation 1 is used for estimating the β_t. In Math Garden, P_t is drawn from a normal distribution $N(P = 0.75, SD = 0.1)$ and restricted to interval $0.5 < P_t < 1.0$ so that the player can maintain an average success rate of 75%. According to [2,27], this success rate provides a reasonable balance between keeping a player motivated and a maintaining measurement precision. The problem i is selected if it has the difficulty rating β_i closest to the β_t. Overall, the CAP system administers easier problems with lower ratings to novice players and gradually introduces more difficult problems with higher ratings as the player gains more expertise.

$$\beta_t = \theta_m + ln\frac{1-P_t}{P_t}; \quad \min|\beta_i - \beta_t| \qquad \text{(Eq. 1)}$$

$$S_{im} = (2x_{im} - 1)\left(1 - \frac{t_{im}}{d_i}\right); \qquad E(S_{im}) = \frac{e^{2(\theta_m-\beta_i)}+1}{e^{2(\theta_m-\beta_i)}-1} - \frac{1}{\theta_m-\beta_i} \qquad \text{(Eq. 2)}$$

$$\tilde{\theta}_m = \theta_m + K_m\big(S_{im} - E(S_{im})\big); \qquad \tilde{\beta}_i = \beta_i + K_i\big(E(S_{im}) - S_{im}\big) \qquad \text{(Eq. 3)}$$

Second, β_i and θ_m are re-assessed based on the player m's performance in the problem i. Performance is defined by the accuracy x_{im} and the response time t_{im}. x_{im} is either zero or one indicating whether the player was able to solve the problem, and t_{im} is a duration of time spent on the problem. These measures are translated into the observed score S_{im} (Equation 2) using the High Speed-High Stakes scoring rule [23] that accounts for the speed-accuracy trade-off. $E(S_{im})$ is the expected score calculated from θ_m and β_i. The term d_i is a time limit for the problem i. The discrepancy between $E(S_{im})$ and S_{im} is used to re-assess β_i and θ_m using a modified Elo update function (Equation 3). The terms K_m and K_i are factors reflecting uncertainties in measurements of expertise and difficulty ratings [28]. Overall, if the player performs well then θ_m increases and β_i decreases. Vice versa is true if the player demonstrates insufficient performance. Accuracy of θ_m estimation increases if the player plays more problems, and accuracy of β_i estimation increases if the problem is played many times.

3.2 Learning experience and the problem set in CAP

Ideally, a problem set should be designed to optimize player's learning experience. In CAP, a new player is assigned a low starting rating θ (for example a rating equal to the lowest difficulty rating in the problem set, $\theta = \min(\beta)$). θ is continuously updated as the player plays more problems. Changes to θ over time reflect player's learning curve. Ideally, the problem set should facilitate an optimal learning curve with a continuous upward trend until learning ceiling of the domain is reached (dashed curve in Figure 1a). Undesirable trends in a suboptimal learning curve (solid curve in Figure 1a), such as downward trends and plateaus, should be avoided. These trends not only slow learning progress but also can be demotivating to players.

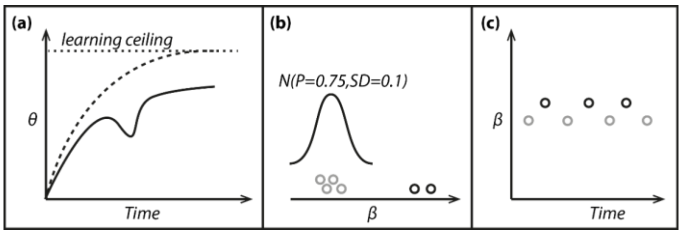

Figure 1. Optimal (dashed curve) and suboptimal (solid curve) learning curves.

One of the causes of the suboptimal curve can be an inappropriate design of the problem set. A player may not be able to transition from easier problems (grey ovals in Figure 1b) to more difficult problems (black ovals in Figure 1b) due to high distance in ratings between the two groups of problems. This problem can be avoided by adding intermediate problems to close the rating gap. Ideally, the problems should be spread uniformly along the difficulty scale to ensure smoother learning experience. Figure 1c shows another pattern of administered problems that can cause a plateau in the learning curve. The black ovals are problems the player is unable to solve, while the grey ovals are problems mastered by the player. This situation can occur if the problem set does not represent a homogeneous learning domain, or not all skills within the domain are represented in the problem set. In both cases, easier problems may fail to train skills necessary for more difficult problems resulting in a learning plateau.

To avoid these problems, a careful design and follow-up refinement of the problem set is required. For this purpose, we propose various instruments for analyzing the problem set. One of the important criteria for these instruments is being practical and accessible to teachers and game developers who do not necessarily have the technical knowledge to perform complex analytics. Therefore, an emphasis is put on visualizing the problem set and the possible relations between problems. Additionally, the proposed instruments analyze the problem set at different levels of complexity offering a degree of customization during a practical use. The instruments are based on the Guttman scale [4], a ranked order, and a Hasse diagram [3,5-6].

4. Methods for analyzing the problem set

4.1 Guttman scale

Items with difficulty ratings and belonging to the same domain can be linearly ordered along the Guttman scale [4,29]. Next, a quasi-order $\succeq$ can be defined among the items to establish dependency. In our case, the problems can be ordered on the Guttman scale by the difficulty ratings. Ideally, given two problems i and j with ratings $\beta_i < \beta_j$, any player who is able to solve j should be able to solve i: $j \succeq i$. For a hypothetical domain Q' with five problems $Q' = \{a, b, c, d, e\}$ where $\beta_a < \beta_b < \beta_c < \beta_e < \beta_d$, the problems can be positioned on the Guttman scale as shown in Figure 2a. The construct is convenient for checking how uniformly the problems are distributed along the difficulty scale. For example, there is a big gap between problems c and e that may slow the learning progress. Also, the small distance between problems e and d may indicate that problems are duplicates in terms of learning content. The issue can be resolved by replacing e with an easier problem closer to c, or by replacing d with a more difficult problem and inserting an additional problem between e and c.

Figure 2a also indicates a possible learning path the player may take. The learning path can be more conveniently visualized as ordered subsets of problems that the player can solve (Figure 2b) at any given time. For example, one player may be able to solve subset $\{a, b, c\}$ while another player may be able to solve subset $\{a, b, c, e\}$ but not the entire problem set Q'. The quasi-order of subsets shows the order in which the problems are learnt.

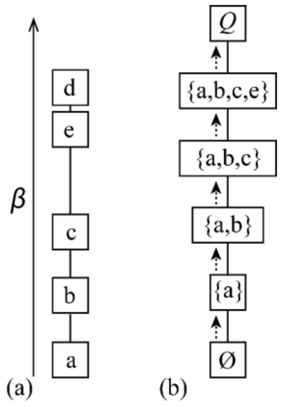

Figure 2. (a) Problems from a hypothetical domain Q' $= \{a, b, c, d, e\}$ are placed on the Guttman scale in an increasing order of difficulty ratings; (b) A quasi-ordinal structure implied by Q' ordered on the Guttman scale.

While being simple and easy to understand, limitations of the Guttman scale hide important dynamics of CAP. There is an unrealistic assumption that every problem on the Guttman scale is played at least once and strictly in the order defined by the scale. In reality, some problems may be skipped because they are too similar in difficulty to problems that the player has already played. The stochastic nature of CAP's problem selection algorithm (Equation 1) allows two players m and n of the same skill rating ($\theta_m = \theta_n$) to have different histories of solved problems. If there are two problems i and j with the same difficulty ratings ($\beta_i = \beta_j$) then it is possible that the player m was administered the problem i and the player n was administered the problem j. Thus, two players of the same skill rating may have diverging subsets of problems they can solve. In other words, there can be multiple learning paths in CAP that are defined by distributions of problems along the difficulty scale. This issue can be addressed by using a difficulty rank scale rather than the rating scale.

4.2 Ranked order

We assume that two or more problems can have the same difficulty rank. In CAP, we can define two problems i and j ($i, j \in Q$) as having the same ranks ($R_i = R_j$) if their difficulty ratings are equal ($\beta_i = \beta_j$). Figure 3a shows problems from a hypothetical domain $Q' = \{a, b, c, d, e, f, g\}$ ordered according to the difficulty ranks. The problems are assumed to have following difficulty ratings: $\beta_a < \beta_b = \beta_c < \beta_d < \beta_e = \beta_f < \beta_g$. Based on the ratings, the problems b and c occupy the same rank as well as the problems e and f: $\beta_b = \beta_c \rightarrow R_b = R_c$ and $\beta_e = \beta_f \rightarrow R_e = R_f$.

The ranked order is more expressive than the rating order. For example, the order in Figure 3a shows that a player who can solve the problem a may be administered either the problem b or c. Similarly, a player who can solve the problem d will be administered either the problem e or f. These alternatives offer four distinct learning paths depicted in Figure 3b. Similar to Figure 2a, the tree structure shows subsets of problems the player was able to solve along the learning path together with the order in which the problems were solved. The graph also provides means to compare different players. For example, two players may have taken different learning paths despite both being able to solve the problem g. This difference could have been reflected in the learning curves since difficulty can be defined by multiple factors as discussed in section 5. Because of the multi-faceted nature of difficulty, the assumption of a quasi-order is not present in the rank order. For example in Figure 3a, we cannot assume that the player can solve the problem b if the player can solve the problems $\{a,c,d\}$. Therefore, Figure 3b represents learning paths strictly based on the psychometric difficulty estimations and ignoring latent dependencies among problems.

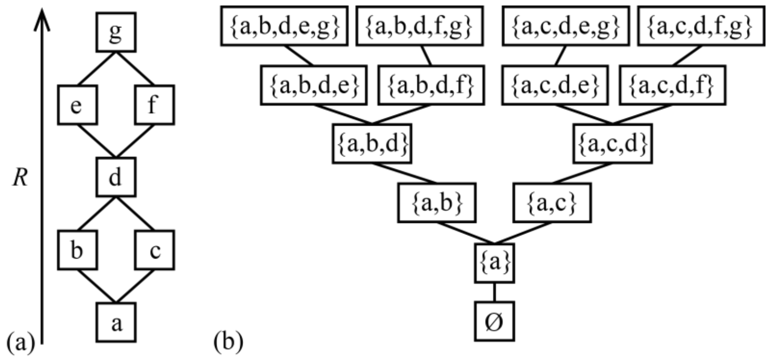

Figure 3. (a) Problems from a hypothetical domain $Q' = \{a, b, c, d, e, f, g\}$ are positioned in an increasing order of difficulty ranks. (b) Four possible learning paths implied by the ranked order. Each node along the path is a player's possible learning state.

In practice, two problems are unlikely to have equal difficulty ratings even if they are exactly the same. Therefore, a fuzzy approach for evaluating the similarity of two difficulty ratings is required. First, we can assume a threshold Th and a function $Diff$ that calculates a degree of difference between two difficulty ratings. The problems i and j are considered to have difficulty ratings that are not significantly different if a degree of difference is below or equal to a threshold: $\beta_i \cong \beta_j$ if $Th \geq Diff(\beta_i, \beta_j)$. If Th is treated as a free parameter then we only need to define the function $Diff$. For this purpose, we can reuse Equation 1. If player's skill rating is equal to problem's difficulty rating $\beta_i = \theta$, the player is equally likely to succeed or fail ($P_t = 0.5$). We can use this property and substitute the player's skill rating with another problem's difficulty rating β_j (Equation 4). If two problems are equal in difficulty $\beta_i = \beta_j$ then $P_t = 0.5$. Otherwise, the difference in difficulties is reflected in divergence of P_t from 0.5. Therefore, the $Diff$ function can be written as an absolute difference between P_t and 0.5 (Equation 5).

$$\beta_i = \beta_j + ln\frac{1-P_t}{P_t}; \qquad P_t = \frac{1}{e^{\beta_i - \beta_j}+1} \qquad \text{Eq. (4)}$$

$$Diff(\beta_i, \beta_j) = |0.5 - P_t|; \qquad Diff(\beta_i, \beta_j) = \left|0.5 - \frac{1}{e^{\beta_i - \beta_j}+1}\right| \qquad \text{Eq. (5)}$$

The output of the function $Diff(\beta_i, \beta_j)$ indicates an increase or a decrease in difficulty. For example, $Diff(\beta_i, \beta_j) = 0.05$ means that the problem i is more difficult or easier than the problem j by 5%. Accordingly, the threshold Th can be viewed as a minimum fractional difference between difficulties of two items. A possible heuristic approach for choosing the value for Th is to limit its value to two standard deviations of the normal distribution from which P_t is drawn: $If\ P_t \sim N(\mu, \sigma)\ then\ Th \leq 2\sigma$.

Note that due to an exponential component in the Equation 4, the value of P is limited to an interval (0, 1). Correspondingly, the value of the function $Diff$ is limited to an interval [0, 0.5). Thus, the maximum estimated difference between two difficulties cannot exceed 50% even if the true difference may be higher. This issue can be safely ignored since setting the threshold above 50% may not be necessary in most cases.

One issue to consider in the proposed method is possible ambiguity into the rank assignment. As an example, we can use $Q' = \{i, j, k\}$ where $\beta_i < \beta_j < \beta_k, Th \geq Diff(\beta_i, \beta_j), Th \geq Diff(\beta_j, \beta_k)$, and $Th < Diff(\beta_i, \beta_k)$. The rank of the problem k is ambiguous. One option is to assign the problem k the same rank as the problems i and j. Alternatively, the problem k can be assigned a higher rank than the problems i and j. In our current solution, the comparison for the same rank is always made with the problem having the lowest difficulty rating. Thus, the problem j is assigned the same rank as the problem i, but the problem k is assigned a higher rank than the problems i and j.

4.3 Expanding the tree structure into the Hasse diagram

Construction of the tree structure in Figure 3b is based on an assumption that the player can play only one problem of the same rank. For example, if the player played the problem b then the problem c is not available anymore. This assumption is unlikely to hold in many cases. In CAP, if the player underperforms then the player's skill rating decreases (Equations 2 and 3). Consecutively, the player may be administered the lower rating problems played before or even previously unplayed problems of similar lower rating. Using Figure 3 as an example, let us assume that the player is located at the node $\{a, b\}$ along the lerning path. If the player is administered the problem d and underperforms then the player's skill rating may decrease to match difficulty ratings of b and c. Therefore, the next problem the player is administered can be either b or c. If c is selected then the player may transition into a node $\{a, b, c\}$ that is not reflected in Figure 3b. Thus, there can be more learning paths than the ones depicted in Figure 3b. The missing learning paths can be inferred from the ranked order by applying two following rules:

Rule 1: Given a set G_R of problems with a rank R and a set $G_{<R}$ of problems of lower ranks ($G_R, G_{<R} \subseteq Q$), a union of any subset of $G_R$with $G_{<R}$ is a node in a learning path.

Rule 2: Given a set G_R of problems with a rank R and a set G_{R-1} of problems with rank $R - 1$, a union of any subset of G_R with any node K_{R-1} containing at least one problem from G_{R-1} is also a node in a learning path.

In other words, *Rule 1* states that in order to be able to solve any problem with some rank R, the player should have solved or, at least, should be able to solve all lower ranking problems $<R$. The gray-shaded nodes in Figure 4 are produced by applying *Rule 1* to the ranked order in Figure 3a. For example, a set $\{b, c\}$ of rank 2 problems has following subsets $\{b\}$, $\{c\}$, and $\{b, c\}$. At rank 1, a set $\{a\}$ has only itself as a subset (via union with an empty set $\emptyset$). Unions of every subset at rank 2 with the subset at rank 1 result in following sets: $\{a, b\}$, $\{a, c\}$, and $\{a, b, c\}$. Applying *Rule 1* to every ranked set of problems and adding an empty set $\emptyset$ result in the gray nodes in Figure 4.

Rule 1 essentially produces a branched, therefore more general, version of the linear learning path produced from the Guttman scale (Figure 2). Similarly, there is an assumption that every problem on the ranked order is played at least once and strictly in the order defined by the scale. This assumption allows putting problems of the same rank in the same nodes thereby producing learning paths missing in Figure 3b. Interestingly, the gray-shaded nodes form a Hasse diagram [3,5-6] of a finite partially ordered set. Importance of this property is discussed later in section 5 in connection with the Knowledge Space Theory [8,9].

Rule 2 is applied to the Hasse diagram produced by *Rule 1*. As an example, consider the sets $\{d\}$ and $\{b, c\}$ in Figure 3a. There is only one possible subset of $\{d\}$ which is itself. In the Hasse diagram composed of gray blocks, there are three nodes that contain problems from $\{b, c\}$, but do not contain the set $\{d\}$: $\{a, b\}$, $\{a, c\}$, and $\{a, b, c\}$. Unions of these nodes with the set $\{d\}$ produces following nodes: $\{a, b, d\}$, $\{a, c, d\}$, and $\{a, b, c, d\}$. In this manner, *Rule 2* can be recursively applied on the Hasse diagram until no new node is added. The new nodes produced by *Rule 2* have white shading in Figure 4. More intuitively, *Rule 2* merges the Hasse diagram with the tree diagram from Figure 3b. In the process of merging, a new Hasse diagram is generated with additional possible nodes and edges. In this particular example, six new nodes (marked with the dashed border in Figure 4) are identified that were not present in both the Hasse and tree diagrams. These six new nodes and the Hasse diagram produced by *Rule 1* account for the possibility that the player may be downgraded to lower rating problems. Overall, the diagram in Figure 4 shows possible learning paths assuming that (1) not all problems of the same rank may be administered to the player and (2) the player may return to the problems of the immediate lower rank due to underperformance in the problems of the higher rank.

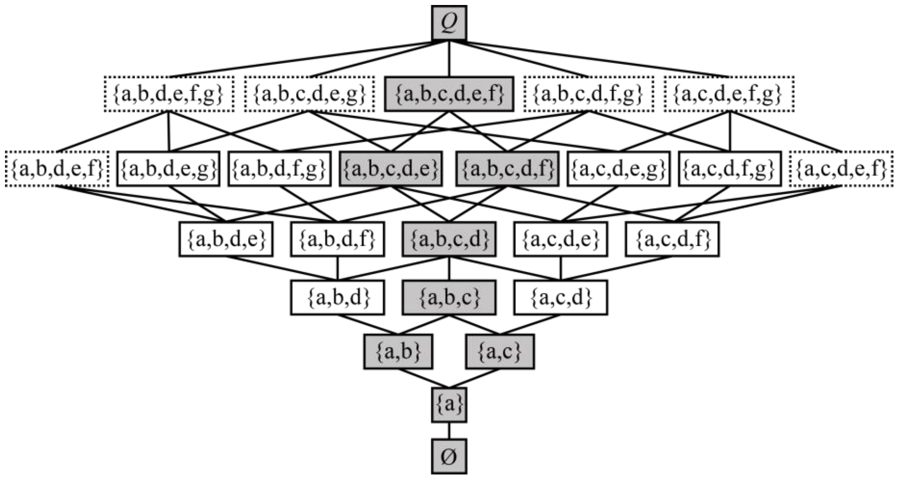

Figure 4. An expanded Hasse diagram resulting from applications of both Rule 1 and Rule 2.

The Hasse diagram can be used for more complex analysis of how different learning paths may result in different learning experiences. For example, the learning paths in Figure 3b can be considered optimal because it will result in a continuously increasing skill rating until the ceiling is reached. On the other hand, additional learning paths in the expanded Hasse diagram can be considered suboptimal because they assume one or more downward trends in the player's skill rating. If many players are in suboptimal learning paths then the problem set may not be designed well and need refinement.

5. A case study based on empirical data

5.1 Exploring potential issues and properties of a problem set

In this section, we demonstrate how a ranked order and a Hasse diagram can be used for exploring the problem set in a serious game from Math Garden [7]. As an example, we used the Number game for training basic arithmetic skills as well as general problem-solving skills.

Given a set S_N of numbers and a set S_O of arithmetic operators, a player has to make a target number T. Operators in S_O can be reused, but each number in S_N has to be used only once. Size of S_N ($|S_N|$) can range from 2 to 5, and S_O can be an any combination of following operators: $+$, $-$, $\times$, $/$, and exponent ^. An example is shown in Figure 5. The player is required to reach the target number 2 using only addition and/or subtraction involving three other numbers 1, 5, and 6. Possible solutions are 6-5+1, 6+1-5, and 6-(5-1). Some problems can be extremely challenging from a computational perspective. For example, the subset of problems where $S_O = (+, -)$ are NP-complete meaning that even computers have difficulty solving these problems by exhaustive search over the solution space [30-31]. However, human players are surprisingly good at solving such problems [32]. This success is predicated on an ability to discover heuristic strategies for efficiently exploring a vast solution space. Later, we will provide an example of such strategy.

From the player’s perspective, there are multiple factors that may define problem difficulty. The obvious two factors are inherent difficulties of operators (e.g., addition is easier than multiplication) and numbers (e.g., 10×10 is easier than 7×8). The less obvious factor is player’s preference for one operator over another (e.g., trying a solution with addition before trying a solution with multiplication). A similar preference may exist for numbers. Finally, these preferences may vary depending on the problem format. Overall, the Number game promotes not only development of player's arithmetic skills but also complex problem-solving skills necessary to find a correct solution in time.

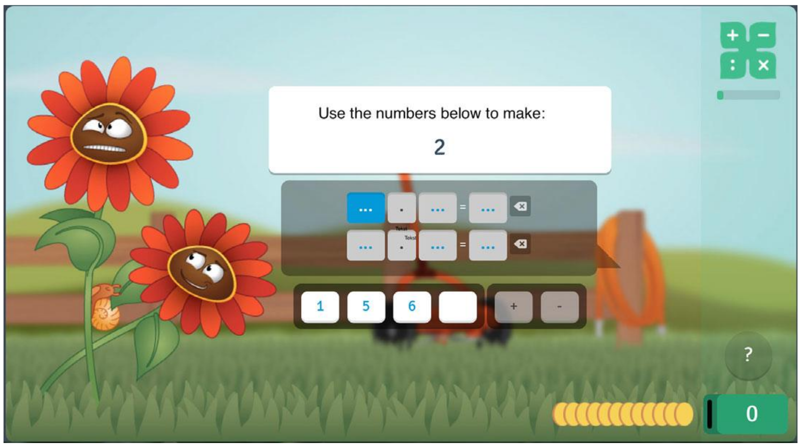

Figure 5. A screenshot from Math Garden showing an instance of the Number game.

All data were extracted from Math Garden in January 2015. In this example, we will limit our scope to a subset of problems where the size of S_N is two ($|S_N| = 2$). To ensure that difficulty ratings are stable, we used problems that were played at least 30 times. For the purpose of simplicity, we also removed problems where S_N contained at least one negative number (e.g., $S_N = (1, -2)$). In total, there were 230 problems classified into seven categories summarized in Table 1. Each category included problems with the same solution pattern. For example, the category $N2 + N1$ consists of 22 problems where a solution is an addition of two numbers. An average difficulty rating of problems in this category is 3.39 with a standard deviation of 1.00. Given a set $S_N = (N1, N2)$, $N2$ represents the bigger number of two ($N2 \geq N1$). The *ID* column contains alphabetic letters unique to each category. These letters are used to represent each category in the ranked order and the Hasse diagram.

Table 1. Categories of problems.

Category	Number of problems	Rating mean	Rating SD	ID
N2 + *N1*	22	3.39	1.00	a
N2 × *N1*	54	24.18	5.07	b
N2 - *N1*	71	24.31	0.86	c
N2 / *N1*	44	32.19	0.44	d
N1 / *N2*	9	35.61	0.21	e
N2 ^ *N1*	25	37.05	1.53	f
N1 ^ *N2*	5	45.17	0.28	g

Figure 6a shows the ranked order and the expanded Hasse diagram produced with threshold $Th = 0.1$. This means that the problems of the same rank do not differ in difficulty for more than 10%, and problems of an immediate higher rank are difficult for more than 10% than problems of the rank below. Problems with addition are the easiest as can be expected. After addition problems, the player may progress to either multiplication or subtraction problems that have similar difficulty ratings. Next, the division of a bigger number by a smaller number is easier than the division of the smaller number by the bigger number. Based on this pattern, we can safely assume that the player needs to learn division without remainder before learning division with remainder. Exponentiation is the most difficult operation. Similar to division, exponentiation problems can be divided into two groups based on the size of the exponents. Figure 6b shows the extended Hasse diagram with learning paths inferred from the ranked order in Figure 6a. The shaded nodes are generated with *Rule 1*. Remaining nodes are derived by applying *Rule 2*. There are ten potential learning paths formed by 16 learning states the player may occupy.

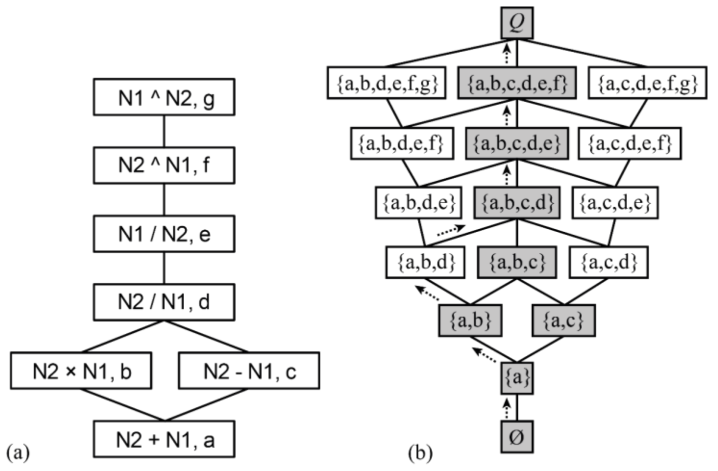

Figure 6. (a) A ranked order ($Th = 0.1$) of seven categories of problems from the Number game. (b) A Hasse diagram of possible learning paths.

This analysis reveals multiple issues and features in the problem set. First, Table 1 reveals three big difficulty rating gaps after $N2 + N1$, $N2 - N1$, and $N2^\wedge N1$. These gaps may present players with sudden and undesirable spikes in problem difficulty. The potential negative impact can be explored using the Hasse diagram in Figure 6b. For example, transitions of players from the node $\{a\}$ to either nodes $\{a, b\}$ or $\{a, c\}$ maybe be slow. For a smoother learning experience, the intermediate problem can be added in these gaps. For example, problems with additions of large numbers or additions of more than two numbers can be added to smoothen the transition from the addition problems to the multiplication problems. Next, problems for multiplication and subtraction are competing for selection. This is rather undesirable since the player should train both multiplication and subtraction skills in equal measure. One solution is too artificially increase the difficulty of either multiplication or subtraction problems so that a linear order is formed. Alternatively, both operators can be included in the same set of problems. In the latter solution, a potential effect of interaction on the difficulty rating should be also considered. Third, the ranked order clearly reveals that the division and exponentiation problems should be categorized based on both operators and operands. A teacher or game developer should reflect these two factors defining difficulties of arithmetic problems in the design of the problem set. Finally, the ranked order rather nicely reflects commonly perceived difficulty of arithmetic operations.

5.2 Exploring a problem set to analyze players' learning strategies

In this example, a ranked order of problems is used to explore potential learning strategies players may use to solve more complex problems in the Number game. We investigate problems where players need to exhibit arithmetic prowess as well as deploy a sophisticated strategy to find a solution. A previous study [32] proposed that players are employing a forward reasoning strategy. The strategy involves only numbers in the set S_N and not the target number T. Since players can operate on only two numbers at the time, there is a need for a selection criterion for numbers to be used with the first operation. It was discovered that larger numbers are preferred to smaller ones. For example, if a solution for a problem involves the addition of three numbers (e.g., $S_N = \{2,5,10\}$, $T = 17$) then the most likely solution is a summation of the two biggest numbers and then the addition of the smallest number: (e.g., (10 + 5) + 2). In the previous study, we referred to it as a greedy selection criterion. We expect the ranked order to reflect the forward reasoning strategy. Problems that are most compatible with the forward reasoning strategy should be ranked lower than the problems that are less compatible with the strategy. Additionally, we expect an interaction between the difficulty of arithmetic operations and compatibility with the forward reasoning strategy. The nature of such interaction is difficult to predict. However, we expect the ranked order to provide some insight.

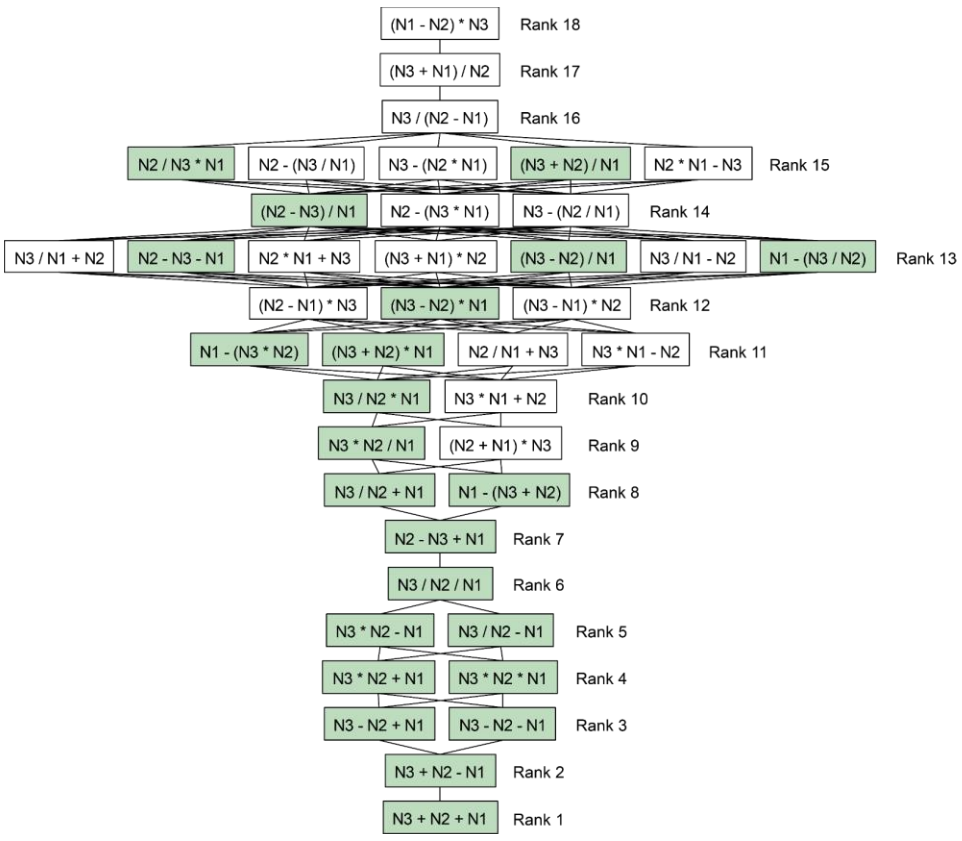

Figure 7. A ranked order ($Th = 0.1$) of 41 categories of problems with $|S_N| = 3$. Numbers on the right indicate rank. Green boxes are problems that have solutions compatible with the forward reasoning strategy and the greedy selection criterion.

All data were extracted from Math Garden in January 2015. We again use a subset of problems where the size of S_N is three ($|S_N| = 3$) and S_O contains four operators $+$, $-$, $\times$, and $/$. We used problems that were played 30 or more times. For simplicity, we removed problems where S_N contained at least one negative number, or where a target number had a fraction. Numbers in a set S_N are labeled as $S_N = \{N1, N2, N3\}$ where $N3 \geq N2 \geq N1$. As in the previous example, the problems with the same solution pattern were assigned into the same category. A total of 509 problems formed 41 categories. The ranked order was created based on these categories. In the remainder of this section, we refer to categories as problems. The threshold Th was set to 0.1.
The resulting ranked order is shown in Figure 7. Interesting details can be observed by focusing on particular parts of the graph. Problems that have solutions compatible with the forward reasoning strategy and the greedy selection criterion are shaded in green. As we expected, the rank-order of problems reflects their compatibilities with the selection criterion. Lower ranks contain problems that are compatible with the greedy criterion while higher ranks mostly contain problems that are not compatible with the criterion.

The ranked order also reflects difficulties of operators. Based on the first six lower ranks, we can deduce addition, subtraction, multiplication, and division in an increasing order of difficulty. This order largely matches the one observed in Figure 6a. We can further investigate how various combinations of different operations affect the difficulty of problems. For example, an interesting asymmetry is observed between problems in the second and third ranks. The problem $N3 + N2 - N1$ is a prerequisite for problems $N3 - N2 - N1$ and $N3 - N2 + N1$. It makes sense that the problem with two subtractions is more difficult than the problem with one subtraction. However, the problem $N3 - N2 + N1$ has the same rank as the problem $N3 - N2 - N1$ despite having the same combination of operations as in $N3 + N2 - N1$. This asymmetry suggests that the order of operations also defines difficulty of problems and, therefore, the structure of the graph. Players not only find addition to be simpler than subtraction but also may prefer to choose addition over subtraction as a possible solution for a problem.

6. Discussion

This work proposes multiple instruments for analyzing a problem set in an adaptive serious game. The Guttman scale [4] can be used to analyze the distribution of problems along the linear scale of difficulty rating. The ranked order introduces a notion of similarity to identify problems that may compete for selection. More importantly, the ranked order enables inference of possible learning paths depicted in a Hasse diagram [3,5-6]. The Hasse diagram provides a clear overview of the learning paths as well as learning states that the player may progress through. These three instruments offer analysis of the problem set at various levels of complexity and granularity. Yet, graph-based nature of these instruments allows visualization that is intuitive and easy to read for teachers and game developers.

As discussed earlier, Knowledge Space Theory [8-9] is an alternative approach to assessment, adaptation, and exploring the problem space. Knowledge structures in KST can also be visualized as Hasse diagrams reflecting dependencies among problems. In Figure 4, the shaded nodes can be seen as a knowledge structure if we assume that nodes are knowledge states and edges represent prerequisite dependencies. On the one hand, KST is a well-established and validated approach. On the other hand, knowledge structures need to be built before its use in serious games [33]. This is an effortful process requiring either domain experts [15-17] or complex analysis of response patterns [18-20]. In contrast, our methodology is automatic, computationally inexpensive, and can construct graph representations in real-time based on up-to-date data. While accuracies of the three graph representations depend on the accuracies of the difficulty ratings, CAP can quickly converge on reliable difficulty ratings given a sufficient number of observations [2].

Furthermore, our methodology promotes explorative analysis of the problems space. This is in contrast to KST where the knowledge structure often remains static once it is built. However, the knowledge structure may not accurately reflect the learning domain. To address this issue, [33] propose a method to automatically refine an existing knowledge structure. However, the method does not address issues caused by an inaccurate or incomplete problem set. With our methodology, teachers and developers can continue to analyze and improve the problem set even after the game's deployment.

7. Future works

A method for inferring individual player's learning state in a Hasse diagram can be useful for multiple reasons. Recent surveys [34-35] indicate that serious games are often used by teachers for a formative assessment to identify a need for an intervention. The Hasse diagram can be invaluable in identifying player's current progress (at what node the student is), potential barriers to learning (what was player's performance in current and preceding nodes of the learning path?), and type of intervention necessary (what are the learning requirements for the higher-level nodes?). Additionally, the analysis of learning states can be used for identifying the most likely learning paths in the problem set. The analysis of prevalent learning paths can be useful for further optimization of the problem set and identification of outlier players who deviate from these paths. Unfortunately, existing methods for inferring player's learning state [36-38] cannot be reused directly. First, CAP assesses the player at three different levels. The challenge is to choose among these three types of measures the most suitable one (or a combination of) for inferring player's learning state. Second, the fact that CAP modulates the order in which the problems are administered needs to be taken into account.

The nature of relations among problems needs to be explored more. In this study, we relate problems by difficulty only. However, multiple factors may contribute to problem difficulty. One factor is a cognitive demand necessary to complete the problem. Another factor, we are particularly interested in, is prerequisite dependencies defined by skills necessary for solving the problems. Given two problems i and j from a domain Q ($i, j \in Q$), i is a prerequisite for j ($j \succeq i$) if we can infer the player's success in i from the player's success in j. In other words, solving j requires the skills (and more) required for solving i. Knowledge Space Theory [8-9] offers multiple methods for response pattern analysis to infer such prerequisite dependencies [18-20]. For example, [39] introduced the k-state procedure, a data-driven approach for building knowledge structures based on the k-modes clustering used in the area of data-mining. This clustering approach can be potentially used on big data offered by the Math Garden to explore prerequisite dependencies.

To apply the methods from KST to our approach, the learning states and Hasse diagrams should satisfy constraints for valid knowledge states, structures, and spaces [9]. For example, transforming a knowledge structure into a knowledge space requires satisfaction of constraints that apply to individual states as well as the overall topography of the structure. While being challenging, such alignment with KST can bring other benefits. It can widen our methodology's relevance and applicability outside of CAP-enabled systems. Furthermore, it will contribute to ongoing lines of research on the automatic construction of knowledge structures [33] and on integrating latent trait theory [19] into KST.

8. Conclusion

In this work, we proposed a methodology and three corresponding instruments involving the Guttman scale, a ranked order, and a Hasse diagram to explore problem set in adaptive serious games integrating CAP system. The proposed methodology is first to use a difficulty measure to explore a problem space and well-suited for serious games that emphasize both knowledge acquisition and consolidation via practice. Different instruments enable analyses of the problem set at different levels of complexity. The methodology can be used throughout the game's lifetime either to improve the problem set or to gain insight into players' learning tendencies such as learning paths or solution strategies. The effectiveness of the method was demonstrated based on two use cases involving real data from an online serious gaming platform Math Garden. The methodology is accessible to teachers and game developers without extensive technical knowledge.
TwoA [26] is a portable library implementing the CAP algorithm for offline use in a smaller and isolated classroom environment. It can be integrated with popular game development platforms, such as Unity3D, to create adaptive games. The motivation is to offer more control to teachers and game developers in designing serious games that meet specific curriculum requirements while still leveraging from difficulty adaptation offered by CAP. TwoA also offers API for generating a ranked order and a Hasse diagram from a set of problems. The generated structures can be used in-game or stored in external files in an XML format. TwoA is available at https://github.com/rageappliedgame/HatAsset under an open-source license.

Acknowledgment

The data from Math Garden was gathered under the support of a grant from the Netherlands Organization for Scientific Research (NWO). The methodology research has been partially funded by the EC H2020 project RAGE (Realising an Applied Gaming Eco-System); http://www.rageproject.eu/; Grant agreement No 644187.

References

[1] Ratan, R., Ritterfeld, U., "Classifying serious games", In Serious games: Mechanisms and effects, pp. 10-24, 2009.
[2] Klinkenberg, S., Straatemeier, M., Van der Maas, H. L. J., "Computer adaptive practice of maths ability using a new item response model for on the fly ability and difficulty estimation", Computers & Education, Vol. 57, Nr. 2, pp. 1813-1824, 2011. https://doi.org/10.1016/j.compedu.2011.02.003
[3] Heller, J., Steiner, C., Hockemeyer, C., Albert, D., "Competence-based knowledge structures for personalized learning", International Journal on E-Learning, Vol. 5, Nr. 1, pp. 75-88, 2006.
[4] Guttman, L., "The quantification of a class of attributes: A theory and method of scale construction", In The Prediction of Personal Adjustment, pp. 319-348, 1941.
[5] Barnes, T., "The q-matrix method: Mining student response data for knowledge", In American Association for Artificial Intelligence 2005 Educational Data Mining Workshop, pp. 1-8, 2005.
[6] Kuznetsov, S. O., "Learning of simple conceptual graphs from positive and negative examples", In European Conference on Principles of Data Mining and Knowledge Discovery, pp. 384-391, 1999. https://doi.org/10.1007/978-3-540-48247-5_47
[7] "Rekentuin", https://www.rekentuin.nl/, Retrieved on February 21, 2018.

[8] Falmagne, J. C., Koppen, M., Villano, M., Doignon, J. P., Johannesen, L., “Introduction to knowledge spaces: How to build, test, and search them”, Psychological Review, Vol. 97, Nr. 2, pp. 201, 1990. https://doi.org/10.1037/0033-295X.97.2.201

[9] Falmagne, J.C., Doignon, J.P., “Learning spaces. Interdisciplinary applied mathematics”, Berlin: Springer, 2011.

[10] Lopes, R., Bidarra, R., "Adaptivity challenges in games and simulations: a survey", IEEE Transactions on Computational Intelligence and AI in Games, Vol. 3, Nr. 2, pp. 85-99, 2011. https://doi.org/10.1109/TCIAIG.2011.215284

[11] Peirce, N., Conlan, O., Wade, V., "Adaptive educational games: Providing non-invasive personalised learning experiences", In Second IEEE International Conference on Digital Games and Intelligent Toys Enhanced Education, IEEE, pp. 28-35, 2008. https://doi.org/10.1109/DIGITEL.2008.30

[12] Vygotsky, L. S., "Interaction between learning and development", In Mind and society, Cambridge MA: Harvard University Press, pp. 79-91, 1978.

[13] Nakamura, J., Csikszentmihalyi, M., "The concept of flow", In Flow and the foundations of positive psychology, Springer Netherlands, pp. 239-263, 2014 https://doi.org/10.1007/978-94-017-9088-8_16

[14] Van Merriënboer, J. J., Kester, L., "The four-component instructional design model: Multimedia principles in environments for complex learning", In The Cambridge handbook of multimedia learning, Cambridge University Press, pp. 71-93, 2005. https://doi.org/10.1017/CBO9780511816819.006

[15] Koppen, M., "Extracting human expertise for constructing knowledge spaces: An algorithm", Journal of Mathematical Psychology, Vol. 37, Nr. 1, pp. 1-20, 1993. https://doi.org/10.1006/jmps.1993.1001

[16] Kambouri, M., Koppen, M., Villano, M., Falmagne, J. C., "Knowledge assessment: Tapping human expertise by the QUERY routine", International Journal of Human-Computer Studies, Vol. 40, Nr. 1, pp. 119-151, 1994. https://doi.org/10.1006/ijhc.1994.1006

[17] Barnes, T., Bitzer, D., Vouk, M., "Experimental analysis of the q-matrix method in knowledge discovery", In International Symposium on Methodologies for Intelligent Systems, Springer Berlin Heidelberg, pp. 603-611, 2005. https://doi.org/10.1007/11425274_62

[18] Kelly, A. E., Birenbaum, M., "Diagnosing knowledge states in algebra using the rule-space model", Journal for Research in Mathematics Education, Vol. 24, Nr. 5, pp. 442-459, 1993.

[19] Birenbaum, M., Tatsuoka, C., Yamada, T., "Diagnostic assessment in TIMSS-R: Between-countries and within-country comparisons of eighth graders' mathematics performance", Studies in Educational Evaluation, Vol. 30, Nr. 2, pp. 151-173, 2004. https://doi.org/10.1016/j.stueduc.2004.06.004

[20] Tatsuoka, K. K., "Rule space: An approach for dealing with misconceptions based on item response theory", Journal of Educational Measurement, Vol. 20, Nr. 4, pp. 345-354, 1983. https://doi.org/10.1111/j.1745-3984.1983.tb00212.x

[21] Elo, A. E., "The rating of chessplayers, past and present", Arco Pub, 1978.

[22] Herbrich, R., Minka, T., Graepel, T., "TrueSkill: a Bayesian skill rating system", In Proceedings of the 19th International Conference on Neural Information Processing Systems, MIT Press, pp. 569-576, 2006.

[23] Maris, G., Van der Maas, H.L.J., "Speed-accuracy response models: Scoring rules based on response time and accuracy", Psychometrika, Vol. 77, Nr. 4, pp. 615-633, 2012. https://doi.org/10.1007/s11336-012-9288-y

[24] Jansen, B. R., Louwerse, J., Straatemeier, M., Van der Ven, S. H., Klinkenberg, S., Van der Maas, H. L., "The influence of experiencing success in math on math anxiety, perceived math competence, and math performance", Learning and Individual Differences, Vol. 24, pp. 190-197, 2013. https://doi.org/10.1016/j.lindif.2012.12.014

[25] Gierasimczuk, N., Van der Maas, H.L.J., Raijmakers, M. E., "Logical and psychological analysis of deductive mastermind", In ESSLLI Logic & Cognition Workshop, pp. 1-13, 2012.

[26] Nyamsuren, E., Van der Vegt, W., Westera, W., "Automated Adaptation and Assessment in Serious Games: a Portable Tool for Supporting Learning", In Advances in Computer Games, Springer, Cham, pp. 201-212, 2017. https://doi.org/10.1007/978-3-319-71649-7_17

[27] Eggen, T. J., Verschoor, A.J., "Optimal testing with easy or difficult items in computerized adaptive testing", Applied Psychological Measurement, Vol. 30, Nr. 5, pp. 379-393, 2006. https://doi.org/10.1177/0146621606288890

[28] Glickman, M. E., "A comprehensive guide to chess ratings", American Chess Journal, Vol. 3, pp. 59-102, 1995.

[29] Lukas, J., Albert, D., "Knowledge structures: What they are and how they can be used in cognitive psychology, test theory, and the design of learning environments", In Knowledge spaces: Theories, empirical research, and applications, Psychology Press, pp. 3-12, 1999.

[30] Hayes, B., "The easiest hard problem", American Scientist, Vol. 90, Nr. 2, pp. 113-117, 2002 https://doi.org/10.1511/2002.2.113

[31] Kurzen, L., "Some Ideas for the Cijferstaak", Internal report, University of Amsterdam, 2011.

[32] Van der Maas, H.L.J., Nyamsuren, E., "Cognitive Analysis of Educational Games: The Number Game", Topics in Cognitive Science, Vol. 9, pp. 395–412, 2017. doi:10.1111/tops.12231 https://doi.org/10.1111/tops.12231

[33] Cosyn, E., Thiéry, N., "A practical procedure to build a knowledge structure", Journal of Mathematical Psychology, Vol. 44, Nr. 3, pp. 383-407, 2000. https://doi.org/10.1006/jmps.1998.1252

[34] Fishman, B., Riconscente, M., Snider, R., Tsai, T., Plass, J., "Empowering Educators: Supporting Student Progress in the Classroom with Digital Games (Part 2)", Ann Arbor: University of Michigan, gamesandlearning.umich.edu/agames, 2015.

[35] Fishman, B., Riconscente, M., Snider, R., Tsai, T., Plass, J., "Empowering Educators: Supporting Student Progress in the Classroom with Digital Games", Ann Arbor: University of Michigan, gamesandlearning.umich.edu/agames, 2014.

[36] Albacete, P., Silliman, S., Jordan, P., "A Tool to Assess Fine-grained Knowledge from Correct and Incorrect Answers in Online Multiple-choice Tests: an Application to Student Modeling", In EdMedia: World Conference on Educational Media and Technology, Association for the Advancement of Computing in Education (AACE), pp. 988-996, 2017.

[37] Baker, R. S., Yacef, K., "The state of educational data mining in 2009: A review and future visions", Journal of Educational Data Mining, Vol. 1, Nr. 1, pp. 3-17, 2009.

[38] Villano, M., "Probabilistic student models: Bayesian belief networks and knowledge space theory", In International Conference on Intelligent Tutoring Systems, Springer Berlin/Heidelberg, pp. 491-498, 1992. https://doi.org/10.1007/3-540-55606-0_58

[39] de Chiusole, D., Stefanutti, L., Spoto, A., "A class of k-modes algorithms for extracting knowledge structures from data", Behavior Research Methods, Vol. 49, Nr. 4, pp. 1212-1226, 2017. https://doi.org/10.3758/s13428-016-0780-7