A Temporal Data-Driven Player Model for Dynamic Difficulty Adjustment

Alexander E. Zook and Mark O. Riedl
School of Interactive Computing, College of Computing
Georgia Institute of Technology
{a.zook,riedl}@gatech.edu

Abstract

Many computer games of all genres pit the player against a succession of increasingly difficult challenges such as combat with computer-controlled enemies and puzzles. Part of the fun of computer games is to master the skills necessary to complete the game. Challenge tailoring is the problem of matching the difficulty of skill-based events over the course of a game to a specific player's abilities. We present a tensor factorization approach to predicting player performance in skill-based computer games. Our tensor factorization approach is data-driven and can predict changes in players' skill mastery over time, allowing more accurate tailoring of challenges. We demonstrate the efficacy and scalability of tensor factorization models through an empirical study of human players in a simple role-playing combat game. We further find a significant correlation between these performance ratings and player subjective experiences of difficulty and discuss ways our model can be used to optimize player enjoyment.

Introduction

Many computer games of all genres pit the player against a succession of increasingly difficult challenges: combat with NPC enemies, puzzles, strategic planning and execution, etc. The player is expected to master a set of skills pertaining to the game mechanic over the course of the game. Some believe that this mastery of the game is a fundamental aspect to having fun in computer games (Koster 2005). However, contemporary computer games are being played by increasingly diverse audiences that differ in their skills and interests in games. This increasing variability in ability, speed of mastery, and growing diversity in tastes for game aesthetic and narrative content has prompted a recent growth of interest in automated methods to fit game content to these diverse abilities and interests. These efforts require both modeling the abilities and interests of players as well as adapting existing game content to those differences.

Many games revolve around *skill-based events*, periods of game play, such as combat or puzzles, in which the player must perform a specific skill. We see two main challenges to adapting computer games to fit individual player differences: *challenge tailoring* and *challenge contextualization*. Challenge tailoring (CT) is the problem of matching the difficulty of skill-based events over the course of a game to a specific player's abilities. For example, in an action role-playing game such as *The Legend of Zelda* challenge tailoring may manifest as configuring the number, health, or damage dealt by various enemies at various times throughout the game. CT is similar to *Dynamic Difficulty Adjustment (DDA)*, which only applies to online, real-time changes to game mechanics to balance difficulty. In contrast, CT generalizes DDA to both online and offline optimization of game content and is not limited to adapting game difficulty. Challenge contextualization (CC) is the related problem of constructing game events that set up the conditions for skill events and motivate their occurrence to the player. For example, the challenge of slaying a dragon may be contextualized by the dragon kidnapping a princess. Challenge contextualization includes common AI problems of quest generation, story generation in games, and interactive storytelling.

In this paper, we focus on the challenge tailoring problem in adventure role-playing games. Realizing challenge tailoring requires both a player model and an algorithm to adapt content based on that model. Effective player modeling for the purposes of challenge tailoring requires a data-driven approach that is able to predict player behavior in situations that may have never been observed. Because players are expected to master skills over time when playing a game, the player model must also account for temporal changes in player behavior, rather than assume the player remains fixed. Modeling the temporal dynamics of a player enables an adaptive game to more effectively forecast future player behavior, accommodate those changes, and better direct players toward content they are expected to enjoy. Further, forecasting enables player models to account for interrelations among sequences of experiences—accounting for how foreshadowing may set up a better future revelation or how encountering one set of challenges builds player abilities to overcome related challenges that build off of those.

In this paper we present and evaluate a temporal player modeling approach. We employ tensor factorization techniques to create temporal models of objective player game performance over time in a turn-based role-playing game and demonstrate the efficacy of our approach over related data-driven methods through comparisons from an empirical study. We model performance instead of difficulty because performance is objectively measurable while difficulty is subjective; we show difficulty and performance are significantly correlated for this particular domain. Finally, we suggest how our tensor factorization player model may be used for challenge contextualization.

Related Work

Smith et al. (2011) overview the landscape of player modeling in computer games. In their taxonomy, we are investigating individual, empirical, generative player models. Research in player modeling has typically addressed the challenge tailoring problem either by developing purely behavioral models or relying on predictions that ignore temporal changes in player data. Hunicke and Chapman (2004) model players by computing the average and variance of player damage and item inventory. Dynamic Difficulty Adjustment is achieved via a hand-crafted policy prescribing actions to take based on player health and inventory states. Magerko et al. (2006) interactive story players using a vector of competence levels for various skills and associated confidence values. The system selects from a pool of challenges based on a best fit between the characteristics of the challenge event and the current state of the skill vector. van Lankveld et al. (2008) role-playing game players using their progress and health, dynamically adjusting sets of presented enemies to enforce a given level of player health over progress. In contrast, our data-driven modeling approach explicitly forecasts changes in player performance, combines information across players, and proactively constructs a long-term set of challenges based on these predictions.

Subjective self-report indications of challenge have also been used to dynamically tailor game play (Yannakakis and Togelius 2011). Pedersen et al. (2009) train a neural network to predict player self-reported experiences of fun, challenge, and frustration based on measures of player behavior and in-game content. Yannakakis, Lund, and Hallam (2006) employ a neural network to predict player self-reported interest. Our approach extends these models by correlating time-varying measures of performance to self-report measures, enabling forecasts of player experience forward in time.

While we believe our work is the first application of tensor factorization to challenge tailoring problems, we note that similar techniques have been used to model student performance over time on standardized tests (Thai-Nghe, Horvath, and Schmidt-Thieme 2011).

Player Model

We explore tensor factorization techniques for modeling player performance in action role-playing games. While we focus on action role-playing games, we believe our techniques generalize to any games that make regular use of skill-based events. Tensor factorization techniques decompose multidimensional measurements into latent components that capture underlying features of the high-dimensional data. Tensors generalize matrices, moving from the two-dimensional structure of a matrix to a three or more dimensional structure. For our player modeling approach we extend two-dimensional matrices representing player performance against particular enemy types to add a third dimension representing the time of that performance measure.

We chose to use tensor factorization due to its favorable scaling properties, ability to cope with missing data, high accuracy, speed in generating predictions, and previous success in other applications. Tensor factorization is an extension of matrix factorization, which has been used in collaborative filtering applications—such as the Netflix Prize data mining competition—to great success. Matrix factorization offers the key advantage of leveraging information from a group of users that has experienced a set of content to make predictions for what a new group of individuals that has only been partially exposed to that content will do. Specifically, user data is represented in a $M = U \times I$ matrix indicating user preference ratings on items and decomposition extracts latent factors relating to users and items. Tensor factorization adds more dimensions, such as time, and extracts latent factors related to these other dimensions as well. Matrix and tensor factorization scale effectively to large numbers of users and items, handle missing information from users and achieve high accuracy (Koren and Bell 2011; Su and Khoshgoftaar 2009).

Formally, we represent player data in a tensor $Z = U \times I \times T$. In our simple spell-casting action role-playing game (described in the next section), U is the player ("user"), I is the spell type ("item") and T is the time of the performance recording. CP decomposition is a generalization of singular value decomposition from matrices to tensors. In CP decomposition the three-dimensional Z tensor is decomposed into a weighted combination of three vector latent factors,

$$Z \approx \sum_{k=1}^{K} \lambda_k w_k \circ h_k \circ q_k$$

where $\circ$ is the vector outer product, λ_k are positive weights on the factors, w_k are player factors, h_k are spell type factors, and q_k are time factors. K is the number of components used for each factor as an approximation of the true structure, keeping the set of the K most important components found (Kolda and Bader 2009). The decomposition can be computed by minimizing the root mean squared error between the factor inner product above and true data values, iteratively fitting each of the factors while fixing values of the other factors until convergence. We employ the N-way toolbox (Andersson and Bro 2000) to perform this decomposition. Predictions in this model consist of taking the inner product of these three factors, a computationally efficient process.

Experiment

We focus on skill-based aspects of games that require procedural knowledge: the ability to correctly act in given circumstances, typically with limited time for decision-making. To test our tensor factorization model we conducted a study of player learning in a turn-based role-playing game. Below we present the game used for the study, study methodology, and the results of comparing our modeling technique to related approaches.

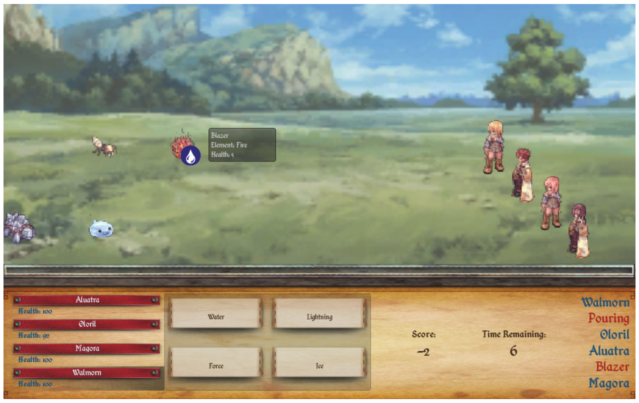

Figure 1: A battle between monsters and the player team.

Table 1: Spell Effectiveness Matrix.

Attack ↓ Def. →	fire	water	acid	ice	light.	earth	force	undeath
fire	1	0	1	2	1	2	1	0
water	2	1	0	1	0	1	2	1
acid	1	2	1	0	1	0	1	2
ice	0	1	2	1	2	1	0	1
lightning	1	2	1	0	1	0	1	2
earth	0	1	2	1	2	1	0	1
force	1	0	1	2	1	2	1	0
undeath	2	1	0	1	0	1	2	1

Game Domain

We implemented a turn-based role-playing game in which the player leads a group of four characters through a sequence of spell-casting battles against groups of enemy monsters. For the purpose of this study, we ignore the quest-like contextualization of the battles. See Figure 1 for the game's battle interface. Turns are limited to 10 seconds to require mastery of a complex spell system.

Each enemy is associated with one of eight spell types and player-controlled characters can attack with four of the eight possible spells. Casting a particular spell against an enemy of a particular type results in an attack being effective, ineffective, or super-effective, resulting in normal damage, no damage, or double damage against an enemy (see Table 1).

We intentionally created a spell system that was difficult to completely memorize, but contained intuitive combinations—water spells are super-effective against fire enemies—and unintuitive combinations—undeath spells are super-effective against force enemies—ensuring that skill mastery could only be achieved by playing the game. Note that pairs of spells—e.g., fire and force—are repeated in Table 1. This ensures a simpler underlying skill structure for players to learn; there are effectively only four spells. A scoring system based on spell effectiveness motivates players to learn; effective spells earn two points, ineffective spells earn zero points, and super-effective spells earn five points. Enemy attacks decrease player score by one. Player characters were assigned different spell sets, forcing players to learn the spell system.

Study Methodology

We recruited 32 participants to play our role-playing game, which was simplified to present a series of battles one after another, as in Figure 1. In our study we first gave players five minutes to review a text document explaining the spell system and the game interface. Once familiar with the game, players completed the sequence of eleven battles while we recorded the performance of the player in terms of effectiveness of spells chosen against enemies. After each battle we additionally asked players to report how difficult and enjoyable the battle was on a 5-point Likert scale.

We recorded each spell players cast on each enemy on each turn and battle in the game along with the associated performance value: 0 for ineffective, 1 for effective, and 2 for super-effective. Because spell effectiveness determines damage to enemies, player behavior traces varied in the number of turns taken, but had the same number of battles. We average performance for each spell type across all turns within a given battle, leaving the performance value missing for any spells not used in the battle.

We hypothesize that a tensor factorization approach will outperform non-tensor approaches. The tensor factorization model organizes player data in a three-dimensional tensor (player, enemy spell type, battle number), where each point in the tensor is the average performance of the player in attacking foes of a given spell type during a given battle. Spell types not used in a given battle were recorded as missing values. This produces a $32 \times 8 \times 11$ tensor for our 32 players, 8 spell types, and 11 battles. We compared the tensor model to two other models: matrix factorization using an unfolded tensor, and matrix factorization averaging over time. The matrix unfolding of this tensor simply concatenates battles column-wise, producing a 32×88 matrix recording player performance on each spell type and battle number combination. The final matrix factorization approach averaged player performance against spell types across all battles, removing any temporal information. Data points represent average player performance over their entire battle history against an enemy type. If our hypothesis is confirmed, then the tensor model captures additional structure lost in the unfolded matrix model when all time points are concatenated together.

We also hypothesize that there is an inverse relationship between objective, measurable player performance and subjective, self-reported difficulty. Should this hypothesis hold it will verify that in the context of action role-playing games we can use skill performance as a proxy for difficulty.

Results

We first compared players according to a variety of collected demographic information (age, gender, race, ethnicity, prior video game and role-playing game experience) and found no significant differences among these groups in our data set, validating the use of a single model across all players. We measured the 10-fold cross-validated *percent variance explained* of the tensor, matrix unfolded, and time-averaged models. Percent variance explained is an error measure that compares the total variation in the data across cases with the variation remaining after a model has been applied to the data. It describes how well a model captures variations in the data, with higher percentages indicating more powerful explanations.

The tensor model outperforms the other techniques for three or more components (see Table 2). While the tensor model does not achieve substantially better performance than the unfolded or time-averaged models on few components it shows much greater performance on larger numbers of components, reflecting its greater capacity to model complex underlying structure in the data. Notably, the tensor model shows comparable high performance for three, four, or five factors, indicating the spell matrix reflects an underlying structure testing the corresponding number of skills. As noted earlier our spell matrix actually only contains four spells (each spell has two names). This intuitively explains a result suggesting four underlying factors—the four unique spells—with similar numbers of factors reflecting a moderate amount of noise around this underlying information. The 100% score for the time-averaged model with 8 factors reflects the fact that this model is modeling 8 spell types with 8 factors and thus can trivially recover the same model.

components	tensor	unfolded	timeavg
2	92.59	90.82	95.86
3	92.18	89.44	72.41
4	88.72	86.30	39.99
5	90.11	61.77	45.02
6	89.11	63.66	24.28
7	87.11	24.29	36.30
8	80.05	-10.81	100.00

Table 2: Comparison of percent variance explained by different models using varying numbers of factors.

We evaluated the scaling of the tensor model in three ways: (1) how well the model scales with additional data; (2) how well the model forecasts into the future for players; and (3) how well the model scales in the number of players used by the system. In all three cases we use 10-fold cross-validated percent variance explained averaged over 10 repetitions of the experiment. Our first assessment hid a random subset of all our data and generated predictions for those missing values as shown in Figure 2. Model performance increased with the amount of total data used, with the simple two-component model showing high performance across all amounts of data while the more complex six- and eight-component models required approximately 60% of the data to achieve high performance. Thus, tensor factorization techniques perform well with this kind and amount of data, with more complex models requiring additional data but achieving high performance with more information.

Our second assessment evaluated the accuracy of the tensor model predictions on future events by hiding all future battles for a group of target players (Figure 3). The simple two-component tensor model achieved high performance regardless of the number of battles used, while the more complex six- and eight-component models required approximately seven battles to achieve comparable performance.

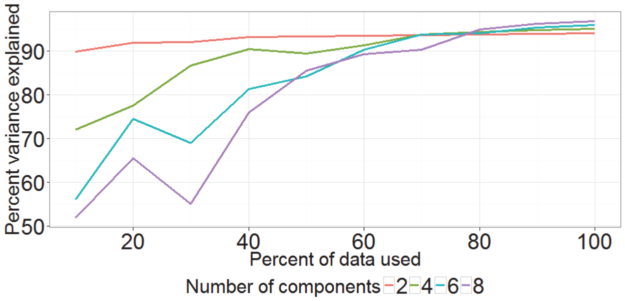

Figure 2: Percent variance explained of the tensor model as a function of the amount of data randomly subsampled from the data set, averaged over 10 repetitions.

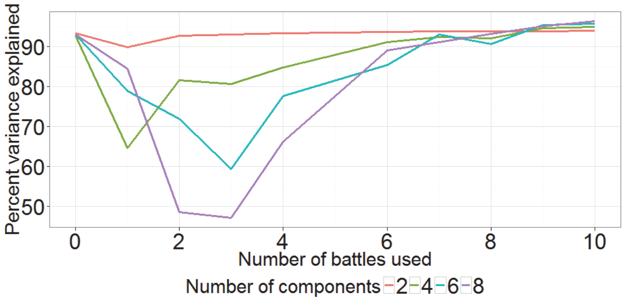

Figure 3: Percent variance explained of the tensor model when using first subset of battle data from randomly selected players, averaged over 10 repetitions.

Thus, tensor factorization models can generate useful forecasts after only a small number of battles observed for any given player. From a game design perspective, this suggests a relatively short training level in which the player can demonstrate skills can yield reasonable predictive power.

Our third assessment examined the tensor model accuracy when training and testing on varying numbers of players. We randomly subsampled from our full data set to use 6, 11, 16, 21, or 27 players and performed the same cross-validation analysis as above. Accuracy improved on the 2 component model from 81.17% variance explained with 6 players to 86.52% with 27 players, while the 4 component model improved from 52.55% to 86.29%, respectively. Other numbers of components showed similar trends, converging to approximately 86% accuracy with 27 players. The results of these three assessments demonstrate the power of the tensor factorization technique to scale with additional players and additional data for those players.

Finally, we found a significant correlation between objectively measure performance and subjectively experienced difficulty. Performance levels significantly differed (Kruskal-Wallis test for analysis of variance, $p < 0.001$) between subjective difficulty ratings. A Kruskal-Wallis test assesses whether responses (here performance) in different groups (difficulty ratings) shows significant differences—it is a non-parametric alternative to the standard analysis of variance tests that assume the data has a Gaussian distri-

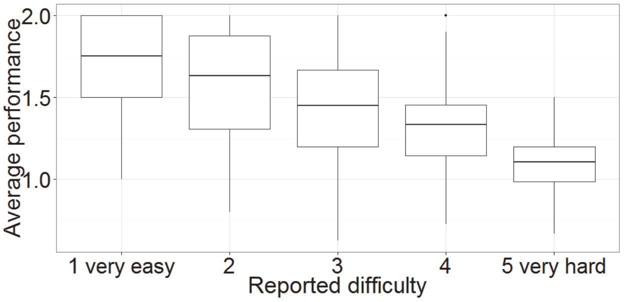

Figure 4: Comparison between objective performance values for subjective difficulty ratings, showing median values and the range from 25^{th} and 75^{th} percentiles.

bution, as our data showed skewing in performance measures for different response groups. A Dunnett's test found all adjacent pairs of difficulty ratings significantly differed ($p < 0.05$) in performance value, ranging from -0.11 to -0.22 (Figure 4). Dunnett's test applies a more powerful version of the Student's t-test to performance multiple comparisons among groups. Thus our second hypothesis is confirmed; objective measures of skill performance are inversely related to perceived difficulty.

Content Adaptation

Our temporal player model can predict player performance on skill-based events in the future. To perform challenge tailoring of a game, the player model must be combined with information about how to use the model. In the next sections we describe (1) a target expected performance to compare predictions of an individual user's performance against, and (2) an algorithm to select and/or parameterize skill-based events to bring predicted player performance in line with target performance.

Target Performance Curve

We expand upon the concept of a *performance curve* (Zook et al. 2012), as an indication of the desired player performance over a sequence of skill-based challenges. In our game this indicates desired player performance across a sequence of battles, provided by the human designer. Figure 5 shows an example of a performance curve. This particular curve seeks a decrease in performance over time until the very end of the game. This game should appear to a player to increase in difficulty from challenge to challenge, even as the player continues to master the skills involved. Other curves are possible as well. A curve expressed by $p = c$ (a horizontal line at a fixed constant, c) indicates a game in which the difficulty appears to remain the same, even as the player's skills improve. That is, the challenges may be parameterized to be more difficult, but because the challenge level is increasing at the same rate as player skill mastery, there should be little or no perceived change in difficulty. More complicated patterns, such as a series of rises and falls, can express complex designer intentions.

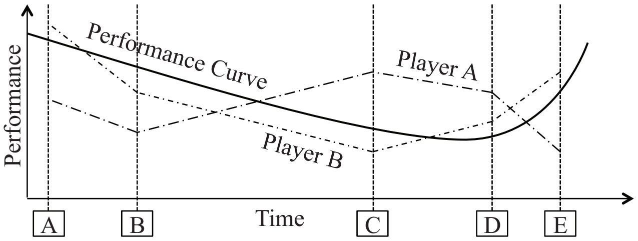

Figure 5: An example performance curve indicating a game that gets progressively more difficult until the very end, even as the player's skills improve. Dotted lines indicated predicted performance for two players.

Note that performance curves are specifications of desired player performance, rather than difficulty. Although our studies show that performance and difficulty are inversely correlated, there may be other aspects of the game—such as ambiguity in game information—that are not necessarily captured in the performance metric. A performance curve bypasses the ambiguity of subjective difficulty while enabling designers to specify how they desire a particular performance metric to vary over the course of an experience.

Challenge Tailoring and Contextualization

Given a performance curve and a temporal player model, challenge tailoring is the problem of selecting and spacing a sequence of skill-based events—in our case, battles—that match the predicted performance over time to the desired performance. This is a discrete optimization problem—battles are discrete units and the goal is to minimize the difference between predicted and desired performance values for these battles. A variety of techniques may be applied to solve these problems including constraint satisfaction, dynamic programming, and heuristic search techniques such as genetic algorithms (Smith and Mateas 2011; Togelius et al. 2011; Sorenson, Pasquier, and DiPaola 2011; Zook et al. 2012). In contrast to the reactive, near-term changes typically employed in DDA (Magerko, Stensrud, and Holt 2006; Hunicke and Chapman 2004), temporal player models are able to also proactively restructure long-term content to optimize a global player experience.

Challenge contextualization is the problem of selecting non-skill-based events that explain, motivate, or justify the skill-based events. Quest generation and narrative generation techniques may be used to provide this contextualization of skill-based events, although other, non-narrative contextualization may exist as well. Challenge tailoring and challenge contextualization may be performed in a pipelined fashion or in parallel. Pipeline techniques afford modular and compositional approaches to tailoring game content. In particular, if tailoring of skill-based events takes precedence over contextualization such that contextualization (i.e., the narrative, quest, or mission) can be sub-optimal then one might choose to solve the discrete optimization problem of selecting and spacing all skill-based events before "filling the gaps" with non-skill-based, contextualizing events (cf., (Magerko, Stensrud, and Holt 2006)). Fixing the skill-based events prior to contextualization makes challenge tailoring easier, but constrains the searchable context space.

Parallel techniques, conversely, may explore the full space of joint skill- and non-skill combinations, but trade this flexibility for greater complexity in the tailoring task. For example, we describe a technique that searches for a complete sequence of skill- and non-skill-based events that simultaneously optimizes for player performance and context (Zook et al. 2012). The question of whether to use a pipeline versus parallel approach is depends on the importance of contextualization relative to skill tailoring and the extent to which skill- and non-skill-based events appear seamless.

Conclusions

As computer games grow in popularity personalization of game content—which we cast as the paired problems of challenge tailoring and challenge contextualization—will also grow in importance. Unlike many other player modeling approaches to challenge tailoring and dynamic difficulty adjustment, we use a data driven approach that explicitly incorporates a temporal component. This temporal component allows us to more accurately forecast *future* player performance by modeling changes in a player's skills.

A performance curve provides authorial control over the game in the form of a target level of performance. Since objective performance is inversely correlated with subjective difficulty, a performance curve can guide an algorithm in the selection and parameterization of skill-based events over time. The tensor factorization model gives a system insight into how the player will perform on various sequences of challenges. With these tools, there are many discrete-optimization algorithms that can solve the challenge tailoring problem. Challenge contextualization, though outside the scope of this paper, can further ensure challenges make sense to the player while promoting player skill mastery.

Mastery of skills is correlated with fun (Koster 2005). A game that is able to tailor its difficulty to meet the abilities of the player may be perceived as more enjoyable to a wider range of players because it is never unintentionally boringly easy or frustratingly hard. This in turn has the potential to increase game replayability and make certain games accessible to a wider diversity of players.

Acknowledgments

The project or effort described here has been sponsored by the U.S. Army Research, Development, and Engineering Command (RDECOM). Statements and opinions expressed do not necessarily reflect the position or the policy of the United States Government, and no official endorsement should be inferred.

References

Andersson, C., and Bro, R. 2000. The N-way toolbox for MATLAB. *Chemometrics and Intelligent Laboratory Systems* 52(1):1–4.

Hunicke, R., and Chapman, V. 2004. AI for dynamic difficulty adjustment in games. In *Proceedings of the AAAI Workshop on Challenges in Game Artificial Intelligence*.

Kolda, T., and Bader, B. 2009. Tensor decompositions and applications. *SIAM review* 51(3):455–500.

Koren, Y., and Bell, R. 2011. Advances in Collaborative Filtering. In Ricci, F.; Rokach, L.; Shapira, B.; and Kantor, P. B., eds., *Recommender Systems Handbook*. Boston, MA: Springer. 145–186.

Koster, R. 2005. *A Theory of Fun in Game Design*. Paraglyph press.

Magerko, B.; Stensrud, B.; and Holt, L. 2006. Bringing the schoolhouse inside the box - a tool for engaging, individualized training. In *Proceedings of the 25th Army Science Conference*.

Pedersen, C.; Togelius, J.; and Yannakakis, G. N. 2009. Modeling Player Experience in Super Mario Bros. In *Proceedings of the IEEE Symposium on Computational Intelligence and Games*.

Smith, A., and Mateas, M. 2011. Answer set programming for procedural content generation: A design space approach. *IEEE Transactions on Computational Intelligence and AI in Games* 3(3):187–200.

Smith, A.; Lewis, C.; Hullett, K.; Smith, G.; and Sullivan, A. 2011. An inclusive view of player modeling. In *Proceedings of the 6th International Conference on Foundations of Digital Games*.

Sorenson, N.; Pasquier, P.; and DiPaola, S. 2011. A Generic Approach to Challenge Modeling for the Procedural Creation of Video Game Levels. *IEEE Transactions on Computational Intelligence and AI in Games* 3(3):229–244.

Su, X., and Khoshgoftaar, T. M. 2009. A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence* 2009:1–19.

Thai-Nghe, N.; Horvath, T.; and Schmidt-Thieme, L. 2011. Factorization Models for Forecasting Student Performance. In *Proceedings of the 4th International Conference on Educational Data Mining*.

Togelius, J.; Yannakakis, G.; Stanley, K.; and Browne, C. 2011. Search-Based Procedural Content Generation: A Taxonomy and Survey. *IEEE Transactions on Computational Intelligence and AI in Games* 3(3):172–186.

van Lankveld, G.; Spronck, P.; and Rauterberg, M. 2008. Difficulty scaling through incongruity. In *Proceedings of the 4th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*.

Yannakakis, G. N., and Togelius, J. 2011. Experience-Driven Procedural Content Generation. *IEEE Transactions on Affective Computing* 2:147–161.

Yannakakis, G.; Lund, H.; and Hallam, J. 2006. Modeling children's entertainment in the playware playground. In *Proceedings of the 2nd IEEE Symposium on Computational Intelligence and Games*.

Zook, A.; Riedl, M. O.; Holden, H. K.; Sottilare, R. A.; and Brawner, K. W. 2012. Automated scenario generation: Toward tailored and optimized military training in virtual environments. In *Proceedings of the 7th International Conference on the Foundations of Digital Games*.