Evolving Personalized Content for Super Mario Bros Using Grammatical Evolution

Noor Shaker[1], Georgios N. Yannakakis[1], Julian Togelius[1], Miguel Nicolau[2], and Michael O'Neill[2]
[1]Center of Computer Game Research, IT University of Copenhagen, Copenhagen, Denmark
[2]Natural Computing Research and Applications Group, University College Dublin, Ireland
{nosh, yannakakis, juto}@itu.dk, {Miguel.Nicolau, m.oneill}@ucd.ie

Abstract

Adapting game content to a particular player's needs and expertise constitutes an important aspect in game design. Most research in this direction has focused on adapting game difficulty to keep the player engaged in the game. Dynamic difficulty adjustment, however, focuses on one aspect of the gameplay experience by adjusting the content to increase or decrease perceived challenge. In this paper, we introduce a method for automatic level generation for the platform game *Super Mario Bros* using grammatical evolution. The grammatical evolution-based level generator is used to generate player-adapted content by employing an adaptation mechanism as a fitness function in grammatical evolution to optimize the player experience of three emotional states: *engagement*, *frustration* and *challenge*. The fitness functions used are models of player experience constructed in our previous work from crowd-sourced gameplay data collected from over 1500 game sessions.

1 Introduction

Procedural content generation (PCG) is a recently emerging area of research in game design. Several attempts can be found in the literature on automatically generating different aspects of content with or without a human designer involvement. Different machine learning and AI techniques have been explored and shown great potential for generating playable and aesthetically pleasing content.

PCG has been successfully used to generate various types of game content for different game genres such as racing tracks for racing games (Cardamone, Loiacono, and Lanzi 2011), rulesets (Smith and Mateas 2010), maps (Togelius, Preuss, and Yannakakis 2010), levels (Smith, Whitehead, and Mateas 2010; Sorenson, Pasquier, and DiPaola 2011), weapons (Hastings, Guha, and Stanley 2009) or even whole games (Browne and Maire 2010; Cook and Colton 2011).

Most of the research in PCG has so far focused on the offline generation of content, while fewer attempts have been made to create methods suited for online generation. Online content generation, however, is an interesting direction since it allows the generation of player adapted content by sensing the player's playing style and altering the generation process accordingly. This allows the personalization of the gameplay experience, the generation of endless content and makes the game infinitely replayable (Yannakakis and Togelius 2011).

Most of the attempts that could be found in the literature on online content generation tackle this problem by using heuristics for identifying different playing styles, and adjusting the game difficulty accordingly by changing the NPC behavior. Several attempts have emerged recently focusing on measuring the game difficulty and altering the gameplay experience by adjusting the difficulty assuming a direct link between challenge and fun (Iida, Takeshita, and Yoshimura 2002; Olesen, Yannakakis, and Hallam 2008). Some recent attempts focus on constructing accurate estimators of player experience using machine learning techniques (Hastings, Guha, and Stanley 2009; Shaker, Yannakakis, and Togelius 2012).

One of the techniques used to automatically generate content is evolutionary computation (EC). Grammatical evolution (GE) is the result of combining an evolutionary algorithm with a grammatical representation for automatic design (O'Neill and Ryan 2001), a domain where it has shown to have a number of strengths over more traditional optimization methods.

GE is used in this paper to construct personalized levels because it maintains a simple way of describing the structure of the levels and it allows the online evaluation of the content generated by defining the content quality measure as a fitness function.

In a previous study (Shaker, Yannakakis, and Togelius 2010), a content adaptation approach has been proposed and implemented based on exploring the search space of content using exhaustive search. This method worked well due to the small size of the search space explored in that study (12000 configurations of all possible values for four content features). In this study, we propose a general method for game adaptation by integrating the adaptation mechanism within the content generation process. This allows us to explore an astronomically larger content space.

In this paper we build on our earlier work on modeling player experience and automatic level design and extend it through (1) introducing an adaptation mechanism based on a GE level generator by employing a previously constructed player experience models as fitness functions for grammatical evolution, (2) conducting an experiment to examine the

efficiency of the adaptation mechanism on different playing styles, (3) evolving levels that optimize the player experience of the three emotional states: engagement, frustration and challenge and (4) conducting a thorough analysis on the obtained results to validate the adaptation mechanism and investigate the relationship between the emotional states.

The testbed platform game used is a modified version of Markus "Notch" Persson's *Infinite Mario Bros* (IMB). For a detail description about the game the reader may refer to (Togelius, Karakovskiy, and Baumgarten 2010).

2 GE-based Level Generator

Design Grammar

GE is a grammar-based form of genetic programming (GP) that uses a design grammar (DG) as its input to specify the syntax of possible solutions (O'Neill and Ryan 2001).

In the case of IMB levels, the design grammar is used to represent the underlying structure of the levels. The levels in IMB are constructed by placing different chunks in a two-dimensional map of predefined width and height. Each chunk can be one of the following: a platform, a gap, a tube, a cannon, a box, a coin, or an enemy. In order to allow efficient representation and variation in the design, the considered chunks have been categorized into two types: *obstruct-platforms* which block the path and enforce the player to perform a jump action, and hills that give the player the option to either pass through or jump over them.

The DG has been defined in a way that allows the placement of the chunks within the level map in no specific order. Each chunk is given a set of parameters that defines its characteristics; the list of parameters includes: the x and y coordinates of the starting position of the chunk; the height, h; the width of a gap, w_g; the width of the platform before w_{before} and after w_{after} specific game elements (gaps, tubes, and cannons); the horizontal spawning of a platform or a hill, w; and the number of coins, w_c. A simplified version of the DG is presented in Figure 1.

Enemies are placed on platforms after the level design have been constructed by evolving the position of each enemy and map it onto the possible positions of flat areas.

```
<level> ::= <chunks>  <enemy>
<chunks> ::= <chunk> |<chunk> <chunks>
<chunk> ::= gap(<x>,<y>, <w_g>,<w_before>,<w_after>)
  | platform(<x>,<y>,<w>)
  | hill(<x>,<y>,<w>)
  | cannon_hill(<x>,<y>,<h>,<w_before>,<w_after>)
  | tube_hill(<x>,<y>,<h>,<w_before>,<w_after>)
  | coin(<x>,<y>,<w_c>)
  | cannon(<x>,<y>,<h>,<w_before>,<w_after>)
  | tube(<x>,<y>,<h>,<w_before>,<w_after>)
  | <boxes>

<boxes> ::=  <box_type> (<x>,<y>)^2 | ...
  | <box_type> (<x>,<y>)^6

<box_type> ::= block_coin | block_powerup
  | rock_coin  | rock_empty

<enemy> ::= (koopa | goompa)(<x>) ^2 | ...
  | (koopa | goompa)(<x>) ^10
<x> :: = [5..95] <y> ::= [3..5]
<w_g> ::= [2..5]  <w> ::= [2..7]
<h> ::= [3..5]   <w_before> ::= [2..6]
<w_after> ::= [2..6]  <w_c> ::= [2..7]
```

Figure 1: A simplified version of the final grammar employed to specify the design of the level. The superscripts (2, 6 and 10) are shortcuts for the number of repetitions.

Fitness Function

The existing GEVA software (O'Neill et al. 2008) has been used to implement the needed functionalities of GE. The fitness functions used by GE are the player experience models constructed in our previous study to predict players' reported level of engagement, frustration and challenge from six key features of game content and players' playing style (Shaker, Yannakakis, and Togelius 2011). The inputs for each model are selected from statistical gameplay features calculated from the interaction between the player and the game and all six content features that have been included in the inputs in order to allow for control over the generated content, namely, the number of gaps in the level; the average width of gaps; the number of enemies; enemies placement, E_p which has been determined by three probabilities which sum to one: on or under a set of horizontal blocks, P_x; within a close distance to the edge of a gap, P_g and randomly placed on a flat space on the ground, P_r; the number of powerups; and the number of boxes.

It is worth noting that the levels used in that experiment were constructed using a very different content generator than the one proposed in this paper which has a different expressivity range (Shaker et al. 2012), and hence we anticipate that using the constructed models as fitness functions would give us only an estimation of the content quality.

The fitness value assigned for each individual (level) in the evolutionary process is the output of the model which is the predicted value of engagement, frustration or challenge. The output is calculated by computing the values of the models' inputs; this includes the values of the six content features which are directly calculated for each level design generated by GE and the values of the gameplay features estimated from the player playing style while playing a *previous* level. The search for the best content features that optimize particular affect is guided by the models prediction of the affective states, with a higher fitness given to the individuals that are predicted to be more engaging, frustrating or challenging for a particular player.

Experimental Setup

The GE experimental parameters used are the following: one run of 50 generations with a population size of 100 individuals. This allows a compromise between the efficiency in finding the best individual and the amount of time needed to apply a realistic online adaptation (each generation round takes about 10.4 seconds). The ramped half-and-half initialization method was used and the maximum derivation tree depth was set at 100, tournament selection of size 2, int-

flip mutation with probability 0.1, one-point crossover with probability 0.7, and 3 maximum wraps were allowed.

3 On-line Content Generation for Optimizing Player Experience

The methodology proposed for generating game content using GE has been employed to generate personalized content. While the level is being played, the playing style is recorded and then used by GE to evaluate each individual design generated. Each individual is given a fitness according to the recorded player behavior and the values of its content features. The best individual found by GE is then visualized for the player to play.

We assume that the player's playing style is largely maintained during consecutive game sessions and thus his playing characteristics in a previous level provide a reliable estimator of his gameplay behavior in the next level. To compensate for the effect of learning while playing a series of levels, the adaptation mechanism only considers the recent playing style, i.e. that which the player exhibited in the most recent level. Thus, in order to effectively study the behavior of the adaptation mechanism, it is important to monitor this behavior over time. For this purpose, AI agents have been employed to test the adaptation mechanism since this requires the player to play-test a large number of levels.

AI Agents

Two AI controllers submitted to the Gameplay track of the Mario AI Competition [1] have been adopted to test the adaptation mechanism. The first agent is based on the A* search algorithm and the second is based on a simple heuristic function written by Sergio Lopez (Togelius, Karakovskiy, and Baumgarten 2010). These two agents have been chosen because of their good performance and because they exhibit quite different playing styles. Figure 2 presents several statistical gameplay features extracted from 100 levels played by each agent where the differences in playing styles can be clearly observed; while the A* agent tends to spend most of the time running and jumping, Sergio's agent appears to be moving slower and performing jumps only when necessary which leads to more time spent to finish the levels compared to the time needed by the A* agent. In general, the A* agent shows better performance than Sergio's agent by winning more frequently. Using such controllers, we are able to test the adaptation mechanism ability to recognize different playing characteristics and evolve levels accordingly.

The methodology proposed to test the adaptation mechanism is as follows: each AI agent plays a test level while its behavior is recorded. The evolution process is then started and GE is initialized with a random population of level designs, each individual is ranked according to the predicted player experience it provides (as predicted by the player experience models) given the player behavior in the test level and the content features extracted from the level design. The levels are then evolved, and the AI agent plays the best level found. The same procedure is repeated for 100 rounds taking into account the player behavior in the previous round to evaluate the level designs in the current round.

[1] www.marioai.org

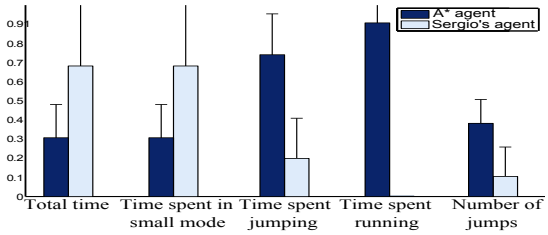


Figure 2: Average and standard deviation values of several gameplay statistical features that have been extracted from 100 different sessions played by the two agents.

Figure 3 presents the fitness values obtained for the best individual in each round when optimizing the level design to maximize engagement, frustration and challenge for the A* and Sergio's agent while monitoring the models' predictions of the other emotional states than the one optimized.

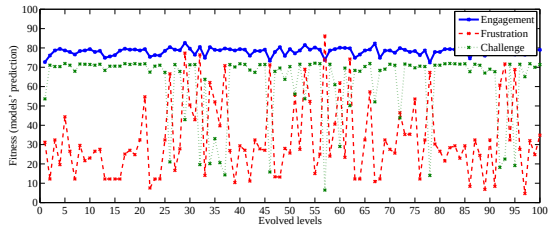
The results presented in Table 1 show that, using the adaptation mechanism, we were able to construct levels for the two agents with high predicted values across the three affective states. The adaptation mechanism appears to maintain the same level of predicted engagement for the two agents compared to the values obtained for frustration and challenge as can be seen from the standard deviations values.

By examining the gameplay data, we observed that most of the low prediction values obtained when predicting challenge are the cause of outliers in the time spent playing (very short or very long gameplay sessions). These outliers are the result of dying very early in the level or being stuck in a dead end as a result of the agents' incapacity to backtrack, which will eventually lead to losing the game by reaching the maximum session time allowed. The same argument holds for predicted frustration but with different effects; while short gameplay sessions resulted in a high level of frustration, being stuck in a dead end till the session time expires resulted in low prediction values of frustration. However, the length of the gameplay session appears to have less influence of the prediction of engagement.

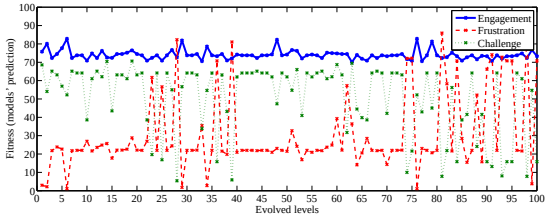
Table 1: Average fitness values obtained for the evolved levels across the three emotional states for each agent along with the standard deviations.

	Engagement	*Frustration*	*Challenge*
	A* agent		
Avg	$78.23\% \pm 1.88$	$79.06\% \pm 7.36$	$66.2\% \pm 10.23$
	Sergio's agent		
Avg	$74.01\% \pm 2.56$	$80.59\% \pm 10.58$	$61.25\% \pm 8.75$

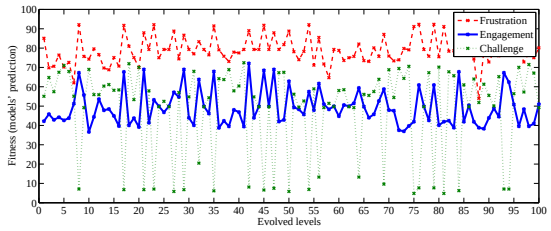
Figure 4 presents the best levels generated to optimize players experiences. Levels with varying content parameters have been evolved for each agent across the three emotional states. The levels can be analyzed according to the six content features optimized to maximize specific expe-



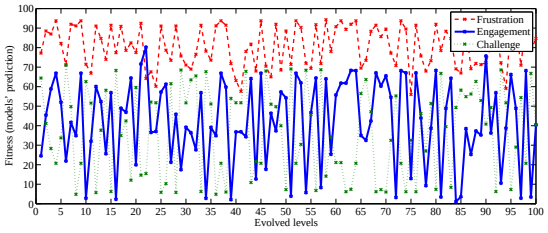
(a) Optimized predicted *engagement* for the A* agent.



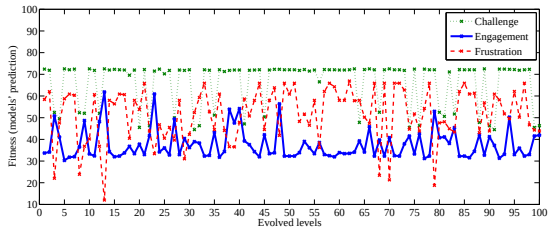
(b) Optimized predicted *engagement* for Sergio agent.



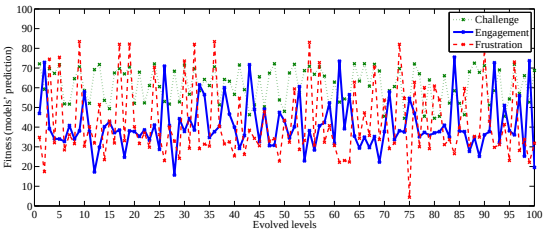
(c) Optimized predicted *frustration* for the A* agent.



(d) Optimized predicted *frustration* for Sergio agent.



(e) Optimized predicted *challenge* for the A* agent.



(f) Optimized predicted *challenge* for Sergio agent.

Figure 3: The fitness function for the optimized levels evolved for each emotional state while monitoring the models' prediction of the other emotional states.

rience. The diversity in the structure reflects the differences in playing style between the two agents; since Sergio's agent exhibit an average playing style (with no running and jumping only when necessary), the evolved level for optimizing engagement presents a flat level with some coins while the most engaging level for the A* agent (which plays more like an expert player by running and jumping throughout the level) contains gaps and enemies with no coins. The best levels evolved for optimizing frustration, on the other hand, exhibit more similar structure with both of them having the same number of gaps while differing in the number and placement of enemies; a small number of enemies scattered around gaps has been generated for the A* agent, while more enemies randomly placed around the level have been evolved for the most frustrating level for Sergio's agent. A slightly more challenging level with more gaps has been evolved for Sergio's agent than the one generated for the A* agent.

Figure 5 presents statistics for the six content features used to evaluate game content extracted from the 100 levels generated to optimize the prediction of the three emotional states. All values are normalized to the range [0,1] using standard max-min normalization. As can be seen from the figure, the most engaging levels evolved for optimizing engagement for the A* agent contain more and wider gaps with enemies mostly placed around gaps and blocks compared to less and narrower gaps with enemies scattered randomly around the levels observed for the levels optimized for Sergio's agent. The opposite observations have been obtained from the levels optimized for challenge. It hence appears that the approach was able to recognize the differences in the playing styles and consequently evolve levels with different characteristics to optimize particular player experience. On the other hand, similar average values for the content features have been observed from the maximum frustrating levels for the two agents.

4 Statistical Analysis

We performed statistical tests (correlation coefficients) to further investigate the results obtained, the method ability to recognize and adapt to particular playing style and the relationship between the three emotional states.

The first test has been conducted to test the ability of the adaptation approach to recognize particular playing styles and adapt accordingly. This is done by testing for significant differences in the predictions obtained from the 100 best levels evolved for each agent. The statistical test showed that the adaptation mechanism evolved significantly more

engaging and challenging levels for the A* agent than the ones generated for Sergio's agent while no significant difference has been observed for the levels evolved to maximize frustration for the two agents (significance is determined by $p < 0.01$). It, hence, appears that it is easier to generate engaging and challenging levels for a player with an expert style than for a player with an average playing style.

To investigate the intra-correlations between the three emotional states, we calculated the correlations between the predictions of an optimized affective state and the predictions of the other non-optimized emotional states. The results presented in Table 2 show that for the two agents, evolving engaging levels resulted in levels that are also challenging while generating challenging levels were found to generate levels that are less engaging for the two agents.

On the other hand, different effects have been observed between the predictions of engagement and frustration; while evolving engaging levels were also found to generate frustrating levels as evidenced by the predictions of these two emotional states to be positively correlated for the A* agent, these emotional states were found to be strongly negatively correlated for Sergio's agent. Evolving frustrating levels, on the other hand, resulted in levels that are also engaging for the two agents.

As for the relationship between predicted frustration and challenge, the correlation analysis showed that when optimizing predicted frustration, less challenging levels are generated for the two agents. However, predicted challenge and frustration were found to be strongly positively correlated when evolving challenging levels for the two agents.

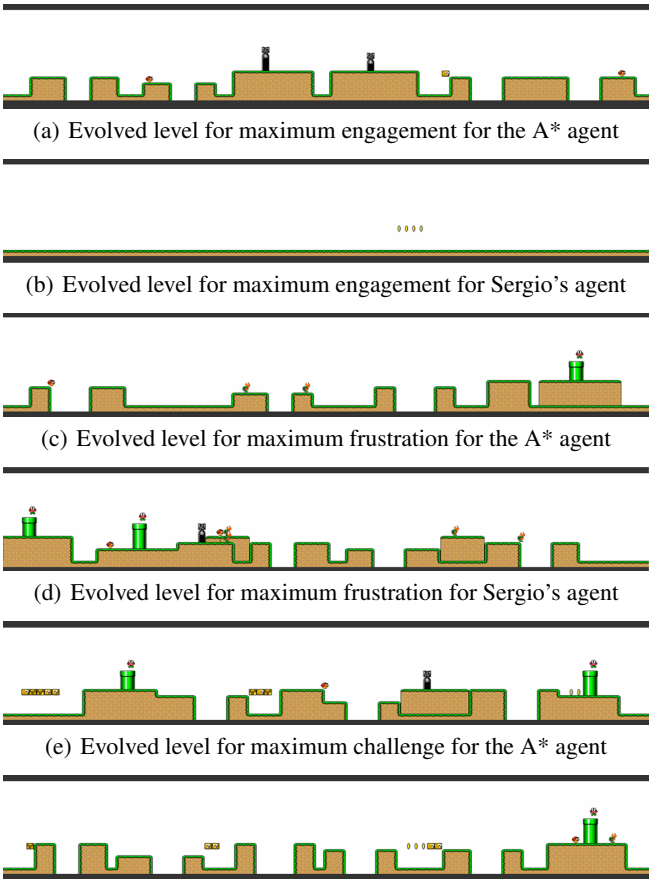
Table 2: Correlation coefficient values obtained between the predictions of engagement (E), frustration (F) and challenge (C) when optimizing each of these affective states (columns) while monitoring the prediction of the other emotional states (rows). Strong correlations are presented in bold.

	Optimized affective states					
	A* agent			Sergio's agent		
	E	F	C	E	F	C
E		**0.62**	**−0.34**		**0.613**	**−0.4**
F	0.23		**0.31**	**−0.45**		**0.36**
C	0.04	**−0.78**		0.12	**−0.67**	

This variety of relationships observed among the predictions of the emotional states between the agents reflects the method's ability to recognize different playing styles and evolve levels with different characteristics.

5 Conclusions and Future Directions

The paper introduced an approach for automatic level design and adaptation using grammatical evolution. A design grammar was defined to represent the underlying structure of the levels in Super Mario Bros and evolution was performed to select the best individuals based on their potential in optimizing a particular affective state for specific playing style. To this end, previously constructed player experience models that map game content to reported player experience



(a) Evolved level for maximum engagement for the A* agent

(b) Evolved level for maximum engagement for Sergio's agent

(c) Evolved level for maximum frustration for the A* agent

(d) Evolved level for maximum frustration for Sergio's agent

(e) Evolved level for maximum challenge for the A* agent

(f) Evolved level for maximum challenge for Sergio's agent

Figure 4: The best levels evolved to maximize predicted engagement, frustration and challenge for the two agents.

(engagement, frustration and challenge) were utilized as fitness functions. Two AI agents with different playing styles were used to test the adaptation approach. The results illustrate the method's ability to recognize differences in playing styles and generate content accordingly.

The analysis conducted to test for the method validity and the dependencies among the predictions of the emotional states showed that it is easier to optimize player experience of engagement and challenge for a player with an expert style than for a player with an average playing style.

The results obtained revealed interesting relationships among the predictions of the emotional states. Engaging levels were found to also be challenging in general while challenging levels are not necessarily engaging. The same observation has been obtained between predicted frustration and challenge; the features that play a role in generating challenging levels appear to also have a positive effect on the perceived frustration, while the influence of the features positively affecting perceived frustration has been observed to be negative on predicted challenge. The right amount of frustration should be present in a level for it to be engaging, this also depends on the playing style; while engaging levels are also found to be frustrating for some players, other

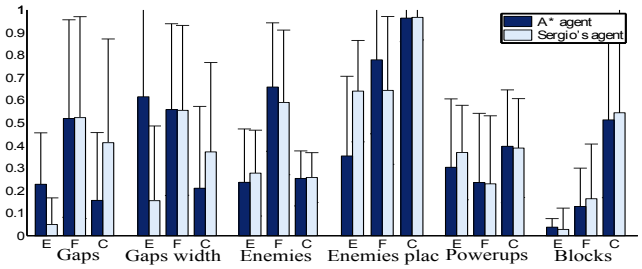


Figure 5: Average and standard deviation values of six statistical content features extracted from 100 levels evolved to optimize predicted engagement (E), frustration (F) and challenge (C) for the two agents. Enemies placement $E_p = 0$ when $P_g = 80\%$, $E_p = 0.5$ when $P_x = 80\%$ and $E_p = 1$ when $P_r = 80\%$.

players might enjoy less frustrating levels. Although this paper focuses on optimizing one aspect of player experience, it is interesting is to investigate the use of multi-objective optimization techniques to optimize, for example, engagement and challenge while minimizing frustration. This will help us explore new dimensions of player experience and ultimately generate content that maximizes some dimensions while minimizing others to optimize player experience.

The analysis presented focuses on optimizing specific predicted affective state for a particular playing style. Future directions include testing the method's ability to generalize over different playing styles. This could be tested by setting different players to play in turns and monitoring the content evolution, similar to an experiment conducted in our previous study (Shaker, Yannakakis, and Togelius 2010).

There is an important limitation imposed in the approach followed that concerns the use of player experience models constructed based on levels generated by a very different content generator. A previous study (Shaker et al. 2012) showed that the two generators have very different expressive ranges; while the previous generator generates content by varying the values of six content features with all other dimensions being fixed, the GE-based generator explores the content space without such limitations. It is, therefore, important to test and validate the adaptation mechanism with human players. This could be done by letting human players play and compare adapted and non adapted generated levels. It is also necessary to study the effect of the content features on players' judgment since we anticipate that new factors of level design will play a role in human judgment. The discovery of more relevant features will allow us to model player experience with better accuracy.

Acknowledgements

The research was supported in part by the Danish Research Agency, Ministry of Science, Technology and Innovation; project "AGameComIn" (274-09-0083).

References

Browne, C., and Maire, F. 2010. Evolutionary game design. *IEEE Transactions on Computational Intelligence and AI in Games,* 2(1):1–16.

Cardamone, L.; Loiacono, D.; and Lanzi, P. 2011. Interactive evolution for the procedural generation of tracks in a high-end racing game. In *Genetic and Evolutionary Computation Conference, GECCO*, 12–16.

Cook, M., and Colton, S. 2011. Multi-faceted evolution of simple arcade games. In *IEEE Conference on Computational Intelligence and Games (CIG)*, 289–296.

Hastings, E. J.; Guha, R. K.; and Stanley, K. O. 2009. Evolving content in the galactic arms race video game. In *Proceedings of the 5th international conference on Computational Intelligence and Games*, CIG'09, 241–248. Piscataway, NJ, USA: IEEE Press.

Iida, H.; Takeshita, N.; and Yoshimura, J. 2002. A metric for entertainment of boardgames: its implication for evolution of chess variants. In *IWEC*, 65–72.

Olesen, J.; Yannakakis, G.; and Hallam, J. 2008. Real-time challenge balance in an rts game using rtneat. In *IEEE Symposium On Computational Intelligence and Games, 2008*, 87–94. IEEE.

O'Neill, M., and Ryan, C. 2001. Grammatical evolution. *IEEE Transactions on Evolutionary Computation* 5(4):349–358.

O'Neill, M.; Hemberg, E.; Gilligan, C.; Bartley, E.; McDermott, J.; and Brabazon, A. 2008. Geva: grammatical evolution in java. *ACM SIGEVOlution* 3(2):17–22.

Shaker, N.; Nicolau, M.; Yannakakis, G.; Togelius, J.; and O'Neill, M. 2012. Evolving Levels for Super Mario Bros Using Grammatical Evolution. In *IEEE Transactions on Computational Intelligence and Games (CIG)*.

Shaker, N.; Yannakakis, G. N.; and Togelius, J. 2010. Towards Automatic Personalized Content Generation for Platform Games. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*. AAAI Press.

Shaker, N.; Yannakakis, G. N.; and Togelius, J. 2011. Feature Analysis for Modeling Game Content Quality. In *IEEE Transactions on Computational Intelligence and Games (CIG)*, 126–133.

Shaker, N.; Yannakakis, G. N.; and Togelius, J. 2012. Digging deeper into platform game level design: session size and sequential features. In *Proceedings of the European Conference on Applications of Evolutionary Computation (EvoApplications)*. Springer LNCS.

Smith, A. M., and Mateas, M. 2010. Variations Forever: Flexibly Generating Rulesets from a Sculptable Design Space of Mini-Games. *IEEE Transactions on Computational Intelligence and AI in Games*.

Smith, G.; Whitehead, J.; and Mateas, M. 2010. Tanagra: A mixed-initiative level design tool. In *Proceedings of the International Conference on the Foundations of Digital Games*, 209–216. ACM.

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.

Togelius, J.; Karakovskiy, S.; and Baumgarten, R. 2010. The 2009 mario ai competition. In *IEEE Congress on Evolutionary Computation (CEC)*, 1–8. IEEE.

Togelius, J.; Preuss, M.; and Yannakakis, G. 2010. Towards multiobjective procedural map generation. In *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*, 3. ACM.

Yannakakis, G., and Togelius, J. 2011. Experience-driven procedural content generation. *IEEE Transactions on Affective Computing* 2(3):147–161.