Proceedings of the Twenty-Ninth International
Florida Artificial Intelligence Research Society Conference

Dynamic Difficulty Adjustment in Tetris*

Diana Lora, Antonio A. Sánchez-Ruiz, Pedro A. González-Calero and **Marco A. Gómez-Martín**
Departamento de Ingeniera del Software e Inteligencia Artificial
Universidad Complutense de Madrid, Spain
dlora@ucm.es, antsanch@fdi.ucm.es, pedro@fdi.ucm.es, marcoa@fdi.ucm.es

Abstract

A game fun to play is the one that provides challenges to the players corresponding to their skills. Most of the games have different preconfigured difficulty levels, but they do not adjust the difficulty dynamically to the player skill. In this work, we explore the idea of creating clusters from previous game traces to capture different playing styles in Tetris and then use those clusters to decide how much help the system should provide to new players giving them good Tetris pieces. In our experiments players report improvements in terms of game experience.

Introduction

The main goal of any game is to entertain its users. According to the players psychological model describe by Daniel Cook (Cook 2007), players are driven by the desire of mastering new skills. Fun is achieved once the players overcome a challenge and masters a new skill. After that, the game gives rewards for the hard work and creates new challenges to conquer. The game creates a loop of learning-mastery-reward which needs to be balance in order to keep players interested.

A game fun to play is the one that provides challenges to the players corresponding to their skills. The difficulty is considered a subjective factor which is derived from the interaction between the player and the proposed challenge. This is not a static property, because it changes depending on the time spent by the player mastering a skill (Missura and Gärtner 2009; Hunicke 2005). With Dynamic Difficulty Adjustment (DDA) is possible to maintain the right balance depending on the player's skills. However, DDA usually involves a great amount of work for game programmers and designers that cannot be reused in other video games. In this context, research is important to decrease the costs related to development of adaptive games using different techniques such us, for example, automatically extracting information from traces of previous games. For the purpose of this paper, a game's trace describes the evolution of different variables during the game and it may contain data of the interaction between the user and the game as well as data about the virtual environment where user interacts.

In this paper we present our approach to DDA in Tetris, a very popular video game. We extract traces from previous games and build a case base in which each case describes how the user places a sequence of consecutive pieces in the game board. Then we use clustering to group the cases according to the skill level of the player. When a player starts a new game we look at his first movements to find the most similar cluster. Then the system provides dynamic help to the player choosing "good" Tetris pieces from time to time. In our experiments, using DDA users obtain higher scores and report improvements in terms of their game experience.

The rest of the paper is organized as follows. First, we discus the features extracted from the game traces to characterize the style of play, and the process to create different clusters to group games depending on the player's skill level. Then, we explain our proposed approach to provide help and dynamically adjust the difficulty of the game. Next, we analyze the results obtained in some experiments with new players when we apply DDA. The paper closes with related work, conclusions and directions for future research.

Tetris and Feature Selection

Tetris (Figure 1) is a very popular video game in which the player has to place different tetromino pieces that fall from the top of the screen in a rectangular game board. When a row of the game board is filled, i.e. it has no holes, the row disappears and all pieces above dropped one row. The pieces fall faster and faster as the game progresses until the board is full and the game is over (Breukelaar et al. 2004).

We use an implementation of the game called TetrisAnalytics that looks like an ordinary Tetris from the point of view of the player but provides extra functionality to extract, store and reproduce game traces. From these traces we can extract the most significant features, determine the skills of the player, and dynamically adjust the difficulty of the game to improve user experience. In order to build the training set, we collected game traces from 15 different players with diverse skills levels and they played around 300 games.

Each time a new piece appears on the top of the game board, the player has to make two different decisions. The first one, that we call *tactical*, is to decide the final location and rotation of the piece. The second decision involves how

*Supported by Spanish Ministry of Economy and Competitiveness under grant TIN2014-55006-R

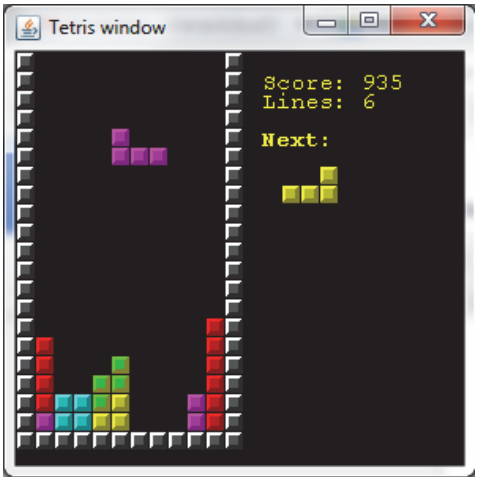

Figure 1: A screen of the Tetris game.

to *move* the piece to its final location (e.g. to rotate the piece twice and move it to the left 3 times). In this work we restrict our study to tactical decisions and how they define the skill level of the player. However, we think that we could also extract valuable information from the concrete movements (speed, cadence, movements undone, ...) and we expect to extend our work in the future.

For each tactical decision (i.e. for each piece in the game) we extract the following features:

- The current and next type of piece.
- The final location (row, column and rotation) of the piece.
- The state of each cell (free / occupied) in the highest 4 occupied rows of the game board. We only store those rows because there is a high chance that the player will place the current piece in that region.
- The points obtained by placing the current piece.
- The maximal points the player could have obtained if she would have performed the best possible action (according to a heuristic and greedy AI player that reduces the height of the board and the number of holes).
- The current score, number of completed lines and speed of the game.
- The height of the board (as the highest occupied row in the board).

Classifying Players Depending on their Skill Level

Unfortunately one single tactical decision is not enough to decide the skill level of the player with confidence, we need to consider longer sequences of pieces. The length of these sequences is an important factor because the more pieces we consider the better we can describe the style of play, but we will also have to wait longer to detect the skill level of a new player and adjust the difficulty of the game.

After some experiments we concluded that 10 tactical decisions (or pieces) is a good compromise. In order to select

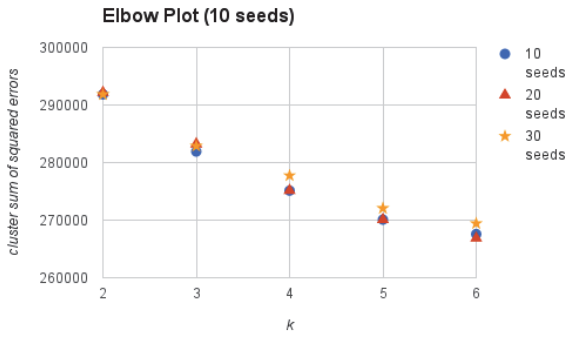

Figure 2: Elbow plot with the mean squared error as a function of the number of clusters.

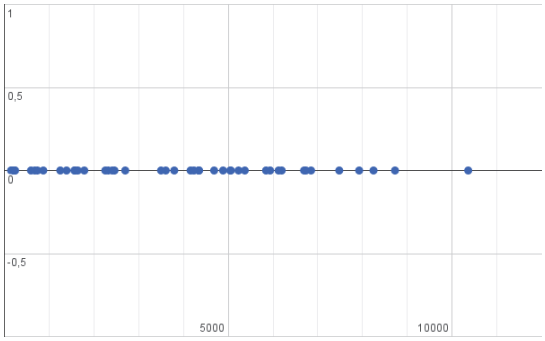

Figure 3: Total score of each game on the x axis.

this value we identified in the game traces a few players with similar scores but very different playing styles and then we performed several clustering algorithms joining the features of n consecutive pieces and increasing the number n. With $n = 10$ those players were classified in the same cluster so we concluded that we should wait at least 10 pieces before trying to predict the skill level of a new player.

From the game traces we created a case base in which each case stores the features of 10 consecutive tactical decisions. Each case represents a small fragment of the game and partially captures the player's playing style. Our next goal is to find clusters in the case base to identify different groups of players and then to use those clusters to predict the skill level of new players.

In order to select an appropriate number of clusters we used the same technique as Drachen et. al (Drachen et al. 2012a), using the k-means algorithm and varying k from 2 to 6. Figure 2 shows an elbow plot with the mean squared error as a function of the number of clusters. The fist clusters add much information (explain a lot of variance) but then the marginal gain drops. Figure 3 shows the total score of each game on the x axis. We can see in some regions a lot of points followed by another region with no point or just a few ones. From the two graphs, we can conclude that a good option for a simple game like Tetris is to establish three clusters that we will label as *newbie*, *average* and *expert*.

Figure 4 shows the scores of the cases assigned to each

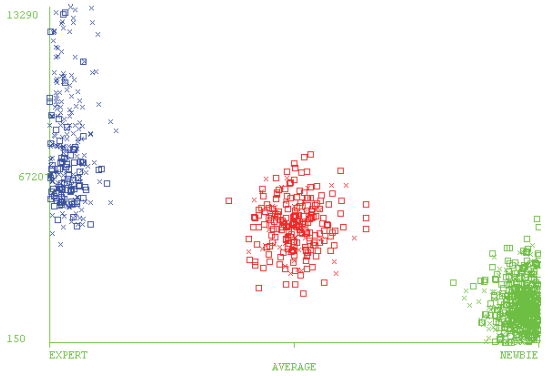

Figure 4: Total Score Vs. Clusters

Cluster	Cases	Characteristics
NEWBIE	34%	Homogeneous distribution of pieces along the board. Tendency to locate the pieces at the right and left sides of the board. Low rotation rate.
AVERAGE	17%	The distribution of majority of pieces along the board are in the lower half (rows 10 to 19). There is no preference to locate the pieces breadthways of the board. High rotation rate.
EXPERT	48%	The distribution of majority of pieces along the board are in the lower half (rows 13 to 19). There is no preference to locate the pieces breadthways of the board. High rotation rate.

Table 1: Clusters, size and interpreted playing styles.

cluster. We can see that each cluster is related to a group of players with a different skill level. Table 1 shows the percentage of cases in each cluster and some descriptions about the different playing styles that are more representative in each cluster. For example, *newbie* players tend to place the pieces more often on the left and right sides of the board than in the middle, and they rotate the pieces less than more experienced players. *Average* and *expert* players, on the other hand, play most of the time placing pieces in the lower half of the game board and only when the game is close to the end and the speed of the falling pieces is very high, they are forced to place the pieces in the upper area.

Dynamic Difficulty Adjustment

To do dynamically difficulty adjustment in a game, first we need to predict the skill level of the new player from the location of the first pieces in the game. As we explained in the previous section, in order to make a prediction with some level of confidence we need to wait until the player has placed at least 10 pieces in the game board. We also require the height of the board (the highest occupied row) to be over 1/3 of the total height. When both requirements have been fulfilled, we have at least one sequence of 10 pieces (probably more) to predict the player's skill level.

Newbie	Average	Expert
50%	30%	10%

Table 2: How often the system helps the user giving her a good Tetris piece.

The prediction is made by classifying the sequence of pieces in one of the three clusters. If there are several sequences of pieces available, the player's skill level is decided by a majority vote. This way, the skill level is decided based on the similarity between the current partial game and the beginnings of the games stored in the training set.

After we know the player's skill level (newbie, average or expert), we have to decide when and how to help them. The difficulty in Tetris depends mainly on two parameters: the type of pieces and the falling speed. In our work we decided to focus only on the former parameter because the new pieces that appear in the game are supposed to be random, and it is very difficult for a player to realize they are not random if we pick them carefully. The other parameter, the speed at which the pieces fall, is supposed to depend only on the number of lines cleared in the game, and we think it is easier for a player that plays several games to perceive changes in that parameter.

Another important decision to make is related to how often we help the user. Table 2 shows in how many pieces the system provides a *good* next piece. Obviously, the lower the level of the player the more help the system provides.

When the system decides to help the player, it checks how good each type of piece is in the current game board according to a heuristic function and selects one of the best three pieces randomly. For each type of piece we evaluate every board that can be obtained by placing the piece in every position and rotation, and then each type of piece is ranked according to its best final board. We use a very simple heuristic function that computes the number of empty rows in the board and subtracts the number of *holes* in the rows with pieces.

It is interesting to note that our first approach was to provided always the best piece but it turned out it was not a good idea because some pieces like the square or the stick are good quite often and they appeared too many times. It was notorious that the system was cheating and the players perceived it negatively.

Experiments and Results

We asked 16 different users to play 4 games of Tetris and then evaluate their game experience using a 5-point Likert scale. We provide help using DDA only in games number 2 and 4, while games 1 and 3 were normal Tetris games to compare the results. Of course, the users did not know in which games DDA was active.

Figure 5 shows the average score among all the players in each game. We can see that the players obtained higher

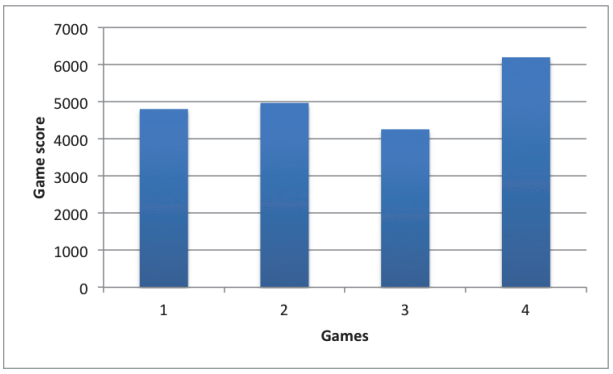

Figure 5: Average score in each game. DDA was active only in games 2 and 4.

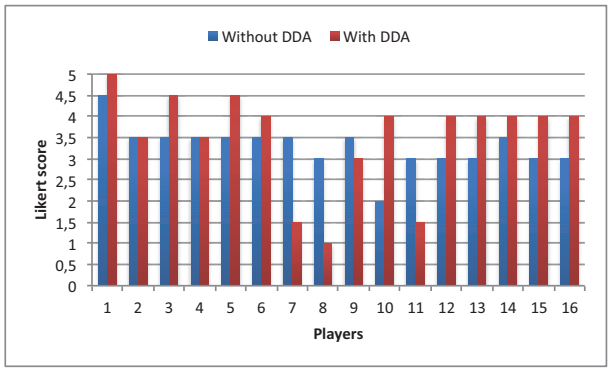

Figure 6: Game experience evaluation with and without DDA.

scores in games 2 and 4 in which DDA was active. Although we do not show it, these results are consistent with the duration of the games that was higher when DDA was active.

More interesting are the results shown in Figures 6 and 7 regarding the subjective evaluation of the game experience. Figure 6 shows the average satisfaction of each player in games with and without DDA active. In the x axis, we have the players and in the y axis the likert score they had given. For each player, the blue bar corresponds to the user experience in the two games DDA was not active. Whereas, the red bar is the user experience had when DDA was active and giving the user the "right" piece depending on their profile. We can see that, in general, the user satisfaction was higher with DDA active. Figure 7 shows the number of players that experienced DDA as a positive and negative effect. From 16 players, the game experience improved in 10 cases, did not affect in 2, and was worst in 4.

Although our results are preliminary, DDA seems to have a positive impact in terms of user satisfaction because the game adjust the difficulty according to the player's skills level. This way, when the player needs help, the game gives them on of three best possible pieces to score more points that would have.

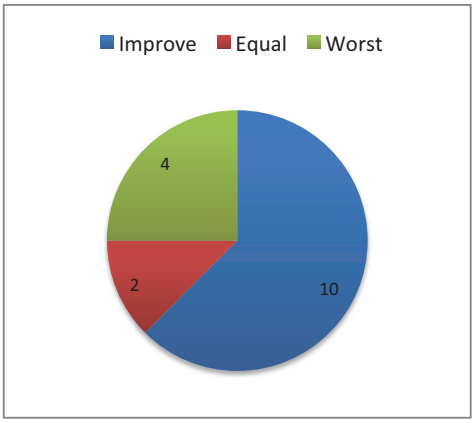

Figure 7: Number of users whose game experience improved with DDA.

Related Work

In video games, behavior analysis is a novel issue compared to other fields of application. One popular way of behavioral analysis is *segmentation* or *categorization*. Categorization is used for any analysis technique aiming at reducing a number of users in a few descriptive profiles, regardless of the method applied (e.g. segmentation, clustering y classification) (Drachen et al. 2012b). Players profiling can be used for testing, improvement of game design, creation of new monetization strategies, game customization, among others (Drachen, Canossa, and Yannakakis 2009; Mahlman et al. 2010).

With unsupervised learning techniques is possible to extract pattern from behavioral data without knowing to much about them (Drachen et al. 2013; Springer 2010b). The methods focuses on the structure and relationships between data, i.e. they look for patterns among features. Clustering is the process of grouping a set of objects such a way that the elements belonging to the same cluster are similar to each other and different from those that are part of other groups. Additionally, clustering allows reducing the dimensionality of the dataset (Springer 2010a). One category of cluster algorithms is centroid based clustering, like k-means (Lloyd 1982) which is often used in players profiling because of its popularity (Han and Kamber 2006).

On top of that, an entertaining game keeps the player so engaged that he could loose track of time. To make this possible, it is important to customize the game to their particular skills. This way the game won't present unintended consequences, that could frustrate the goals and aspirations of the user. For this reason, having a balanced difficulty level and consistency is very important in video games. Here is where becomes obvious that a game should provide challenges according to who is playing. With DDA is possible to modulate difficulty based on players interaction. In commercial games, the main objective is to make games fun and interesting in order to improve revenue. Whereas, in serious games helps to increase the learning level of the player (Missura and Gärtner 2009; Fields and Cotton 2011). One popular approach of DDA is *Rubber band AI* which cre-

ates a virtual link between the player and its enemies. This way, if the player "pulls" in one direction, i.e. if the user plays better or worse, then its enemies will be dragged in the same direction showing a simple or complex behavior (Missura and Gärtner 2009). Supervised learning techniques focuses on imitation, analysis and prediction of the behavior of the player; developing better opponents and dynamic adaptation of the difficulty of the game (King and Chen 2009; Missura and Gärtner 2009; Yannakakis and Hallam 2009).

Conclusions

Fun games provide challenges to the players according to their skills. As the player masters their skills the game should adapt itself to provide new challenges. In this work we have presented our approach to Dynamic Difficulty Adjustment (DDA) in Tetris. First, we collect traces from players with different skill levels and extract some features describing how they place the pieces in the game board. Then we build cases gathering together sequences of consecutive pieces and capturing fragments of the games. Next we find clusters in the case base to represent different playing styles and relate them to 3 skill levels: newbie, average and expert. Based on the skill level of new players, we provide some help in the form of "good" next Tetris pieces. Our experiments show that DDA has a positive effect for most players in their game experience.

As part of the future work we would like to extend our study to consider not only tactical decisions but the specific movements of the pieces until they are placed in their final positions. Our intuition is that the speed and number of times an expert player presses the keyboard is probably quite different from that of a newbie player. We would also like to explore other ways to capture playing styles and provide personalized help. For example, some particular player can have problems with a specific type of Tetris piece or with some configurations of the game board. Now, our difficulty adjustment is made once at the beginning of the gameplay when the pieces reached 1/3 of the board, but our intention is to dynamically identify when a player is receiving too much help from the game and it is necessary to recalculate their profile. On top of that, we would like to use clustering of time series data in order to create players profiles. This way, it won't be necessary to join several tactical decisions together to find chronological patterns in user behavior.

Finally, we would like to use our approach to other games and see at what extent we are able to personalize the difficulty using only traces from previous experiences. The use of data mining techniques in video games can help to minimize the cost and effort of the development team to implement DDA manually. Game traces contain valuable information to determine the profiles of new users and personalize the behavior of the game to each specific player. The selection of variables to characterize the player's skill level and the decision of which variables to modify in order to make the game easier or harder for the user, depend on the nature of each particular game. However, the data mining techniques described in this paper to extract information during game play, create clusters and identify different types of users are standard and can be used in any game. In summary, we strongly think that game traces contain valuable information regarding past experiences of other players and therefore they are a promising source of knowledge.

References

Breukelaar, R.; Demaine, E. D.; Hohenberger, S.; Hoogeboom, H. J.; Kosters, W. A.; and Liben-Nowell, D. 2004. Tetris is hard, even to approximate. *Int. J. Comput. Geometry Appl.* 14(1-2):41–68.

Cook, D. 2007. The chemistry of game design. http://www.gamasutra.com/view/feature/129948/the_chemistry_of_game_design.php. Consultado: 2015-05-26.

Drachen, A.; Sifa, R.; Bauckhage, C.; and Thurau, C. 2012a. Guns, swords and data: Clustering of player behavior in computer games in the wild. In *2012 IEEE Conference on Computational Intelligence and Games, CIG 2012, Granada, Spain, September 11-14, 2012*, 163–170.

Drachen, A.; Sifa, R.; Bauckhage, C.; and Thurau, C. 2012b. Guns, swords and data: Clustering of player behavior in computer games in the wild. In *2012 IEEE Conference on Computational Intelligence and Games, CIG 2012, Granada, Spain, September 11-14, 2012*, 163–170.

Drachen, A.; Thurau, C.; Togelius, J.; Yannakakis, G.; and Bauckhage, C. 2013. Game data mining. In *Game Analytics, Maximizing the Value of Player Data*. Springer. 205–253.

Drachen, A.; Canossa, A.; and Yannakakis, G. 2009. Player modeling using self-organization in tomb raider: Underworld. In Proceedings of IEEE Computational Intelligence in Games (CIG) 2009 (Milan, Italy).

Fields, T., and Cotton, B. 2011. *Social Game Design: Monetization Methods and Mechanics*. MK Pub.

Han, J., and Kamber, M. 2006. *Data Mining: Concepts and techniques*. Morgan Kaufmann.

Hunicke, R. 2005. The case for dynamic difficulty adjustment in games. In *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*, 429–433. ACM.

King, D., and Chen, S. 2009. Metrics for social games. Presentation at the social games summit 2009, game developers conference. San Francisco, CA.

Lloyd, S. 1982. Least squares quantization in pcm. IEEE Transactions on Information Theory 28(2).

Mahlman, T.; Drachen, A.; Canossa, A.; Togelius, J.; and Yannakakis, G. 2010. Predicting player behavior in tomb raider: Underworld. In Proceedings of the 2010 IEEE Conference on Computational Intelligence in Games.

Missura, O., and Gärtner, T. 2009. Player modeling for intelligent difficulty adjustment. In *Discovery Science, 12th International Conference, DS 2009, Porto, Portugal, October 3-5, 2009*, 197–211.

Springer. 2010a. Clustering. In Sammut, C., and Webb, G., eds., *Encyclopedia of Machine Learning*. Springer US. 180–180.

Springer. 2010b. Unsupervised learning. In Sammut, C., and Webb, G., eds., *Encyclopedia of Machine Learning*. Springer US. 1009–1009.

Yannakakis, G., and Hallam, J. 2009. Real-time game adaptation for optimizing player satisfaction. *IEEE Trans. Comput. Intellig. and AI in Games* 1(2):121–133.