Recommender system for board games

Jan Zalewski, Maria Ganzha
Department of Mathematics and Information Science,
Warsaw University of Technology, Warsaw, Poland
Email: j.zalewski,Maria.Ganzha@mini.pw.edu.pl

Marcin Paprzycki
Systems Research Institute Polish Academy of Sciences,
Warsaw, Poland
Email: Marcin.Paprzycki@isbpan.waw.pl

***Abstract*—Since the beginning of the twenty-first century, renaissance of interest in board games can be observed. As a matter of fact, share of board games within the entertainment market, it is growing constantly. For instance, local clubs where board games are played, as well as shops dedicated to this hobby materialize across Poland (but the trend is global). Since this hobby is relatively new, for example in comparison with watching movies, there are no applications that would recommend games to players. Here, the natural example could be the FilmWeb site, which recommends movies. At the same time, demand for good board game recommendations exists, which is illustrated by the popularity of game reviewers. Moreover, a huge database is available within the BoardGameGeek.com site, which contains information about games, including user-generated reviews. Hence, the idea to fill the existing gap, and to create a board game recommending system. In this note we report on the results of out work in this direction.**

***Index Terms*—Recommender system, board games, clustering**

I. Introduction

Historically speaking, board games could have been divided into two main categories. First, games that confronted skills of individual players in a one-to-one competition, such as chess, senet, checkers, mankala, etc. Second, games that involved element of chance (often combined with strategy considerations). Here, number of players can be larger than two, while moves involve roll of dice (or multiple dice) and move of a "pawn" by as many "fields" as it was suggested by the result of the roll.

Games began to dynamically develop at the beginning of the 20th century. The two largest markets became Germany and US. While the popularity of board games diminished during World War II, in the 1950s, board games evolved in different directions. In the US, more interest was devoted to role role-playing games, often simulating wars and allowing players to "take part in an adventure" (so called *ameritrash games*). These games did not shy away from reality, or even some brutality. In Germany, due to "war taboo", game designers focused on mundane subjects, e.g. planting beans or building a castle (so called, *European-style games*). These games began to focus on mechanics of the game and development of proper strategy. While this division remains visible, several other genres of board games appeared, such as:

- cooperative games (e.g. Pandemic by Z-Man), in which groups of people cooperate "against the" game instead of competing;
- war games (e.g. Axis & Allies by Milton Bradley), i.e. complex simulation games reflecting the realities of historical skirmishes;
- dexterity games (e.g. Pitch Car by Ferti), which involve manual efficiency, in addition to the strategic thinking and luck.

Finally, in recent years, hybrid games materialized, which combine multiple game types. For example, Mage Knight (by Wizkids), combines elements of the amerithrash game with mechanics typical for European-style games.

Since the community of players is large, they find various ways of seeking game recommendations. (1) They use forums, such as boardgamegeek.com and reddit.com, as well as gry-planszowe.pl (for Polish speakers). In there, special sections, for those seeking and recommending games, exist. Typically, seeker writes what games she liked so far, what she expects from the next game, and other users respond by suggesting games. (2) Some users check reviews on YouTube.com. (3) There are several well-known sources of reviews, e.g. Dice Tower (by Tom Vasel), Rahdo (by Richard Hama), Shut Up and Sit Down (by several UK-based players) or BoardGame-Girl.pl (by Ann and Jakub Polkowski; in Polish).

The main disadvantage of these approaches is that suggestions are, very often, the same "mainstream" games, with almost no recommendations of lesser-known titles. Hence, players, especially new to the field, when looking for games, they fall under the "halo effect" ([1]). There, when someone likes a given game then (s)he is looking for games of the same author, publishing house, or having a specific mechanics. Note that "manual recommendation-based solutions" are not scalable. Taking into account rapid growth of board game market[1] a different approach is needed. Missing is a recommender system, similar to that found for instance for movies (e.g. FilmWeb). The aim of this contribution is to summarize our work aimed at developing the needed application. Specifically, we have considered game reviews available at: BoardGameGeek.com site and attempted at using them in a game recommender system. Due to the limited space limitations, we proceed immediately to the description of the dataset used for our work and the experiments we have performed. Hence, we omit standard background material concerning recommender systems and machine learning techniques.

[1] https://www.prnewswire.com/news-releases/board-games-market---global-outlook-and-forecast-2018-2023-300763553.html

II. Available data and its preprocessing

As noted, we have decided to use information available within the BoardGameGeek.com service. This service provides an API for developers that allows them to download data in the XML format. Here, we have encountered a technical problem. When attempting to download over 200,000 records of data, the service (API) stopped responding to queries, because it recognized our attempt as a DDoS attack. To overcome this problem we have set delays in queries (though, as a result, total time that it took to get the entire dataset was about 12 days). Here, observe that the proposed approach does not depend on frequent (partial) updates, except for individual cases, like the newest titles (especially those that are very fast gaining popularity). In the case when someone wanted to turn our results into a production-class application, subsequent updates could be handled using additional scripts.

Initial analysis of collected data pointed out to existence of very large number of games with very small number of reviews. For example, out of 84,081 games, 24,534 do not have a single rating, 58,644 have less than 10 ratings, and 75,853 have less than 100 ratings. Obviously, games with very small number of reviews have limited value in being used for generating recommendations. It is likely that their rankings originate from relatives and friends of the game's author. Therefore, it was decided that only games with minimum 300 reviews are going to be considered. Obviously, this restriction affects the recommendation process and one may want to investigate this effect further.

Here, let us note that there exist subsequent (refreshed editions) of the same game. However, such games have, usually, only minor improvements. This being the case, and since it is almost impossible to automatically verify how much did the game change in comparison with the earlier version, we have decided to "ignore" this problem and treat all versions of a given game as the same game.

In board games, an important role is played by add-ons/extensions, which make games more attractive. However, since they are not independent games (cannot be used without the original game) we have decided to not to include any additions in the dataset.

The last problem that had to be addressed, before starting data processing, was to check the available data for anomalies. It turned out that several games have inaccurate data (e.g., time of several years of gameplay, where games usually last below an hour; or number of players specified as few thousand). All such items have been removed from the final dataset.

This resulted in inclusion of 3081 games, in the final set used for clusterization and development of the recommender system discussed in what follows.

III. Game clusterization

The first step towards building a recommender system was clusterization of available data. In Figure 1, we present sample information available on a page of a game. This figure contains information that can be used for clustering.

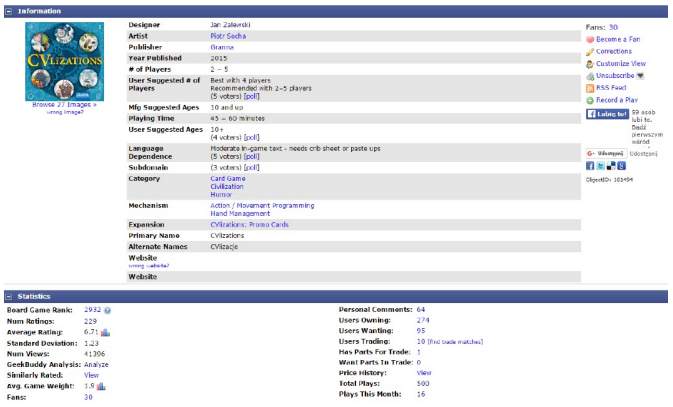

Fig. 1. Sample game description

As stated above, this information can be extracted as an XML demarcated content (via BoardGameGeek.com provided API). In Figure 2 we present sample XML file for the same game.

Fig. 2. Sample game data in XML

Overall, the following parameters have been available for game clusterization. Note that parameters 4-7 have been provided by game publishers (are available on the "game box").

1) *AverageRating* – arithmetic average rating (by users) expressed on a scale of 1-10, where 1 is the lowest (worst), and 10 is the highest (best).
2) *RatingsNumber* – number of ratings by users; minimal number of ratings (to be included) was 300; the largest number of ratings was 57332.
3) *GameWeight* – game difficulty; arithmetic average of the difficulty ratings (by users) expressed on a scale 1-5, where 1 corresponds to the simplest game, while 5 denotes the most difficult one.
4) *PlayingTime* – average playing time; typical duration of games is from 30 minutes to 3 hours; there are few very

short games (few minutes) and few games that can last up to several days.
5) *MinAge* – suggested minimum age designated by the publisher; usually it is from 4 to 14 years.
6) *MinPlayers* – minimum number of players; most often between one and three.
7) *MaxPlayers* – maximum number of players; most often between two and six.
8) *Rank* – position in the BoardGameGeek ranking; the higher the assessment by the users, the higher the position; portal owners to not provide complete details as to how the ranking is calculated (especially for games with very small number of opinions); overall, 84,081 games have been ranked (at the time of data collection).
9) *YearPublished* – year of the first game publication; the oldest game is Go (2200 BC); majority of games come from the late 20th and the 21st century.

A. Preliminary parameter analysis

Due to the lack of space we will present only summary of most important findings obtained about all (84,081) available games. Specifically, we have created histograms for all parameters. We have used all games, regardless of the number of reviews, to get a general overview of the dataset.

- *AverageRating* – for all games, average rating is close to 7, with most games scoring between 6 and 8.
- *GameWeight* – games are not very complex, majority of them score between 0 and 2.5, with the number of games with increasing complexity rapidly decreasing afterwards.
- *PlayingTime* – typical game duration is below 2 hours, with a "peak" at about 50 minutes.
- *MinAge* – as expected, minimum age for majority of games is below 15 years old, however clear peaks can be observed at 6, 10 and 12 (which is the standard categorization provided by game publishers).
- *MinPlayers* – over 75% of games require minimum of 2 players, which could have been expected as this is what makes marketing sense for game publishers.
- *MaxPlayers* – very few games allow more than 8 games to play.
- *Rank* – majority of (3081) games that were to be considered further (due to the fact that minimum of 300 opinions were required) were in the first 1000 of ranked games. However, interestingly, a large spike was observed at about 10,000 position. This was caused by very popular, but low ranked games (e.g. Monopoly or Mensch ärgere Dich nicht).
- *YearPublished* – as expected, majority of games available in the BoardGameGeek.com portal come from the most recent years.

B. Data correlation

After performing initial analysis of data, as related to the individual parameters, we have asked a question about parameter correlation. Note that existence of highly correlated parameters may reduce the quality of clusterization. After performing principal component analysis, we have established that:

- *AverageRating* and *Rank* are, obviously, related, because the higher the *AverageRating* the higher the rank (lower *Rank* value). The correlation coefficient turned out to be 0.85.
- *RatingsNumber* and *AverageRating* turned out to also be correlated. This connection follows from the way that the BoardGameGeek.com ranking algorithm works. Specifically, in order to minimize the possibility of manipulating rankings, they created an algorithm that initially assigns to each game large number of mid-point grades (5). Hence, to reach high rank, very large number of high grades is needed. Overall, the correlation coefficient is 0.28, which is also affected by existence of very large number of games with a very small number of ratings.
- *YearPublished* and *RatingsNumber* seemed like a natural candidate to be related. Out initial thinking was that the longer the game is on the market, the more people can get to know and evaluate it. However, the correlation coefficient turned out to be 0.1. Upon further scrutiny we have established that this is the "cult of the new" effect. Specifically, players are very often excited about the latest titles and want to play them, and evaluate them in large groups.
- *GameWeight*, *PlayingTime*, *MinAge* and *AverageRating* are all correlated. Here, for instance, *GameWeight* and *PlayingTime* have the correlation coefficient of 0.63. These correlations can be explained as follows: games with long play-time are considered "heavy" and are predominantly games designed for (and played by) older audiences. Moreover, "heavy games" are often highly ranked by the players.

IV. Clustering – preparatory analysis

During data preprocessing, values of all parameters have been normalized, with resulting data having variance equal to one and an expected value equal to zero. Next, one of the key questions had to be addressed: how many clusters should we try to generate. Here, we have decided to experiment with two methods of generating "best" number of clusters. Let us now briefly outline each one of them.

A. Elbow method

This method consists of analysis within sum of squares (WSS) in clusters, compared to the number of clusters [2]. In other words, we observe the variance resulting from clusterization. When the value of WSS decreases vis-a-vis the number of clusters, it is decided that the division is optimal. When the value of the WSS is plotted against the number of clusters, the optimal number of clusters "looks like an elbow", and a sudden decrease is observed (hence, the name of the method). Note that, unfortunately, this method does not involve any quantitative criteria. Therefore, since it is based on "visual" interpretation of charts, some situations may lead to different interpretations.

B. Silhouette method

Due to the lack of a quantitative criterion in the elbow method, we have decided to use one that includes such criterion. Hence, the Silhouette method ([3]) was selected. Silhouette statistics establishes how a given data point is suited for a given cluster, as compared to other clusters. Let us assume that for each point i, a_i is an average of squares of distances to other points in the cluster, while b_i is the minimum of squares of distances between i and the remaining points in the set. Then the Silhouette value s_i is defined as

$$s_i = (b_i - a_i)/max(a_i; b_i).$$

Values of s_i belong to the interval $[-1, 1]$. Values close to 1 indicate that i fits very well with the cluster (distance to the nearest point outside of the cluster is considerably larger that than average distance within the cluster). Negative s_i means that the point has been mistakenly assigned (distance to the nearest point outside of the cluster is smaller than the average distance within the cluster). It may also mean that the number of clusters is too small, or wrong (too small) is the number of clusters.

In the Silhouette approach, values of s_i are considered for the entire data set, depending on the number of clusters. The selected number of clusters is the one, which corresponds to the maximum of s_i.

C. Selecting clusterizations variables

After analyzing individual parameters, we have selected the final set of parameters to be used in clustering. Since *Rank* and *AverageRating* parameters are very closely correlated, we have decided to use the *AverageRating*, since it resulted in received a better separation in the Silhouette method and because, when two games are separated by only one position in the rankings, they can substantially differ in the average grade. Moreover, we have decided to not to include *RatingsNumber*, as it became obvious that it does not contain any information other than the "popularity of the game". However, this information (popularity) is partly included in the *AverageRating*.

These decisions have been based on application of the *Elbow* and the *Silhouette* methods to suggest the best number of clusters. Specifically, we have run both methods with all parameters, all parameters without *Rank*, and all parameters without *Rank* and *RatingsNumber*. In each case the same number of suggested clusters to be found was the same.

V. Data clusterization

Data clusterization was performed using standard K-means algorithm. We have decided that (as suggested by the Elbow and Silhouette methods) we will build 6 clusters. While we have also experimented with 5 clusters, results obtained for 6 are more insightful and thus they will be reported. Before proceeding, let us note that "cluster names" are based on direct experiences of the first author of this paper, who has played more than 800 games and is an author of a few. However, even in this case, it is impossible to manually check if all 3081 games have been associated with the "correct" cluster. Nevertheless, obtained results make sense, from the perspective of the active board game player and developer.

TABLE I
Results of clustering games into 6 clusters

Cluster #	# of elements	type of games
1	1110	gateway games
2	1059	traditional family games/mass-market games
3	15	party games
4	407	complex/hard games
5	476	family games
6	14	ancient abstract games

In the first Cluster there were 1110 games. These games are known as "gateway games", i.e. those from which people start their adventure with modern board games. Example titles from this category are *Carcassonne*, *Settlers from Catan*, *Town to the train*, *Alhambra*, *Blue Moon City*, *Mr. Jack*. These games are relatively short in time, and easy to play.

The second Cluster consists of 1059 games and is dominated by "family titles", which are also very easy to obtain (mass-market games). These games can often be found in homes. These games are somewhat more complex than games from Cluster #3 (though are still relatively simple) and are designed for smaller number of players. Usually, such games are addressed to families, and sample titles: *Rummikub*, *Monopoly*, *Stratego*, *Blokus*, *Coloretto*.

The third Cluster consists of only 15 games with simple rules and designed for a large number of people. Among people playing board games, for obvious reasons, these games are called "party games". Examples of such games are: *Jungle Law*, *Wolves*, *Bones of Mars*.

The next Cluster (#4) consists of 407 games. What distinguishes them from other games is level of complexity of rules and length of play time. As a matter of fact, manuals for these games can several dozen pages long. Interesting, in this cluster we find games representing both the "European" and "ameritrash" styles. This shows that the distinction between these two styles is purely imaginary and does not materialize in the dataset.

Cluster #5 consists of 476 games. These are mostly "family-friendly titles". They are somewhat more complicated than those in Cluster #2. Moreover, here a lot of "lesser known" games can be found. Examples are: *Sleuth*, *Bang!*, *Lost Valley*.

Finally, in Cluster #6, we find only 14 games. These are all ancient abstract games such as: *go*, *pachisi*, *backgammon*. Here one of defining characteristics is the small number of players (in most cases, these are two player games).

Overall, the most interesting (and unexpected) result was division of family-type games into two clusters. It can be claimed, that such division was imposed by the K-means algorithm, which was requested to deliver 6 clusters. However, upon in-depth analysis, from the point of view of game practitioners, such division makes sense. This, in turn supports the assumption that division into 6 clusters should be performed.

VI. Developing and applying user profiles

Let us now consider how the profile of the player can be represented. We have decided to develop the profile related to the available clusters. Obviously, there exists large body of literature related to user profile generation and personalization of information delivery (see, for instance [4], [5] and references collected there).

Nevertheless, we have opted for a somewhat simpler approach. We propose a simple aggregation of user preferences and application of the same approach that was used for game clustering. Specifically, assuming that the result of clustering already exists, we can see how the user rates games that are in a given cluster. Here, the assumption is that if the user evaluates games in a given cluster better than the average score, it means that (s)he likes games in this cluster. On the other hand, if (s)he scores them low, it means that (s)he does not like this cluster. To capture this, we propose the following formula.

$$v_j(u) = \frac{1}{n_j} \sum_{i=1}^{n_j} (x_{ji}(u) - \hat{x}_{ji}).$$

Here:

- j is the Cluster number (since we use a 6-cluster partition, $1 \leq j \leq 6$),
- n_j denotes number of elements (games) in Cluster j,
- i is an ordinal number assigned to a given game, belonging to the j Cluster,
- u is the player's ID (nickname),
- $v_j(u)$ is a numerical evaluation of how much the Cluster j matches preferences of player u,
- $x_{ji}(u)$ is the rating of game i from Cluster j by player u,
- $\hat{x}_{ji}$ is the average rating of game i in Cluster j.

After applying the formula, for each player we can obtain ratings for each cluster (numbers in the range $[-9, 9]$). Obviously, values close to -9 mean that a given cluster is really not liked, while values close to 9 indicate that such cluster is very close to user preferences.

Note that, on the BoardGameGeek.com portal, in addition to numerical evaluation, users can add comments. We have considered whether to take into account. Finally, we came to the conclusion that establishing "value" of a verbal evaluation is a difficult task. Furthermore, it would require NLP processing (similar to sentiment analysis used for social media), which was out of scope of our initial interests in the subject. Hence, we have decided to omit it.

A. Collecting user data

After establishing how to match users to the clustered game data, it was necessary to acquire player data. Here, we had to deal with the "cold start" problem (see, [6], and references collected there). In other words, to be able to see how the proposed approach works, we needed to have data of active users, to start with. Here, active means – large number of evaluated games. Hence, we have decide to download data form the BoardGameGeek.com. Since we were not able to do it (in a simple way) directly, we have decided to use the Friendless add-on, which allows registered users to "cross-link" their accounts. In this way we were able to obtain data of 200 most active users. Scripts to acquire the data from Friendless have been written in Python, while scripts for data acquisition from BoardGameGeek.com have been prepared in Java. Here, it should be noted that as mentioned above, the first author of this contribution knows most of users, whose data we have harvested, in person or through the community contacts. This allowed us to further cross-check obtained results.

B. Applying user profiles to facilitate recommendations

Let us now summarize what we have discussed thus far. We have created 6-cluster division of 3081 games. We have proposed a way in which we can create a user profile. Such profile will consist of six numbers from interval $[-9, 9]$ representing "closeness" of user to each of the created clusters (note that this approach generalizes also to other applications with similar structure). Obviously, user profiles will be "valuable" only when a user completes at least one evaluation of a game from each cluster. However, it should be getting better with increasing number of completed game rankings. To be able to investigate if our proposed approach makes sense, we have harvested data of 200 active users. Upon reflection we have came to the conclusion that there exist at least two approaches to providing recommendations.

Approach 1 – direct profile closeness

1) To find recommendations for user X
2) Find user Y, who is the closest (in profile) to user X
3) Find games rated by Y, but have not been rated by X
4) Sort them according to evaluations of Y and recommended the top N to X

Approach 2 – include favorite cluster into consideration

1) To find recommendations for user X
2) Find user Y, who is the closest (in profile) to user X and the favorite cluster C (of Y)
3) Find games, from C, rated by Y, which have not been rated by X
4) Sort them according to evaluations of Y and recommended the top N to X

The initial "test" has been performed by the first author, by applying both approaches "to himself". He has applied the algorithm from *Approach 1* and found that Zee Garcia, his favorite reviewer form the Dice Tower channel has been selected. Next, he has tried *Approach 2* and has found games that he has not played before, and he liked them. This gave us some confidence in the proposed approach.

Here, the problem of actual experimental verification has materialized. While we could have tried to run some experiments on selected users of BoardGameGeek.com, we did not have a chance to create a group that would be methodologically correct. For instance, we could have tried to work with the Friendless add-on and those who are cross linked with the first author, but these are "friends" and thus should be

disqualified from experimental validation of software whose author is (very well) known to them.

To address this problem we have tried a number of approaches. Let us briefly summarize them.

1) Backtracking: The idea was very simple and was based on addressing the following scenario. Assume that user X likes game G. Let us remove this game from the list of games she knows. Would the proposed algorithm recommend it? The realization of this idea was as follows. We have eliminated from the set of scores of X the highest ranked game, and performed the recommendation algorithm and checked if this game was recommended. After the first tests, it turned out that the algorithm did not always recommend the removed game. The problem was as follows. Let us assume that we are backtracking the recommendation process for user X and ignore her evaluation of game G. If user Y (the "closest one") did not judge G at all, or it had a low score, it had no chance to be recommended.

2) Extending search: Here, instead of considering evaluation of only one closest user (Y), scores of several nearest (let's call them Z_1 to Z_n) players could be considered. Here, the algorithm would proceed as follows: (1) for a given user X we find M nearest users (Z_1 to Z_M); (2) for each of selected users, search for the K highest ranked games, which X did not play and combine them into list L; (3) sort list L (combining scores if multiple users like them) and recommend to X. However, this solution would ignore information how close users are. Specifically, the nearest user would have the same "power" as the fifth. Moreover, question how many users to include (and why) remains open.

3) Cluster focused approach: It is possible to focus just on clusters. Knowing which cluster user U likes (e.g. cluster D) it would be possible to suggest highest ranked games from this cluster. The main problem with this approach is lack of, so called, *serendipity effect*. Since all games would be recommended from the same cluster, there would be no chance of finding an unusual game and expending ones "game horizon".

4) Ultimate solution: The final solution was based on the following idea. Instead of looking for a similar user, let's examine ratings issued by all players, while the "weight" of their evaluation will depend on the distance from the user. Hence, the algorithm presented itself as follows:

1) for a given user X, find set G of all games that (s)he did not play
2) for each game $G_i \in G$ find a set of grades (E) assigned by all others players
3) for each evaluation E_j (by user Y_j) the score is modified by the distance between X and Y_j
4) combine and normalize scores of all games; sort them and L top scorers.

It turned out that this algorithm gave the best results, and its only disadvantage is the operation time and potential lack of scalability (with increasing number of players, games and scores).

C. Testing

Let us now briefly summarize the tests. For each of two hundred users we have picked ten of their highest rated games. Here, since these were very active players, who have evaluated large number of games, all considered games had maximal (or almost maximal) rankings. Then, for each player, we have randomly removed one of the games from their "top 10 list". Then we have run the recommendation algorithm and distinguished two outcomes: (1) success – if the game was "back in the top 10", and (2) failure – otherwise. We have run 10 rounds and found out that success, defined as above, has been observed in between 56% and 68% of cases. Averaging the 10 experiments, success was observed in 64% of cases.

VII. Concluding remarks

This paper considers development of board game recommender system (which is a completely novel area for application of recommender systems). We have proceeded as follows. We have applied Principal Component Analysis to establish, which parameters describing games should be taken into account. Next, we have clusterized more than 3000 games that have been evaluated, by at least 300 users, within the BoardGameGeek.com portal. Obtained 6 clusters make sense from the point of view of board game players (have real-world interpretation). Finally, we have explored possibilities of representing user profiles and applying such profiles to obtain recommendations. While not experimenting on actual players, we have developed an approach that is coherent and delivers plausible results in 64% of synthetic cases.

The main issues that remain are: (1) cold start problem for new users in the system (how can the recommender system recommend them games if they did not complete any evaluations); (2) scalability of the developed algorithm for finding recommendations; and (3) how to handle systematic increase of number of game titles, users and recommendations (adaptability to incoming data). In the future, we plan to address these issues (as well as test the proposed system on board game players).

References

[1] E. L. Thorndike, *A constant error in psychological ratings*, Journal of applied psychology 4 no. 1, (1920) 25-29

[2] D. J. Ketchen and C. L. Shook, *The application of cluster analysis in Strategic Management Research: An analysis and critique*, Strategic Management Journal, v. 17 no. 6, (1996) 441-458

[3] R. C. de Amorim and C. Hennig, *Recovering the Number of Clusters in Data Sets with Noise Features Using Feature Rescaling Factors*, Inf. Sci. 324 no. C, (Dec.,2015) 126-145

[4] Maciej Gawinecki, Zygmunt Vetulani, Minor Gordon, Marcin Paprzycki, *Representing Users in a Travel Support System*. In: Proceedings of the ISDA 2005 Conference, IEEE Computer Society Press, Los Alamitos, CA, pp.393-398, 2005

[5] Grzegorz Frackowiak, Maria Ganzha, Maciej Gawinecki, Marcin Paprzycki, Michal Szymczak, Myon-Woong Park, Yo-Sub Han, *Considering Resource Management in Agent-Based Virtual Organization*, in: N. Nguyen, L. C. Jain (Eds.), Intelligent Agents in the Evolution of Web and Applications, Springer, Berlin, 2009, 161-190

[6] Mateusz Kruszyk, Maria Ganzha, Maciej Gawinecki, Marcin Paprzycki, *Introducing Collaborative Filtering into an Agent-Based Travel Support System*. In: Proceedings of the MaSeB Workshop, IEEE CS Press, Los Alamitos, CA, 2007, 439-443