ReSkill: Relative Skill-Level Calculation System

Chairi Kiourt
School of Science and Technology
Hellenic Open University
Patra, GR-26335, Greece
chairik@eap.gr

George Pavlidis
Athena Research Centre
University campus at Kimmeria,
Xanthi GR-67100, Greece
gpavlid@ceti.gr

Dimitris Kalles
School of Science and Technology
Hellenic Open University
Patra, GR-26335, Greece
kalles@eap.gr

ABSTRACT

The introduction of social dynamics in multi-agent environments with synthetic agents is an effective way to simulate real-life conditions. Nowadays there is a trend towards the integration of social dynamics in multi-agent virtual environments to better assess the performance of synthetic agents in competitive situations. This assessment is usually carried out using human rating methods, such as *Elo* and *Glicko*, two of the most widespread methods, primarily used for chess. This paper introduces a web-based system that was developed to provide a way for everyone to be able to use these well-known human rating systems in various multi-agent rating experiments. A large-scale experiment has been conducted and the results have been used to present and prove the functionality of the developed system.

CCS Concepts

• General and reference→Metrics • Computing methodologies→Multi-agent systems • Software and its engineering→Interactive games.

Keywords

Rating Systems; Performance Rating; Social Events; Strategy Board Games.

1. INTRODUCTION

DOI: http://dx.doi.org/10.1145/2903220.2903224Since complex problems began to be studied as Multi-Agent Systems (MAS), the study of Social Learning (SL) problems has become very exciting [1] [2]. Diverse scientific areas such as sociology, economics, computer science, mathematics and marketing use social learning as an Artificial Intelligence (AI) tool for developing MAS [2]. Ferber [3] argues that the two Social Organization (SO) extremes, namely cooperation and competition, may be autonomously studied or in combination, depending on the case at hand. As it is quite usual in such cases, the social environments are populated with game playing agents [4]. For a game agent, the social environment is represented by a game with all its components and entities [3] [4]. Learning in a game is said to occur when an agent changes a strategy choice in response to new information [4] [5] [6]. All relevant studies suggest that the simulation of complex social environments and the analysis of their data makes for a formidable problem of developing social learning mechanisms for agents. The need to developing a system with multiple rating methods arises from the need to evaluate players' or agents' relative skills and performances, amongst others, as we have already done in extensive experiments [7] using the *de facto* standard rating methods available. Although these rating systems are quite useful for Multi-Agent Systems, they sometimes lack consistency when compared to each other and this may be due to the not-very-accurate simulation of a human player by a synthetic agent. As a result, we still lack a proper, let alone standardized, rating method for virtual agents in competitive environments with social dynamics. To facilitate such a development, we have implemented various rating systems and integrated them all into a web application and offer it to the scientific community for use in various multi-agent experiments which involve scoring and ranking. We named this system **Re**lative **Skill**-Level Calculation System (ReSkill).

In the context of Multi-Agent Systems (MAS) endowed with learning capabilities, one needs to define a "game" to allow for two opponents (of varying strength, tactics and motives) to compete against each other, then to create an environment where arbitrary collections of agents compete against each other, given a limited amount of learning resources (time, allowable number of practice games, allowable number of defeats: one can really think of several such resources) and, then to design an evaluation toolkit to measure how two distinct groups of agents manage their intra-group training with respect to their inter-group face off. The allowable degrees of freedom for such experiments are more than a few; besides learning resources, one can experiment with a variety of learning mechanism configurations (thus, simulating different characters; for example, fast vs slow learners, risky vs conservative learners, exploiters vs explorers, etc.), as well as a variety of opponent selection mechanisms (opting to play against a stronger or a weaker opponent, opting to play against an opponent of unknown stature, etc.), all of them leading to a wealth of social interactions which can be recorded and analyzed with the objective of identifying interesting (or promising) behaviors.

The main contribution of the system presented in this paper, is to help evaluate an agent's performance with realistic rating methods [8] so as to facilitate the comparison between different playing characters and to help highlight the range of tools required to support the investigation of multi agent systems.

This paper is structured as follows: a section with a relevant bibliography review is provided to introduce the current state of the art in the field of rating methods for competitive games. Then comes a section that describes the developed system, its architecture, its functionalities and the user interface. The last section presents how this system was used to evaluate the results of a large-scale game playing multi-agent experiment. At the end, conclusions and future work plans are presented.

2. SKILL RATING METHODS

Rating systems were first used in chess to calculate an estimate of the strength of a player, based on the player's performance against an opponent. The *Ingo* and *Harkness* system was the first chess

SETN '16, May 18-20, 2016, Thessaloniki, Greece
© 2016 ACM. ISBN 978-1-4503-3734-2/16/05$15.00
DOI: http://dx.doi.org/10.1145/2903220.2903224

player rating system [9], initially used to allow the members of the United States Chess Federation (USCF) to track their individual progress in terms other than tournament wins and losses [9].

The Elo rating system was first introduced by Arpad Elo in 1960 as a simple skill calculation of players, based on their wins, their losses and their opponents in chess [10]. Chess, however, is a competitive two-agent system, where each agent's performance is based solely on its skill. In multi-agent systems, it is used as a calculation of fitness for many different learning or search algorithms, with promising results [11]. The Elo system assumes that each player has a skill that is drawn from a random distribution (an agent may have a "good" game or may have a "bad" game); it attempts to find the center of that distribution and converge to that value. The calculation is performed after each match, in a game between two agents A and B, with respective ratings RA and RB. Unrated players generally start with a rating of 800 Elo, which reflects poor playing or a beginner level.

The *Glicko* rating system was first introduced by Mark Glickman in 1995 as an improvement of the Elo rating system [12]. The Glicko rating system is a method for assessing a player's strength in games of skill, such as chess and go. The main contribution of this measurement method is "ratings reliability", the so-called ratings deviation (*RD*). *RD* measures the accuracy of players rating. After a game, the amount the ratings change depend on the *RD*: the change is smaller when the players' *RD* is low, and also when their opponents' *RD* is high. The *RD* itself decreases after playing a game, but increases slowly over time of inactivity. The current version, *Glicko-V2*, introduces the concept of rating volatility, σ [12]. A slightly modified version of the Glicko-2 rating system is used by the Australian Chess Federation. In Glicko rating systems, unrated players start with their rating set to 1500 and *RD* set to 350. A player's most recent rating is used to calculate the new *RD* from the previous with a specific set of formulas provided by the Glicko rating systems**.**

The *TrueSkill* rating system [13] has been successfully used for calculating players' rankings in commercial massively multiplayer online game (MMOG). TrueSkill employs a Bayesian inference technique for ranking players but has not been yet used or tested for evaluation in contexts where Elo and Glicko are applicable; therefore we have not yet invested in exploring its potential.

3. THE ReSkill SYSTEM

In order to contribute towards the evaluation of the usage of human performance rating systems in Multi-Agent Systems we have implemented the most well-known rating systems into an integrated web-based system that is open to the scientific community for extensive testing and evaluation of agents' performance. The entire system is built in JAVA based on the input–process–output (IPO) model [14]. The main operation of the systems is based on the idea of the Work Flow system model. As shown in Figure 1, the first step of the process is to read and analyze the input file. The next step includes processing of the players'/agents' data and rating to provide performance ratings. Finally, the system produces an output file for additional data analysis along with various performance curves.

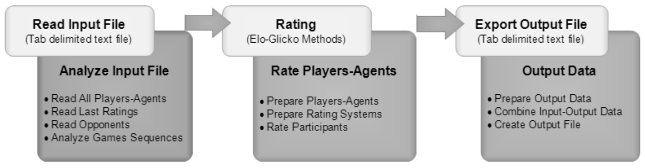
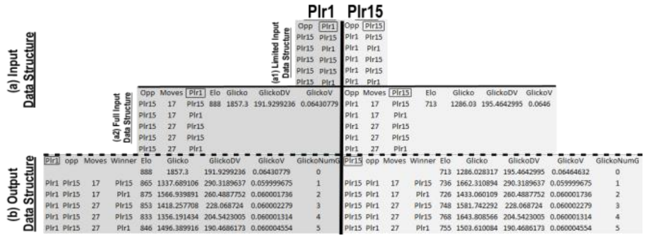


Figure 1. The workflow in ReSkill.

The input files contain the players/agents, their initial ratings (could be set to default values) and the results of all the matches between them in any competitive matching scenario, as shown in Figure 2 (a) for two different players/agents. Some generic additional information are also optionally included in the input files. It should be highlighted that the system supports two different input file structures:

- In the first structure, *Limited Input Data Structure* (LIDS) (Figure 2 (a1)), every player's/agent's input data is represented by two columns, the opponent and the winner. This input data structure could, alternatively, be constructed by using a specific tool of the system, where the user manually inputs the data values.
- In the second structure, *Full Input Data Structure* (FIDS) (Figure 2 (a2)), every player's/agent's input data is represented by seven columns, where the user has to provide the last known ratings for each agent, presented in seven (7) columns:
    - the first column of a player/agent shows the opponents,
    - the second column may be used as generic information about the game (like the average moves per game),
    - the third column shows the winner of the match,
    - the following four columns show the initial ratings of the player/agent (default values can be used based on the bibliography).


(a) Input Data Structure
(a1) Limited Input Data Structure
(a2) Full Input Data Structure
(b) Output Data Structure

Plr1

Opp	Plr1
Plr15	Plr15
Plr15	Plr1
Plr15	Plr15
Plr15	Plr15
Plr15	Plr1

Plr15

Opp	Plr15
Plr1	Plr15
Plr1	Plr1
Plr1	Plr15
Plr1	Plr15
Plr1	Plr1

Opp	Moves	Plr1	Elo	Glicko	GlickoDV	GlickoV
Plr15	17	Plr15	888	1857.3	191.9299236	0.06430779
Plr15	17	Plr1				
Plr15	27	Plr15				
Plr15	27	Plr15				
Plr15	27	Plr1				

Opp	Moves	Plr15	Elo	Glicko	GlickoDV	GlickoV
Plr1	17	Plr15	713	1286.03	195.4642995	0.0646
Plr1	17	Plr1				
Plr1	27	Plr15				
Plr1	27	Plr15				
Plr1	27	Plr1				

Plr1	opp	Moves	Winner	Elo	Glicko	GlickoDV	GlickoV	GlickoNumG
				888	1857.3	191.9299236	0.06430779	0
Plr1	Plr15	17	Plr15	865	1337.689106	290.3189637	0.059999675	1
Plr1	Plr15	17	Plr1	875	1566.939891	260.4887752	0.060001736	2
Plr1	Plr15	27	Plr15	853	1418.257708	228.068724	0.060002279	3
Plr1	Plr15	27	Plr15	833	1356.191434	204.5423005	0.060001314	4
Plr1	Plr15	27	Plr1	846	1496.389916	190.4686173	0.060004554	5

Plr15	opp	Moves	Winner	Elo	Glicko	GlickoDV	GlickoV	GlickoNumG
				713	1286.028317	195.4642995	0.06464632	0
Plr15	Plr1	17	Plr15	736	1662.310894	290.3189637	0.059999675	1
Plr15	Plr1	17	Plr1	726	1433.060109	260.4887752	0.060001736	2
Plr15	Plr1	27	Plr15	748	1581.742292	228.068724	0.060002279	3
Plr15	Plr1	27	Plr15	768	1643.808566	204.5423005	0.060001314	4
Plr15	Plr1	27	Plr1	755	1503.610084	190.4686173	0.060004554	5

Figure 2. Input-output files' structure, sample.

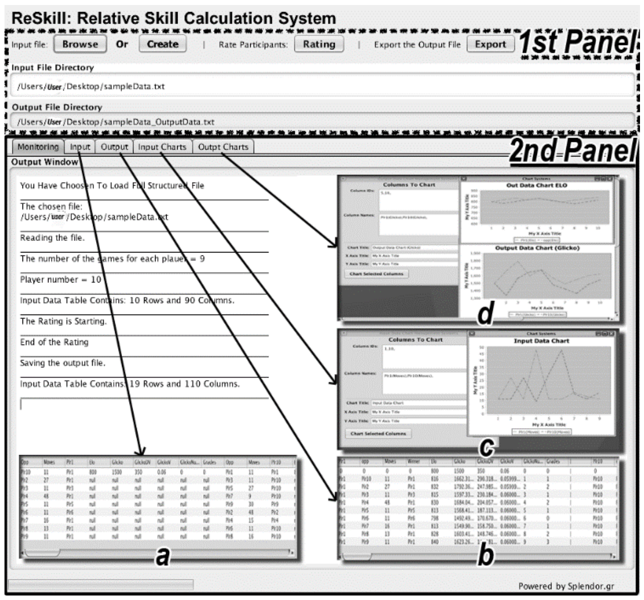
The output file has similar structure with small changes and some additional columns. Specifically, it consists of nine (9) columns where two new columns are added, as in Figure 2(b). The first column shows the agent under study whereas the last column is an incremental number that counts the matches. All the other columns are updated versions of the corresponding columns from the input files. Both input and output files are CSV structured for maximal interoperability and interconnectivity with external data analysis tools (and acceptable readability and editability by humans, which other data interchange techniques, like JSON, lack).

A web based Graphical User Interface (GUI) was developed to provide an integrated environment for further testing and experimenting with player's/agents' rating. Figure 3shows the web-based GUI of the application. For convenience the GUI is divided into two different panels, the first (upper) being the panel of the files and data and the second (lower) being the panel of the results.

In the first panel, the user chooses either to upload the input data or manually generate the data through the GUI. The second option is shown in Figure 3 (a) and it is limited on the creation of LIDS-type files. After uploading or generating the data, the "Rating" option provides the computation of the player's/agent's ratings. At the end, the "Export" option stores the results in a CSV text file; in case the input data were manually created then an input file is also created and stored.

The second panel of the system provides several tab panels with various functionalities:

- The first tab (Monitoring) provides full information for the entire process of a rating session.
- The next two tabs show the input and output data files in an interactive dynamic table format (Figure 3 (a) and (b)). All the cells of both tables are editable. By selecting a label of a column in the Output-tab table, the column is automatically added for plotting to the corresponding chart tab (Figure 3 (c) and (d)). Also, each row of the Output-tab table, dynamically produces bar graphs of the ratings of the row, in a variety of combinations. This charting system may provide rating status for every row of the Output-tab table. The user may create multiple charts that are dynamically managed by the internal pop-up window management system of each tab. In addition, the chart's properties (chart title, axis titles, resolution etc.) may be managed through the graphical user interface, furthermore, all charts may be exported to several image formats.
- The final two tabs host the graphs generated for the input and the output data respectively.


ReSkill: Relative Skill Calculation System
Input file: Browse Or Create
Rate Participants: Rating
Export the Output File Export
1st Panel
Input File Directory
Output File Directory
Monitoring Input Output Input Charts Outpt Charts
2nd Panel
Output Window
Columns To Chart
Out Data Chart ELO
Output Data Chart (Glicko)
Input Data Chart
Chart Selected Columns
a
b
c
d
Powered by Splendor.gr

/Users/User/Desktop/sampleData.txt

/Users/User/Desktop/sampleData_OutputData.txt

You Have Choosen To Load Full Structured File

The chosen file: /Users/user/Desktop/sampleData.txt

Reading the file.

The number of the games for each plauer = 9

Player number = 10

Input Data Table Contains: 10 Rows and 90 Columns.

The Rating is Starting.

End of the Rating

Saving the output file.

Input Data Table Contains: 19 Rows and 110 Columns.

Figure 3. Web-based GUI of ReSkill.

A demo of the platform along with usage instructions and sample input and output files can be found at http://reskill.splendor.gr

4. CASE STUDY: A LARGE-SCALE GAME PLAYING MULTI-AGENT SYSTEM

In order to test the functionality of the developed system, we used the experimental results produced by running a set of matches in *RLGame* [15] and specifically its tournament version, *RLGTournament* [4]. The configuration of this large scale tournament was as follows: 126 agents, all with different characteristics, were used in a round-robin tournament with 100 games per match (each match was repeated 100 times). Each agent played 125 matches against different agents, resulting in a total number of 787,500 experiments.

4.1 RLGame

The game used for the experiments was the RLGame [15], a strategy board game, which is played on an *n x n* square board by two players and their pawns. Two *a x a* square bases on opposite board corners are initially populated by *β* pawns for each player, with the white player starting from the lower left base and the black player starting from the upper right. The goal for each player is to move a pawn into the opponent's base or to force all opponent pawns out of the board (it is the player and not the pawn that acts as an agent in this scenario). The base is considered to be a single square, therefore a pawn can move out of the base to any adjacent free square. Players take turns and pawns move one at a time, with the white player moving first. A pawn can move vertically or horizontally to an adjacent free square, provided that the maximum distance from its base is not decreased (backward moves are not allowed).

The learning mechanism of each agent is based on approximating its (reinforcement-learning-inspired) value function with a neural network [2] [3]. Each autonomous (back propagation) [16] neural network is trained (Figure 4) based on an input of board positions along with some flags on overall board coverage. A single node in the output layer denotes the extent of the expectation to win when one starts from a specific game-board configuration and then makes a specific move.

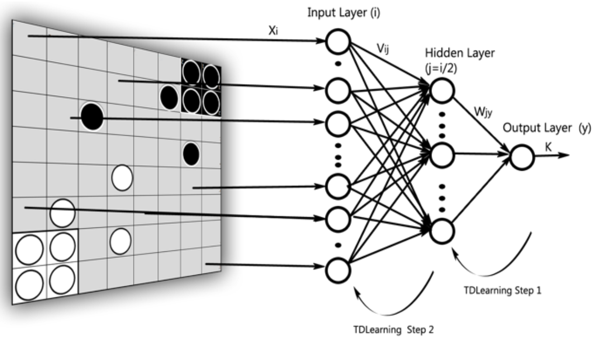


Figure 4. RLGame and learning mechanism.

RLGame was transformed into a tool for studying multi-agent systems via its tournament version, RLGTournament [4].

4.2 Analysis of Experimental Results

Figure 5 and 6 demonstrate the data and ratings analysis results of the presented experiment. Specifically, Figure 5 presents the comparison of two randomly selected agent's (*Plr1* and *Plr10*) evolution (progress) based on their *Elo* results after each game, as provided by the charting tool of *ReSkill*. As it may be difficult to read a graph for a very large number of games per agent (about 12,000 for each one, in our case), *ReSkill* allows for the customized rendering of some graphs to make them more readable.

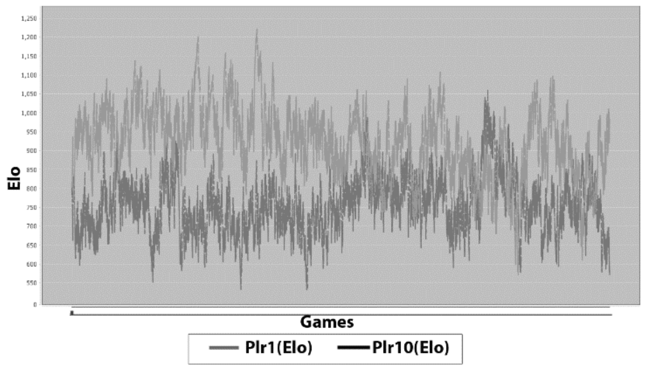


Figure 5. Elo Rating of the evolution (progress) of two agents.

A study of the graph in Figure 5, reveals that none of the agents demonstrates a stable evolution, and this can be readily verified by creating a corresponding chart that presents the evolution of the selected agents based on their *Glicko* ratings.

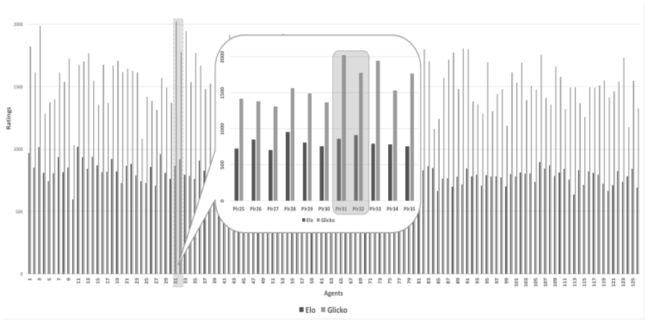


Figure 6. Last ratings of agents, Elo and Glicko.

Figure 6 shows the final Elo and Glicko ratings of all the participants of the tournament in the experiment, which generate the final rankings of the agents. This bar chart is automatically created by the ReSkill system after the completion of the rating process and appears at the Output-tab. It is apparent in this graph that there is a disagreement regarding the ratings provided by the two rating methods. As in previous results comparing Elo and Glicko ratings [7], it turns out that the two rating methods "disagree" in how they rank the agents in most cases, with a "strong disagreement" in many cases. For example, one may easily notice that *Plr31* ranks 1st according to Glicko while being 22nd according to Elo, which looks like a significant disagreement between the two rating methods. On the contrary, *Plr32* ranks 12th and 13th according to the same methods respectively, which is a sign of consistency between the methods in this case. The major issue though is that there are many significant differences in the rankings produced by the two methods, which highlights an inconsistency in applying them on virtual agents and indicates that they should be carefully examined and, maybe, modified before being fully adopted.

5. CONCLUSIONS AND FURTHER WORK

The emergence of multi-agent systems has given rise to a need for benchmarking agents' evolution and progress. It is still a common approach to employ human performance rating systems in multi-agent systems, as reported in the literature. ReSkill was developed to address the lack of open and widely accessible agents' rating systems. It was primarily built to test and assess any rating method and provide comparative results for newly proposed rating methods. ReSkill, at its present version, incorporates the two most widely-used human rating systems (Elo and Glicko). We use standard and open formats in all data processing stages, to maximize data interoperability provide various forms of results and to facilitate assessment and take-up by fellow researchers. ReSkill provides customizable charting functionalities and was built as an open and modular integrated environment, so that future developments maybe easily integrated. Current plans of development include a subsystem to process and integrate user-defined rating methods and provide data analysis and presentation tools (for example, progress graphs), along with the interconnection of ReSkill with existing data mining-analysis tools, and recoding parts of the system to improve cross-platform functionality given the recent (and projected) development on the Java front.

6. REFERENCES

[1] Gilbert, N. and Troitzsch, K. G., 2005. *Simulation for the Social Scientist*. Open University Press, 2nd ed.

[2] Shoham, Y. and Leyton, K. B., 2009. *Multiagent Systems: Algorithmic, Game-Theoretic and Logical Foundations*. Cambridge University Press, New York.

[3] Ferber, J. 1999. *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Addison-Wesley.

[4] Kiourt, C. and Kalles, D. 2012. Social Reinforcement Learning in Game Playing. In *IEEE International Conference on Tools with Artificial Intelligence*, 322-326, Athens, Greece.

[5] Caballero, A., Botia, J., and Gomez-Skarmeta, A. 2011. Using cognitive agents in social simulations. *Engineering Applications of Artificial Intelligence*, 24, 7, 1098-1109.

[6] Marivate, V. N. 2008. Social Learning Methods in Board Game Agents. In *IEEE Symposium Computational Intelligence and Games*, 323-328, Perth, Australia.

[7] Kiourt, C., Kalles, D., and Pavlidis, G. 2015. Human Rating Methods on Multi-Agent Systems. In *13th European Conference on Multi-Agent Systems* Athens, Greece.

[8] Kiourt, C. and Kalles, D. 2015. Learning in Multi Agent Social Environments with Opponent Models. In *13th European Conference on Multi-Agent Systems* Athens, Greece.

[9] Harkness, K. 1967. *Official Chess Handbook*. McKay.

[10] Elo, A. E. 1978. *The Rating of Chess Players, Past and Present*. New York: Arco Publishing.

[11] Logan, Y. and Kagan, T. 2013. Elo Ratings for Structural Credit Assignment in Multiagent Systems. In *Twenty-Seventh AAAI Conference on Artificial Intelligence (Late-Breaking De-velopments)*.

[12] Glickman, E. and Albyn, J. C. 1999. Rating the chess rating system. *Chance,* 12, 2, 21-28.

[13] Herbrich, R., Minka, T., and Graepel, T. 2007. TrueSkill(TM): A Bayesian Skill Rating System. In *in Advances in Neural Information Processing Systems 20,* MIT Press.

[14] Grady, J. O. 1995. *System Engineering Planning and Enterprise Identity*. Taylor & Francis.

[15] Kalles, D. and Kanellopoulos, P. 2001. On Verifying Game Design and Playing Strategies using Reinforcement Learning. In *Proceedings of ACM Symposium on Applied Computing, special track on Artificial Intelligence and Computation Logic* Las Vegas.

[16] Sutton, R. and Barto, A. 1998. *Reinforcement Learning: An Introduction*. MITPress,Cam-bridge, MA.