Entertainment Computing 34 (2020) 100358

Contents lists available at ScienceDirect

Entertainment Computing

journal homepage: www.elsevier.com/locate/entcom

Finding appropriate settings for fairness and engagement in a newly designed game through self-playing AI program: A case study using Japanese crossword game 'MyoGo Renju'

Liu Xinyue[a,*], Fu Luping[a], Mohd Nor Akmal Khalid[a,b,1], Hiroyuki Iida[a,b]

[a] School of Information Science, Japan Advanced Institute of Science and Technology, Nomi, 923-1211 Ishikawa, Japan
[b] Center for Entertainment Science, Japan Advanced Institute of Science and Technology, Nomi 923-1292, Japan

ARTICLE INFO

Keywords:
Crossword puzzle game
Comfortable settings
Game refinement theory
Game design
Fairness

ABSTRACT

This paper explores an innovative way to find the comfortable settings of a newly designed game under development using a computer program. A Japanese crossword game 'MyoGo Renju' has been chosen as a benchmark for this research, whereas some important aspects such as fairness and engagement are evaluated to find the most optimum settings for the player. The game 'MyoGo Renju' can be played by Japanese language learners with competitive purposes as well as educational purposes. An artificial intelligence program of the 'MyoGo Renju' has been developed and various parameters had been evaluated such as board size, word length, bonus point rule, block system, weighted score system, and round mode. The experiment is performed using self-playing Myogo AI where some interesting results have been demonstrated where the 5 × 5 board with the minimum word length of 3-g, 3 number of blocks, and 15 Hiragana characters chosen in a single round is expected to provide the best-expected fairness and engagement to the overall game experiences. The limitations and future works of the research are also discussed.

1. Problem background

With the rise of video games, research on artificial intelligence (AI) in game had flourished. In the early days of AI development, most game AI researchers were trying to make an excellent AI program that beat humans in board games. The reason is that board games contain some essential elements of human-like intelligence. The history of AI experts looking for chess AI can be traced back to the era of Alan Turing and Claude Shannon where they tried to use the Minimax algorithm to play games. As Abramson proposed the Monte-Carlo method and applied in the game of Go in 1990 [1], more and more researchers applied it to board games. Recently, the Monte-Carlo method is applied to not only many kinds of complete information games, but also some non-complete information games.

Nevertheless, the development of the game itself may also be as important as the development of the game AI. The feeling of engagement experienced by the player during the gameplay is an important aspect to consider by game designers [2]. From the perspective of the game designer, a self-playing AI program can be employed to assess the game and possibly improve it. This had proven to be fruitful in designing game content through the procedural content generation (PCG) [3]. The recent development of high-performance game-playing program (such as AlphaZero [4]) had been showed to achieve winning rate beyond what to be considered to be "fair" between two players in chess and chess-like games [5], which raised the concerns for designing and developing a game that maintains the perceived fairness among the players [6].

The history of game design had shown that fairness is an important aspect that determines the survival of any game. A two-player game is perceived to be *fair* if and only if the winning ratio of the players is equal or nearly so statistically [7,8]. Also, the outcome of engaging games should always be uncertain until the last moment of gameplay where the variation in the available options stays constant throughout the games, called the "seesaw" state [9]. This situation sometimes makes one player quickly dominate over the other, which likely causes the available options to be diminished; thus, leading to an uninteresting game.

The goal of this research involves the development and assessment of the Japanese crossword game "MyoGo" which is conducted by applying the Myogo AI program to self-play. The self-play experiment is expected to maximize engagement and maintain the perceived fairness by utilizing the game progress model and the swing model, respectively, from the game refinement theory [10–12]. This is achieved by

* Corresponding author.
E-mail addresses: kinetsu.ryu@jaist.ac.jp (L. Xinyue), s1710271@jaist.ac.jp (F. Luping), akmal@jaist.ac.jp (M.N.A. Khalid), iida@jaist.ac.jp (H. Iida).
[1] Principal corresponding author.

https://doi.org/10.1016/j.entcom.2020.100358
Received 31 May 2019; Received in revised form 5 January 2020; Accepted 15 February 2020
Available online 20 February 2020
1875-9521/

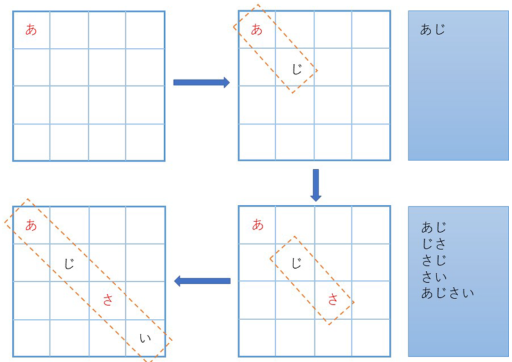

Fig. 1. The game of Myogo Renju.

testing several parameters of the game where the average score reversal (corresponds to the game's engagement) and the player's winning rate (corresponds to the game fairness) through the aforementioned models are observed.

The structure of this research is given as follows: Section 2 introduces the overview and regulations of the Myogo game. The developed AI program based on the Monte-Carlo tree search for the Myogo game is described in Section 3. Then, the assessment methodology of the Myogo game using self-playing AI is given in Section 4. Finally, Section 5 concludes the study.

2. 'MyoGo' and game refinement measure

2.1. Rules of MyoGo

'MyoGo' is a two-player Japanese crossword puzzle that is still under development. Players form words by filling Hiragana characters into the board which scored according to the words they make up (see Fig. 1). The player with the highest scores in the end wins. Because the current stage of the game is still being developed, the basic rules and evaluation methods of the game are introduced. As a starting point of the game design, the game can be played on a 4×4 or 5×5 board. Since the game is a two-player game, black and white are used to represent the player and define black to play first.

The following are the specific rules of the game:

- Players fill a Hiragana character into the board and form words to get scores. The player with the highest score wins and the game ends.
- Blocks will appear randomly in the grids of the board as obstructions, and players cannot fill the Hiragana character into the blocks.
- The score distribution for each word is determined by the number of Hiragana characters the player fills in. Among the words, the more Hiragana characters the player fills in, the more chance the player can get more score.
- Each word has a different rate bonus according to the level of the word, and the higher the level is, the higher the rate is;
- If the Hiragana characters the player filled in are without duplication, the player can receive an additional reward for the final score.

t is worth to note that, compared to the largely popular Scrabble crossword puzzle [13], the game 'Myogo' differs for several reasons (an exception to its language component). For example, Scrabble has some restrictions for players, such as it limits the number of letter tiles players could hold and limits the first player that he has to form the first word through the grid with the star mark on the board. In contrast, Myogo has a high degree of freedom where Myogo game has no restrictions on the number of pieces players hold and the number of times the same Hiragana will be used. Other than that, the rules such as tile block, rate bonus of the level of the word, and non-duplication reward can only be found in the 'Myogo' game.

2.2. Game refinement theory

Game refinement (GR) theory as a mathematical theory of entertainment optimization for game creators was proposed by Iida et al. [10]. Recently, it has been expanded to various games such as score limit games [14], board games [11] and games of social activities [15]. The rationale of using the GR theory to address the perceived fairness and engagement is due to the following reasons. Firstly, the game progress model of the GR theory is characterized by the game element (turn, speed, available option, etc.) over the whole period of the game, which enabled the theoretical quantification of enjoyment. Secondly, the swing model, derived from the game progress model, has provided the medium to observe the game's engagement where the outcome of a game is uncertain due to the change of the player's dominance. Thirdly, the perceived fairness of the game can be validated when the player's winning rate is approximately 50% and the game parameters conform with both models. Therefore, adopting GR theory provided the general "means" of measuring a game capacity for enjoyment, engagement, and fairness.

2.2.1. Game progress model

A general model of GR was proposed based on the concept of game progress and game information progress [12]. The term "game progress" here means two things: one is game speed or scoring rate during gameplay and another one is the game information progress with a focus on the uncertainty of game outcome. Game information progress presents the measure of confidence of the game's result along with period or number of steps. Having full information of the game progress, i.e. after the game ended, the game progress $x(t)$ can be obtained (see Eq. 1) where time t and $x(t)$ satisfies $0 \leqslant t \leqslant t_k$ and $0 \leqslant x(t) \leqslant x(t_k)$, respectively.

$$x(t) = \frac{x(t_k)}{t_k} t \tag{1}$$

Then, to observe the game information progress during the in-game period, it is assumed that game information progress should be exponential because the game outcome is unknown until the very end of the game. Hence, a realistic model of game information progress is given by Eq. 2 where n represents a constant parameter based on the perspective of a game observer. Therefore, the acceleration of the game information progress can be obtained, which is given as Eq. (3). Solving it at $t = t_k$, Eq. (4) is obtained.

$$x(t) = x(t_k)(\frac{t}{t_k})^n \tag{2}$$

$$x''(t) = \frac{x(t_k)}{(t_k)^n} t^{n-2} n(n-1) \tag{3}$$

$$x''(t) = \frac{x(t_k)}{(t_k)^2} n(n-1) \tag{4}$$

The current model assumed that game information progress is translated and transferred in the brains. Although the physics of information in the brain is not yet known, the acceleration of information progress is likely subject to the forces and laws of physics. As such, it is expected that the larger the value $\frac{x(t_k)}{(t_k)^2}$, the more exciting the game becomes, due in part to the uncertainty of the game outcome. Thus, the root square form, $\frac{\sqrt{x(t_k}}{t_k}$ is utilized as the GR measure for the considered game, called *GR* value, whereas denoting $x(t_k)$ and t_k as G and T, respectively (see Eq. (5)).

$$GR = \frac{\sqrt{G}}{T} \tag{5}$$

In the previous works, the game progress model has been applied to various games to verify its effectiveness [10,14,11,12]. The appropriate zone of GR measure are $GR \in [0.07, 0.08]$. The results of GR value for some games are shown in Table 1.

Table 1
Game refinement values of some games adopted from [12].

Games	G	T	GR value
Soccer	2.64	22.00	0.073
Basketball	36.38	82.01	0.073
UNO	0.98	12.68	0.078
Badminton	46.34	79.34	0.086
Table Tennis	54.86	96.47	0.077
DotA	68.60	106.2	0.078

G: successful shoots; T attempts.

2.2.2. Swing Model

In score-based board games such as SCRABBLE, the measure of game progress may differ due to the nature of the turn-based gameplay of the board game. As such, a *GR* measure requires an arbitrary measure to quantify entertainment. As such, a *swing model* was proposed by [12] which defined:

> *Swing* stands for a state transition of advantage during the game progress. Let S and N be the average number of swings and the game length, respectively. The swing is considered as the successful shoot and the game length is the number of attempts, respectively.

Based on this definition, the GR measure in the swing model is given by Eq. (6). In the context of the Myogo game, the S and N corresponds to the player score and the total number of turns, respectively.

$$GR = \frac{\sqrt{S}}{N} \tag{6}$$

3. Monte-Carlo Methods for 'Myogo' AI

The Monte-Carlo method is a stochastic simulation method based on probability and statistics theory that was first proposed by S.M. Ulam and J. von Neumann in the 1940s [16]. With the development of computer technology, the Monte-Carlo method has also been rapidly popularized by the high-speed operation capability of computers. It has broad applications in mathematics, financial engineering, macro-economics, bio-medicine, computational physics, and other fields. The basic idea of the Monte-Carlo method is to establish a probability model for the problem, and then calculate the statistical characteristics of the parameters obtained by observation or sampling test of the model or process, and finally give the solution of the problem.

In many cases, the Monte-Carlo method yields solutions that approximate the optimal one. The larger the sample size is, the closer the value comes to the optimal one. In computer games, the Monte-Carlo method is usually used as an evaluation function to find the action leading to the best average reward from a given state [17]. Simplicity and high-speed are the two major advantages of the Monte-Carlo method.

3.1. Monte-Carlo tree search

The Monte-Carlo tree search (MCTS) is a tree search method, based on the Monte-Carlo method, for finding optimal decisions in a given state by taking random samples in the decision space [18]. Unlike the general tree search method, the MCTS gradually expands each node of the game tree by repeating the randomly simulating events from the initial state. Therefore, its process is a dynamic search process compared to the search method that originally expanded the game tree.

3.1.1. MCTS approach

MCTS has four process steps: selection, expansion, simulation, and back-propagation. As shown in Fig. 2, each node in the figure represents a state in the game process, and each edge represents a child node of the corresponding state that can be obtained by taking an action from the parent node.

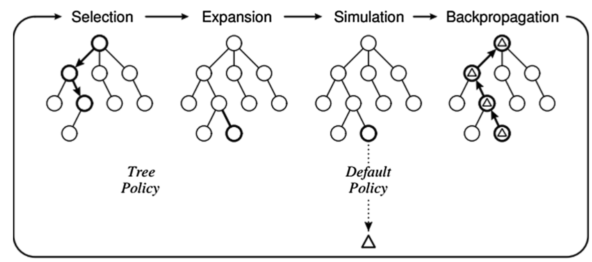

Fig. 2. Steps of MCTS.

The general descriptions of each of the processes of the MCTS are as follows:

- **Selection:** Simulation looks up the leaf nodes starts from the root node of the game tree iteratively, usually based on a selection strategy such as UCB1 formula, until extensible leaf nodes are found.
- **Expansion:** In this step, if all leaf nodes found by the selection step have been accessed, and at least one more node can be extended, then expansion selects an extensible node and add a child node of this leaf node to the game tree.
- **Simulation:** Simulation performs Monte Carlo simulations from the selected leaf nodes to randomly simulate the game until the termination of the game, and estimate the value of the node.
- **Back-propagation:** Back-propagation reversely updates the estimated value of all parent nodes on the simulated path up to the root node based on the results of the simulation step.

3.1.2. Basic algorithm

Algorithm 1 describes the procedures of the MCTS. Firstly, the algorithm sets the current node as the root node and calls the tree policy to search for the leaf nodes that have not been expanded from the root node, then sets the current node as the selected node to expand. Next, it calls the default policy to perform random simulations from the current node, calculates the result of every simulation and updates the value of the parent nodes. Finally, it returns an optimal child node that can be obtained based on the root node of the initial state. So we can get the best action by the optimal child node.

Algorithm 1. Basic MCTS Algorithm

s_0:= initial state node;
v_0:= root node with state s_0;
while *within computational budget* **do**
  $v_l \leftarrow$ TREEPOLICY (v_0);
  $\Delta \leftarrow$ DEFAULTPOLICY $(s(v_l))$;
  BACKUP (v_l, Δ);
end while
return $a \leftarrow$ BESTCHILD (v_0);

3.1.3. The UCT algorithm

In probability theory, a multi-armed bandit problem (sometimes called the N-armed bandit problem) is a problem analogous to a gambler whom at a row of slot machines, deciding which machines to play, number of times and order to play at each machine, and continue to do so at the current machine or different machine [19,20]. The objective is to maximize the sum of rewards earned through a sequence of lever pulls where each machine provides a random reward from a probability distribution specific to that machine. To solve this problem, a trade-off between exploring new machines to get more information (exploration) and choosing the machine with the highest reward to obtain a maximum benefit (exploitation) have to be made. Similar to the multi-armed bandit problem, the UCB1 formula derived by [21] is the first formula that was applied to multi-stage decision-making problems to maintain some balance between the exploration and the exploitation, which is given as in Eq. 7.

$$\bar{v}_i + C\sqrt{\frac{\ln N_i}{n_i}} \tag{7}$$

wherein the context of multi-armed bandit problem, $\bar{v}_i$ stands for the mean payout for machine i, N_i stands for the total number of plays, and n_i stands for the number of plays of machine i. The first component of Eq. (7) corresponds to exploitation and it is high for moves with a high average win ratio. The second component corresponds to exploration and it is high for moves with few simulations. The UCT algorithm is based on the UCB1 formula and applies UCB1 to MCTS. Algorithm 2 shows the procedures of the UCT algorithm.

Algorithm 2. The UCT Algorithm

```
Data: the current state
Result: the best action
s0 := initial state node;
v0 := root node with state s0;
Function UCTSEARCH(s0, v0):
    while within computational budget do
        vl ← TREEPOLICY (v0);
        Δ ← DEFAULTPOLICY (s(vl));
        BACKUP (vl,Δ);
    end while
    return a ← BESTCHILD (v0);
End Function
Function TREEPOLICY(v):
    while v ≠ leaf node do
        if v not fully expanded then
            EXPAND (v);
        end if
        v ← BESTCHILD (v);
    end while
    return v;
End Function
Function EXPAND(v):
    choose a ∉ A(state(v));
    add a new child v′ to v;
    return v′;
End Function
Function DEFAULTPOLICY(s):
    while s is non-terminal do
        choose a ∈ A(s) uniformly at random;
        s ← f(s, a);
    end while
    return reward for state s;
End Function
Function BACKUP(v, Δ):
    while v ≠ NULL do
        N(v) ← N(v) + 1;
        Q(v) ← Q(v) + 1;
        v ← parents of v;
    end while
End Function
Function BESTCHILD(v):
    return
      arg max_{v′ ∈ children of v} Q(v′)/N(v′) + C sqrt(ln N(v) / N(v′));
End Function
```

4. Game assessment

In this section, the MCTS method is implemented for the game AI for MyoGo (denoted as MyoGo AI). Then, data collection was conducted through self-plays experiments by two competing MyoGo AI. Several experiments utilizing the swing model on the parameters of the Myogo game were conducted where the winning percentage, the number of words, and the *GR* values were collected. For each group of experiments, self-plays between two Myogo AI players for 100 games are conducted.

4.1. Parameters of 'Myogo'

- **Board size:** Here two kinds of board sizes are defined, which are 4×4 and 5×5 boards.
- **Minimum word length (n_w):** Limit the minimum length of words players can make. This is because longer word length corresponds to higher player requirements; thus, the game becomes more difficult.
- **The number of blocks (n_b):** Randomly generated blocks can increase the randomness of the game. Hence, it is important to determine the suitable number of blocks relative to board size considered. The two sets of the number of blocks are given in Table 2.
- **Scoring mechanism (s_w):** The game gives players different scores according to the length of the words, the longer the word is, the higher the scores any player can get; thus, inducing challenge to the player for difficult words. The three sets of the scoring mechanism are given in Table 2.
- **Bonus mechanism:** There are two kinds of reward mechanism: word-level bonus (s_l) and non-repetition bonus (s_r). Word level bonus is to give the corresponding rate bonus according to the different levels of words based on the Japanese-Language Proficiency Test N1-N5. The non-repetition bonus is to reward the players with a rate bonus if the filled Hiragana characters are without duplication. The sets of word-level bonuses and non-repetition bonuses are given in Table 2.
- **Round mode (m):** Two modes are considered for Myogo game: the general and the two-hand modes. The former requires players to fill in one Hiragana character in turn, while the latter requires players to fill in one Hiragana character in the first turn, and then they take turns filling in two Hiragana characters until there is no space in the board.

4.2. Experimental results and analysis

Regardless of the board sizes (see Fig. 3), it was observed that the winning percentage of the offensive player is higher than that of the defensive player. Considering the branching factors of players on a 4×4 board, assuming that the total number of Hiragana characters is about 70, then both the offensive and the defensive players have about $16 \times 70 = 1120$ branching factors in the first round. As the search space reduces in the board (due to smaller board size), the branching factors will decrease rapidly. However, the number of choices the offensive player can make is always more than that of a defensive player, which implies that the offensive player has more opportunities to make up words and always at an advantage. Besides, the increase in board size (from 4×4 board to 5×5 board) only manages to further increase the differences between the winning percentage of the players.

The Zermelo's theorem in 1913 indicated that in a limited two-player complete information game, there must be only one of the three situations: the offensive must have the method for a win, or the offensive must have the method for a draw, or the defensive must have the method for a win [22]. Many board games, such as chess and gobang, have been proved by mathematical induction that the offensive must be invincible. Therefore, it is believed that MyoGo also has the characteristic that the offensive has the advantage.

Due to several numbers of the possible combinations of the Myogo

Table 2
Parameters for scoring mechanism, word-level bonus, non-repetition bonus, and number of blocks.

Set No.	Word Length				Word Level			Non-repetition Bonus	No. of Blocks (4 × 4 board)	No. of Blocks (5 × 5 board)
	2	3	4	5	N1	N2	N3-N5			
1	2	3	4	5	1.2	1.1	1.0	×1.5	2	3
2	2	5	12	20	1.5	1.2	1.0	×2.0	4	5
3	2	8	20	40	1.5	1.25	1.0			

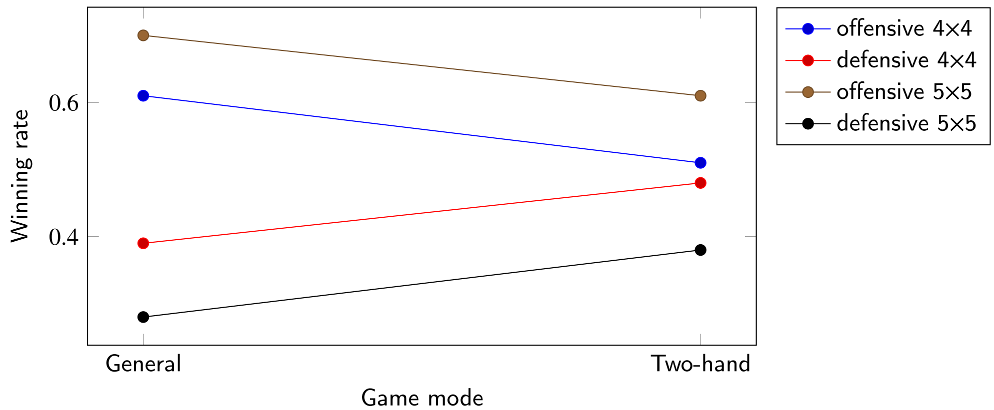

Fig. 3. Winning rate of two modes in the 4 × 4 board and 5 × 5 board.

AI parameters, a methodical approach is needed for testing it to determine the most viable parameters that greatly affect the game. Hence, the general mode is compared with the two-hand mode as the starting point on the 4 × 4 board concerning the winning rate, the number of words, and the *GR* value. Table 3 lists the results of self-play experiments with different parameter combinations of the game. The general mode represents the original game which sets the minimum word length to be 2-g ($n_w = 2$). Considering the state transitions of the player's advantage during game progress, players can fill in two Hiragana characters each time in the two-hand mode, so they have much more chance to reverse the score. According to the simulation results, there will be an average 3.48 reversals (corresponds to the average swing) in each game of the general mode, and 3.24 reversals in each game of the two-hand mode.

Moreover, it was found that the scoring mechanism with bigger gaps among the scores based on different lengths of words does not weaken the advantage of the offensive and play a role in balancing the game. However, the bonus mechanism can narrow the gap between the offensive and the defensive player; thus, adjusting the perceived fairness of the game. As such, the set 2 of the word level bonus (N1 = 1.5, N2 = 1.2, and N3-N5 = 1.0) and the non-repetition bonus (×2.0) is considered relatively ideal for the game on 4 × 4 board.

Nevertheless, the 4 × 4 board is considered to be small causing complex word-formation to be difficult for the players. In the case of Myogo, enlarging the board size would increase the branching factors by about 70 (the size of the Hiragana table of the game is about 70) for each additional grid on the board. Therefore, expanding the board size would cause a dramatic increase in the number of branching factors. As such, while it is reasonable to expand the board to increase the length of the game, it is still important to reduce the branching factors. This is addressed by adopting the minimum word length to be 3-gram ($n_w = 3$).

Table 4 lists the results of self-plays with different parameters on 5 × 5 board. From the results, higher bonuses increase the average of swings, which means that the bonus mechanism could make players have more chances to reverse the score and make the game more interesting. Also, the longer the minimum word length, the fewer available branching factors the players have, and also the lower the average of swings and GR values. It is worth mentioning that, from the total

Table 3
Results of self-plays experiments with different parameters on 4 × 4 board with minimum word length of 2-g.

AI parameters	Win rate (Black)	Win rate (White)	Number of words	GR value	Average Swing
(1, 1, (1, 1), 1)	0.61	0.39	2367	0.133	3.48
(1, 1, (1, 1), 2)	0.51	0.48	2342	0.129	3.24
(2, 1, (1, 1), 1)	0.63	0.48	1519	0.135	2.62
(1, 1, (2, 1), 1)	0.58	0.42	2221	0.127	3.15
(1, 1, (3, 1), 1)	0.61	0.37	2157	0.125	3.08
(1, 2, (1, 1), 1)	0.60	0.40	2191	0.120	2.83
(1, 3, (1, 1), 1)	0.64	0.36	2122	0.128	3.20
(1, 1, (1, 2), 1)	0.57	0.43	2140	0.129	3.24
(1, 1, (2, 2), 1)	0.55	0.45	2138	0.124	2.99
(1, 1, (3, 2), 1)	0.64	0.34	2101	0.124	2.99
(1, 2, (2, 2), 1)	0.66	0.34	2110	0.132	3.40
(1, 1, (2, 2), 2)	0.48	0.52	2152	0.132	3.43
(1, 1, (3, 2), 2)	0.46	0.53	2140	0.128	3.20

(☆) number correspond to (n_b, s_w, (s_l, n_r), m).
n_b: no of blocks; s_w: scoring mechanism; s_l: word level bonus.
n_r: non-repetition bonus; m: game mode.

Table 4
Results of self-play experiments with different parameters on 5 × 5 board minimum word length of 3-g.

AI parameters	Win rate (Black)	Win rate (White)	Number of words	GR value	Average Swing
((1, 1), 1)	0.70	0.28	994	0.079	3.02
((1, 1), 2)	0.61	0.38	1154	0.080	3.09
((2, 2), 1)	0.64	0.36	1081	0.082	3.24
((3, 2), 1)	0.54	0.46	1102	0.084	3.39
((2, 2), 2)	0.56	0.43	1148	0.083	3.25
((3, 2), 2)	0.55	0.44	1180	0.084	3.40

(☆) number correspond to ((s_l, n_r), m).
s_l: word level bonus; n_r: non-repetition bonus; m: game mode.

Table 5
GR value with various game lengths with respect to the lower swing (2.8), middle swing (3.2), and upper swing (3.5) bounds.

Game length	*GR* value		
	Swing = 2.8	Swing = 3.2	Swing = 3.5
16	0.1046	0.1118	0.1169
20	0.0837	0.0894	0.0935
25	**0.0669**	**0.0716**	**0.0748**
30	0.0558	0.0596	0.0624
36	0.0465	0.0497	0.0520

number of words composed, it can be seen that limiting the minimum word length increases the difficulty of the game.

Although the number of blocks and the scoring mechanism do not affect the expected fairness of the game, these parameters are not considered in the 5×5 board anymore. This consideration seems reasonable since the GR values of games on the 5×5 board are closer to the sophistication zone $GR \in [0.07, 0.08]$. This implies that the 5×5 board provides sufficient sophistication for both novice and advanced players, while present enough uncertainty of the outcome that makes it entertaining and interesting to play. Therefore, the minimum word length of 3-gram and the number of blocks of 3 on the 5×5 board may be appropriately oriented for mid-level and advanced players. Thus, the 5×5 board is more reasonable as the conventional Myogo game setting.

Furthermore, from the perspective of balancing the Myogo game for both novice and advanced players, the experimental results show that other elements of the game with the same board size have little effect on the GR value. However, it is interesting to observe that the game length affects the balance of the Myogo game. As such, the swing model is applied to the various game swing measures (corresponds to the lower bound and upper bounds of the average swing) for several fix game lengths to determine the influences of the game length. Based on Table 5, it can be observed that the length of the game has a significant influence on the GR value. According to the GR value, an appropriate game length that provides the best balance of the Myogo game is about 25.

Utilizing the game refinement theory, the best game setting had been obtained where the 5×5 board with the minimum word length to 3-g, 3 number of blocks, and 15 Hiragana characters chosen in a single round by the player, provides the best-perceived fairness and entertainment experience to the game. Besides, longer game length can make the game more competitive, high-rate bonuses and two-hand mode can make the game more interesting, and limiting the number of Hiragana characters a player can choose in each round increases the complexity of the game.

While the best game setting obtained is based on data collected only from the AI players, such a setting provides "educated" guess to the expected perception of an actual human player in terms of the amount of enjoyment and attractiveness. Also, based on the game refinement theory, the appropriate setting of the Myogo game had been found which balances the necessary skill and the expected chance within each gameplay. As such, either a novice or advanced human player could expect to similarly be "entertained" with the found best game setting.

5. Conclusion

In this study, the game program and the game AI of the Japanese crossword game, MyoGo, was implemented. The self-play game data of the Myogo AI was collected to evaluate and optimize the game parameters. For optimizing the balance of the game, we add blocks to the board to increase game randomness, and add bonus mechanisms to increase counter-attack opportunities, and limit the minimum word length to provide game skill and depth for both novice and advanced players.

Although a proper game setting was achieved through the experiments conducted from the present study, there is still a lot of works to be done. Firstly, several sets of parameters for each element of the game was assumed to observe the trend of the game results under different settings. However, these parameters are not necessarily reasonable. For example, the parameters of the game scoring mechanism may be unreasonable for games on a 4×4 board, because even if players can get higher scores based on the length of words, it is difficult for them to form longer or complicated words due to the board size. Future work needs to modify the parameters through previous experimental feedback to make the parameters more reasonable.

Besides, there are about 216 different combinations of parameters that have been assumed, where only a little part of it was tested. Although some settings have been proven to be unsuitable, continue experimentation with them using different combinations of parameters may be reasonable for a complete and comprehensive experimental study.

The game was evaluated with self-play AI data and a theoretical model, which lacks the real feedback of human players. Potential future directions that collect actual players' (both novice and advanced players) feedback on the game experience and evaluate the game effects on other perspectives of the game (such as the educational potential of playing Myogo) through long-term observation may be worthwhile endeavors.

Finally, the application of the game refinement theory as a new tool for determining the game design sophistication and evaluating the perceived game fairness among the players, adopted alongside the self-playing AI, may provide a comprehensive framework for the game designer in the future.

Declaration of Competing Interest

None.

Acknowledgement

This research is funded by a grant from the Japan Society for the Promotion of Science, within the framework of the Grant-in-Aid for Challenging Exploratory Research.

References

[1] B. Abramson, Expected-outcome: a general model of static evaluation, IEEE Trans. Pattern Anal. Mach. Intell. 12 (1990) 182–193.
[2] S. Xiong, L. Zuo, R. Chiewvanichakorn, H. Iida, Quantifying engagement of various games, in: The 19th Game Programming Workshop 2014, Information Processing Society of Japan, 2014.
[3] M. Guzdial, M. Riedl, Automated game design via conceptual expansion, in: Fourteenth Artificial Intelligence and Interactive Digital Entertainment Conference, 2018.
[4] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, et al., A general reinforcement learning algorithm that masters chess, shogi, and go through self-play, Science 362 (2018) 1140–1144.
[5] H.P.P. Aung, M.N.A. Khalid, H. Iida, Towards a fairness solution of scrabble with komi system, in: 2019 International Conference on Advanced Information Technologies (ICAIT), IEEE, 2019, pp. 66–71.
[6] H. Iida, Fairness, judges and thrill in games, Japan Advanced Institute of Science and Technology, Tech. Rep., 28, 2008.
[7] H.J. Van Den Herik, J.W. Uiterwijk, J. Van Rijswijck, Games solved: now and in the future, Artif. Intell. 134 (2002) 277–311.
[8] H. Iida, On games and fairness, in: 12th Game Programming Workshop in Japan, 2007, pp. 17–22.
[9] H. Iida, N. Takeshita, J. Yoshimura, A metric for entertainment of boardgames: its implication for evolution of chess variants, Entertainment Computing, Springer, 2003, pp. 65–72.
[10] H. Iida, K. Takahara, J. Nagashima, Y. Kajihara, T. Hashimoto, An application of game-refinement theory to mah jong, International Conference on Entertainment Computing, Springer, 2004, pp. 333–338.
[11] J. Takeuchi, R. Ramadan, H. Iida, Game refinement theory and its application to volleyball, Research Report 2014-GI-31 (3), Information Processing Society of Japan (2014) 1–6.

[12] K. Suwanviwatana, H. Iida, First results from using game refinement measure and learning coefficient in scrabble, arXiv preprint arXiv:1711.03580 (2017).
[13] J. Edley, J. Williams, Everything Scrabble, Gallery Books, 2018.
[14] N. Nossal, H. Iida, Game refinement theory and its application to score limit games, in: Games Media Entertainment (GEM), 2014 IEEE, IEEE, 2014, pp. 1–3.
[15] L. Zuo, S. Xiong, H. Iida, An analysis of hotel loyalty program with a focus on the tiers and points system, Systems and Informatics (ICSAI), 2017 4th International Conference on, IEEE, 2017, pp. 507–512.
[16] S.M. Ulam, Von Neumann: The Interaction of Mathematics and Computing, Birkhäuser Boston, Boston, MA, 1986, pp. 215–222.
[17] G. Chaslot, J.-T. Saito, B. Bouzy, J. Uiterwijk, H.J. Van Den Herik, Monte-carlo strategies for computer go, Proceedings of the 18th BeNeLux Conference on Artificial Intelligence, Namur, Belgium, 2006, pp. 83–91.
[18] C.B. Browne, E. Powley, D. Whitehouse, S.M. Lucas, P.I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, S. Colton, A survey of monte carlo tree search methods, IEEE Trans. Comput. Intell. AI Games 4 (2012) 1–43.
[19] D.A. Berry, B. Fristedt, Bandit Problems: Sequential Allocation of Experiments (Monographs on Statistics and Applied Probability) vol. 5, Chapman and Hall, London, 1985, pp. 71–87.
[20] J. Gittins, K. Glazebrook, R. Weber, Multi-armed Bandit Allocation Indices, John Wiley & Sons, 2011.
[21] P. Auer, N. Cesa-Bianchi, P. Fischer, Finite-time analysis of the multiarmed bandit problem, Mach. Learn. 47 (2002) 235–256.
[22] U. Schwalbe, P. Walker, Zermelo and the early history of game theory, Games Econ. Behav. 34 (2001) 123–137.