# Biased Random Sequence Generation for Making Common Player Believe it Unbiased

Sila Temsiririrkkul, Hisamitsu Nomura, Kokolo Ikeda
School of Information Science
Japan Advance Institute of Science and Technology
Ishikawa, Japan 923–1211
Email: temsiririrkkul@jaist.ac.jp, s1210043@jaist.ac.jp, kokolo@jaist.ac.jp

***Abstract*—The players' dissatisfaction in randomness due to their biases is the major problem of game's development. Many games were complained even though the developper selected a good algorithm to generate the randomness. The aim of this study was to reduce negative feedback from player in randomness due to cognitive biases of players. By analyzed the randomness in players' perspective, it was demonstrated that the idealistic randomness is different from the randomness in players perspective. According to the result, the characteristic feature of randomness were defined. Incoperated into pseudorandom algorithm, the natural randomness for players were generated by spacial method. To assure the capability of method and sequences, the evaluation of naturalness were done in the game of "sugoroku". As a result, the natural sequences is natural for player and is able to reduce the dissatisfaction of players in pratical uses.**

## I. Introduction

The generation of pseudorandom numbers is a well-known problem which aims to create random sequences with regards to "distribution quality", "length of sequence period", and "speed of generation" [1]. The pseudorandom generators are used in many processes such as stochastic optimization, Monte-Carlo method [2], and reinforcement learning. In recent years Mersenne twister (MT) has interested researchers and developers due to its ability to produce good quality random sequences. It has been greatly improved in terms of mathematical requirements mentioned above when compared to the old generation algorithms such as linear congruential [3]. In mathematical terms, it is close to an ideal random number generator.

In computer games, pseudorandom algorithms have been used to generate randomness such as in the rolling virtual dice in board games and in Trump-shuffling and drawing in card games(fig.1 (1)). In these game, all the processes are completely controlled by the program. Thus it is easy to modify or control the randomness (or to cheat players). In fact, many old generation games used this advantage to eases the weakness of computer players. Because of resources limitation, it was hard to develop smart computer players. The arbitrary modification of random sequences such as this has often been noticed (fig.1 (2)). This affected to the feedback of the game directly, and it might affected to doubt of other games.

In some cases, players accepted the modification of randomness (fig.1 (3)). For example, in some computer board games (e.g. "Momotaro dentetsu"[1]), randomness is adjusted by the system, due to the story line or characteristic of character in the games. Such as the charactor's ability force the dice to be only appears as "5" or "6". In such cases, players could accept the modified randomness because it is explicitly shown the intention.

In earlier cases, the satisfaction of players conform to the modification of randomness. However there are some exception (fig.1 (4)). "Culdcept" [4], a Japanese small computer board game series are very popular even in Europe. However, players complained that the random sequences in the game appeared to be modified in order to disadvantage players. Even though the developer declared that there was no modification in this game, many players still weren't convinced [5]. Another game, "Dungeons & Dragons online" [6] also faced to the same credibility problem. Many users complained that the dice roll generally gave low values.

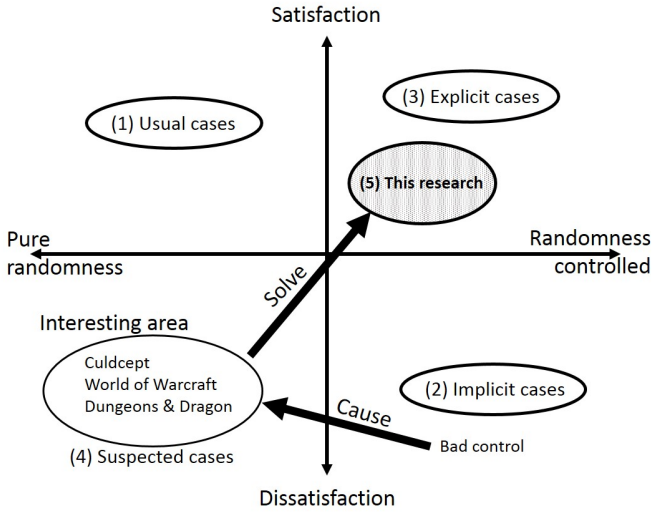


Fig. 1. Relationship between the uses of random and satisfaction of player.

In such cases, the randomness generated by a good algorithm might be the best way to deal with problems. However, the algorithm which gives a good quality of randomness does not necessarily guarantee players satisfaction. Even though the

[1]"Momotaro dentetsu" is a famous Japanese computer board game. The game is based on "Sugoroku" Japanese classical board game that similar to "Monopoly"

developers used the new generation algorithm such as MT, complaints about the randomness still persisted.

Putting into conduction the hypothesis that the ideal randomness in mathematic might not look natural to common players. the aim of this research is to decrease the dissatisfaction of player due to the pseudo-randomness in games by imitating the characteristic of randomness in players' belief by incorporating with a pseudorandom algorithm (fig.1 (5)).

In this research, common players' perceptions, misunderstandings, beliefs, and bias of randomness were analyzed. As the result of analyzing, the different between players' beleived randomness and theoretical randomness were demonstrated. Therefore, a special method to generate natural random sequences from the viewpoint of common players by modifying pseudorandom sequence is presented. To confirm the capability of method, the evaluation were done in the game of "Sugoroku". As a conclusion, it is shown that natural random sequences are able decrease the dissatisfaction of player in practical uses.

## II. Related works

Generally, randomness has been used in many computer processes or method. To generate randomness, pseudorandom algorithms have been used. Many algorithms had been proposed in order to produce a quality of randomness. Among them, the Mersanne twister has demonstrated very good performance. It is able to produce uniformly distributed sequence in 623 dimensions with an enormous size of period of up to $2^{19937}$ [3]. It can be said that this algortihm is very close to a theoretical randomness, and as such it is sufficient for the usual purposes.

In the game industry, pseudorandoms have been used widely. For example, in the popular game series the Pokemon , a pseudorandom generator (PRG) is used in many processes such as in the "monster egg hatch" [7]. Another example can be found in the game "World of Warcraft". The system called "roll" which controls looting items in this games, uses a pseudorandom algorithm called IBBA [8]. In these two games, the quality of randomness apparently had a significant effect on the players satisfaction. High quality, accurate random sequences are required in order to make the game unpredictable and thus entertaining to players. For example, in card games like poker and blackjack, drawing cards should be based on good randomness in order to be fair to the players. If the drawn cards often appear to be too bad or frequently appear to be good, the entertainment quality of the game will suffer.

However, the judgment of randomness (or non-randomness) depends on players' perception. Normally, human often predicts some events which are random, by referring to previous patterns and trends. Such as, it is very natural that we predict tomorrow weather by referring yesterday and todays weathers. Such behaviors strongly affect their randomness believability, and they might misperceive real probability and make the judgment in randomness deviated too. As the example in section I, stated how the Culdcept series has recieved strong complaints about the randomness of dice used in the games, even though the developers have stated that the sequences in the game were not being modified. World of Warcraft, has also faced the same problem. The roll system which random the priority number for looting item in the game, seem to be fair for every player. However, a number of players are still unsatisfied. This misperceptions which deviate judgments is called cognitive biases [9].

Previous literature also showed that the randomness from human perspective is different from the theoretical definition. In 1960, Bakan performed an experiment with 70 undergraduate students who were requested to simulate 300 coin tosses. The experiment proposed to analyze the misunderstanding of the frequency of the change from H to T or T to H (Head / Tail). As the result, subject showed more than 176 times by average, however the theoretical value is only 150 times[10]. Schilling had shown the longest possible consecutive appear of heads or tails such as HTTTTHT or TTHHHHHT. Theoretically for 200 tosses, the consecutively run of head or tail might possible to be 7-10 times. However, in human randomness perception, the consecutive runs are less than 5 times that is much balancing of frequencies. It shows the difference between human beliefs and actual randomness. [11] [12].

According to previous studies, the use of randomness which is similar to real randomness, might not satisfy players but adjustment of the players' perceiving is hard. Thus, the generation of "natural" random sequences was proposed to solve such problem.

There are some patents that have been proposed by game companies to ease such players' complaints. For example, in lottery games among two or more players, both "too lucky" cases and "too unlucky" cases are avoided by controlling the lottery result, in other words, bad results are given to lucky players, a good results are given to unlucky players [13].

In this recent research, the solution to generate believable randomness for most game players was demonstrated. To decrease players' negative feeling, pseudorandom sequences were modified in order to make them similar to the players' rendomness perception. In other words, this methods imitated players' cognitive biases.

## III. Research approaches

The approach was divided into 4 steps. (1) Analysis of players' cognitive biases, (2) generation of believable random sequences by imitating players' biases, (3) evaluation of generated random sequences in the game of Sugoroku, (4) individual randomness generation. The process were shown in fig.2.

1) Preliminary questionnaires: trends of misunderstanding, bias, and belief in random numbers and its' distribution in common players were surveyed by preliminary questionnaires. (as shown in section 4).
2) Generation of natural random sequences: a method for generating intended sequences was proposed. According to the trend of cognitive bias retrieved in step (1), sequences were optimized to make usual players "feel" them unbiased. This method was evaluated through simple emotion experiment using human subjects.
3) Sugoroku: a very simple board game using dice is employed. In game, not only the sequence of dice numbers, but also the sequence of "whether

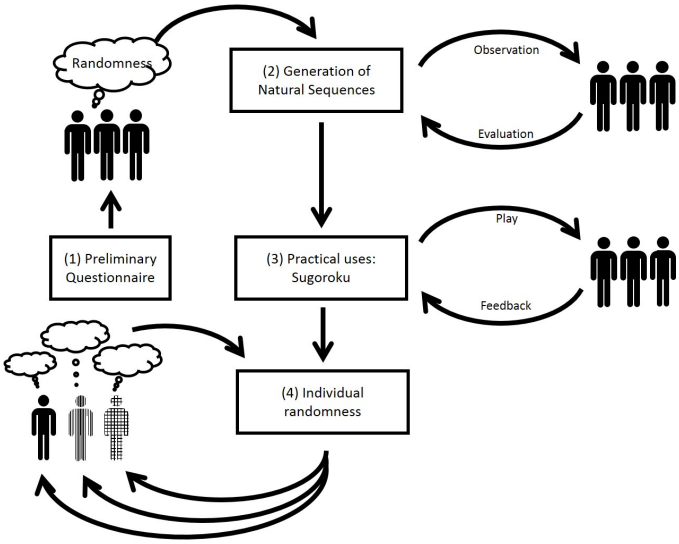


Fig. 2. research approach

being trapped in undesirable cells" must be controlled to decrease complaints. For example, assuming the possibility of being trapped is 1/3, a player will get angry if trapped 3 times in a row, or a player will be bored if not trapped 6 times in a row. A method for controlling both sequences is proposed.

4) Individual randomness: the previous steps are the generation of natural sequences for average players. However, our method based on common players' biases, it might be unnatural for experienced players or players who have mathematical knowledge. The online approach can be proposed in the near future to solve this problem, by analyzing players' trend of biases online and then generating random sequences for each player.

The first three approaches were done and are presented in this paper in chapter IV to VII.

## IV. Preliminary Questionnaires

Humans have different biases of randomness. One of our goal is to generate believable random sequences for common players. Preliminary questionnaires were conducted in order to identify common biases, misunderstanding and belief in randomness of players. At first, the human biases were investigated to create the criteria for random sequences. Hundred digit length sequences of numbers in the range [1-6] were simulated by subject players (not using real dice) as shown in fig 3. The first 40 digits of 2 sample sequences simulated by subject players are:

- 4525143326144641355542665654121422351611
- 1523645326413253412156362436152342615243

The first sequence frequently showed consecutive part (such as 33, 44, 555) but no such part was found the second sequence that is far from theoretical randomness. However, it was assumed that generating random sequences without repeating might be look "natural" in this subject's view. Though this is an extreme case, it has been found that almost all subjects tend to avoid such repeatitions.

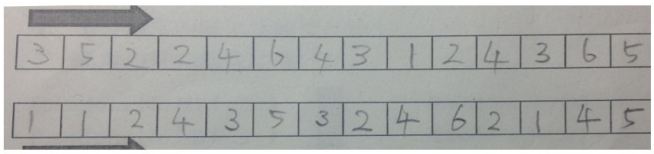


Fig. 3. Sample of manual generated sequence

### A. Random sequences' bias feature

As the criterias for analyzing sequences, fifteen characteristic pattern features were defined. Regarding these features, theoretical random sequences and subjects' artificial random sequences were analyzed and compared. Fifteen features and theoretical sequences' values (for a hundred digits numerical sequences) are captured as follows.

| | | |
|---|---|---|
| F1 | $\chi^2$ distribution of sequence. | (5.0) |
| F2-F5 | $\chi^2$ distribution of about quarter length subsequences; 1st -30th, 24th -53rd, 48th -77th ,70th -99th | (5.0) |
| F6 | The frequency of the flips from even to odd (or vice versa). | (49.5) |
| F7 | The frequency of the same numbers appearing consecutively 2 times. | (16.5) |
| F8 | The frequency of the same numbers appearing consecutively 3 times. | (2.7) |
| F9 | The frequency of the same numbers appearing consecutively 4 times. | (0.45) |
| F10 | XXYY, XYXY, XYYX (Two pairs); the frequency of four consecutive numbers which comprise two kinds of numbers. | (6.7) |
| F11 | XXYYY (Full house); The frequency of a pair of the same numbers and a three of the same numbers occur consecutively | (1.5) |
| F12 | XYXX, The frequency of subsequences in which 3 of 4 digits are the same numbers. | (4.5) |
| F13 | XYXZX, The frequency of subsequences in which 3 of 5 digits are the same numbers. | (5.6) |
| F14 | XYXZXX, The frequency of subsequences in which 4 of 6 digits are the same numbers. | (1.8) |
| F15 | XXYXZWX, The frequency of subsequences in which 4 of 7 digits are the same numbers. | (2.5) |

(W, X, Y, Z represent the number which appear in subsequence)

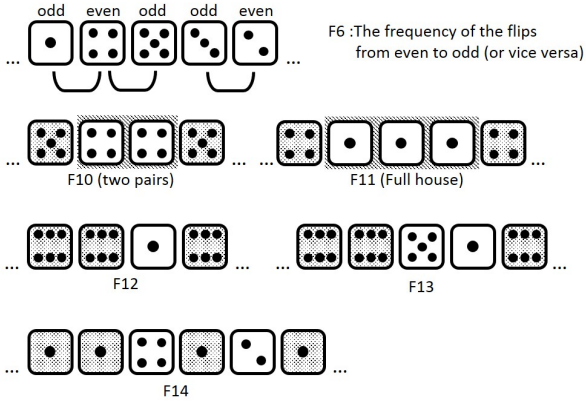


Fig. 4. Example of Feature F6,F10 - F14

### B. Natural sequences' features

To identify common cognitive biases about the naturalness of random sequences, artificial sequences from sixteen subject players' were analyzed. The summary of the analysis is shown in Table 1 (the result of F3-F5 resemble to F2).

| Feature | Theoretical value | Average | Average of Top 8 values | Average of bottom 8 values |
|---|---|---|---|---|
| F1 | 5.0 | 2.4 | 3.3 | 1.5 |
| F2(-F5) | 5.0 | 2.2 | 3.0 | 1.3 |
| F6 | 49.5 | 55.6 | 62.0 | 49.3 |
| F7 | 16.5 | 10.9 | 15.9 | 6.0 |
| F8 | 2.7 | 0.8 | 1.5 | 0.1 |
| F9 | 0.45 | 0.06 | 0.1 | 0.0 |
| F10 | 6.7 | 3.3 | 5.3 | 1.3 |
| F11 | 1.5 | 0.4 | 0.8 | 0.0 |
| F12 | 4.5 | 1.4 | 2.8 | 0.0 |
| F13 | 5.6 | 1.8 | 2.8 | 0.9 |
| F14 | 1.8 | 0.3 | 0.5 | 0.0 |
| F15 | 2.5 | 0.5 | 1.0 | 0.0 |

TABLE I. FEATURES OF SUBJECT PLAYERS' SEQUENCES

F1 and F2 show the distribution of numbers in the sequence. The average value of artificial sequences was a bit lower than the theoretical values. It is interprete that players prefered **evenly appears of each of the digit**.

F6 is revealed that the change from even to odd or vice versa were often shown. These changes sound to be "Random" in subjects' view. This result conformed the experiment by Bakan (1960).

F7 to F9 represent the consecutive occurrence of the same number. The practical values are significantly lower than the theoretical values. After the generation experiment, subjects were requested to answer the following question (How many times should 3 consecutive same digits appear?). More than half of the subjects' answers are close to theoretical values. However, players tend to feel that scatter numbers are random, thus the appearance of the same number as previous was avoided.

F10 to F15 represent the frequency of pattern appearance. It is significantly different from the theoretical values. The same trend as features of F7 to F9 appeared. The average of top 8 values are far from theoretical values. It is shown that subject try to avoid the pattern with or without intention.

Regarding the random sequences simulated by subject players compared to the theoretical values, the significant differences is necessary for the generation of "natural" random sequences. However The difference of personal belief which shown in the difference between average of Top 8 values and average of bottom 8 values was important too.

## V. NATURAL RANDOM NUMBER GENERATION: HOW TO MEASURE THE NATURALNESS AND HOW TO OPTIMIZE

According to the result showing in section 4 and the hypothisis, an artificial sequence might look natural when it show the same or similar features as generated by subject. A method for generating random sequences which make common players feel "natural" or "unbiased" was proposed. The symbols are defined as follow.

- $s$ : pseudorandom sequences
- $f_i(s)$ : statistic value of sequence $s$ about i-th features
- $[\alpha_i, \beta_i]$ : favorable range of feature $f_i$
- $err_i$ : the amount of deviation from range when

$$err_i(x) = \begin{cases} \alpha_i - x & \text{if } x < \alpha_i \\ x - \beta_i & \text{if } \beta_i < x \\ 0 & \text{if } \alpha_i < x < \beta_i \end{cases}$$

- $\gamma_i$ : weight of deviation

For sequence S, the following equation was defined.

$$err(s) = \sum_i (\gamma_i err_i(f_i(s)))$$

The values of err(s) is minimized by using the algorithm, and err(s)=0 only when all feature values are in the favorable ranges.

### A. Example of $err(s)$

The length of random sequences in the experiment is fixed to 50 digits. Regarding the result of the first experiment $\alpha, \beta, \gamma$ were defined and shown in table II.

| Feature | Theoretical value | Lower bound $\alpha$ | Upper bound $\beta$ | weight $\gamma$ |
|---|---|---|---|---|
| F1 | 5.0 | 2 | 5 | 3.0 |
| F2(-F5) | 5.0 | 2 | 5 | 3.0 |
| F6 | 24.7 | 27 | 30 | 1.0 |
| F7 | 8.3 | 5 | 8 | 1.0 |
| F8 | 1.3 | 0 | 1 | 3.0 |
| F9 | 0.2 | 0 | 0 | 10.0 |
| F10 | 3.4 | 1 | 3 | 4.0 |
| F11 | 0.9 | 0 | 0 | 4.0 |
| F12 | 2.2 | 0 | 1 | 4.0 |
| F13 | 2.8 | 1 | 2 | 4.0 |
| F14 | 0.9 | 0 | 0 | 4.0 |
| F15 | 1.2 | 0 | 0 | 4.0 |

TABLE II. EVALUATION CRITERIA

300 standard pseudorandom sequences were prepared by using pseudorandom generator. The best sequence gave $err()$ as 2.7 and the worst gave $err()$ as 239.0. The average $err()$ is 52.1. The best sequence and the worst sequence are shown as follows.

- The best Sequence(2.7) : 52166232635543316563431216615323356642653154342315
- The worst sequence(239.0) : 31443255533554554656246445591543564454542566656544

According to table II, the practical values of all features except F6, were lower than theoretical values, such as F11, "Full house" was prohibited even though the theoretical average is 0.9 (not extraordinary). Thus the best sequences which

values of feature F7 to F15 are very small, supposed to look natural in players‘ view. In the other hand, the worst sequences held many groups of consecutive same numbers. That the pseudorandom generator might be able to give such “natural” and “unnatural” output is demonstrated.

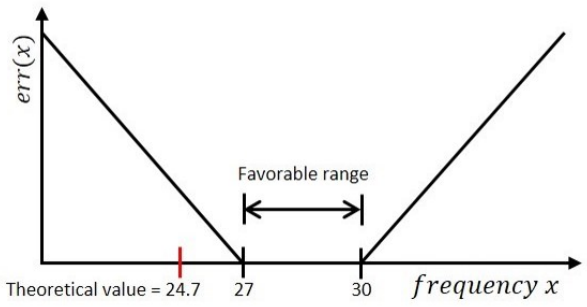


Fig. 5. The example: $err_{F6}(x)$ :The frequency of the flip from even to odd or vice-versa.

### B. Optimization algorithm

To find the sequences with the smaller $err(s)$, the local search algorithm was employed, the summary of the algorithm is:

1) Initialize 50 digits sequence $s$ by standard pseudo-random algorithm.
2) $s'$ is generated by changing a randomly selected digit of $s$
3) If $err(s') < err(s)$then assign $s'$ to $s$.
4) Repeat (2) and (3) until $err(s)$ is becoming 0 or process over 1000 times.

In preliminary design, simulated annealing and local search were proposed. However, the setting at this time was the local search which could be able to give the $err(s) = 0$ as well. And the time spent in optimization was around 0.02 second, exhibiting the acceptable performance with a standard PC and good enough to be used in usual video games. The simulated annealing could be necessary if the number of features or limit upper or lower bound was increased.

## VI. Naturalness Evaluation

To confirm that the obtained optimized pseudorandom number sequences with $err(s) = 0$ look “natural” for common players, the subject experiment was performed.

### A. Evaluation preparation

First, sets of sequence, three types of pseudorandom sequences were prepared as follows.

1) Standard random sequences [Standard]: sequences generated by a pseudorandom algorithm without modification, *Random* build in C#.net.
2) Low Rank Random Sequences [Low rank]: 60 bad sequences from the last 20% of 300 pseudorandom sequences were sorted by the value of $err()$
3) Optimized Random Number [Optimized]: Optimized pseudorandom sequences, which $err(s) = 0$.

48 sequences were prepared for each [Standard] [Low rank] and [Optimized].

### B. Evaluation method

Sixteen subjects were employed to evaluate the naturalness of each [Standard],[Low rank], and [Optimized] Sequences. Each subjects received nine experimental programs (based on Microsoft Windows Operating system, see fig. 2) which display random sequences. One new number would appears every second, and last six number were displayed. Subjects ran an experimental program twice per program (the same sequences) and give a score for the program. The definitions of score are:

1) This Random Sequence is obviously have a Bias.
2) This Random Sequence might have a Bias.
3) Don’t know.
4) This Random Sequence might be a standard and natural random sequence.
5) This Random Sequence is obviously pseudorandom sequence.

The set of nine programs contained three of each [Standard], [Low rank], [Optimized] sequences which were prepared in VI.A, then $3 \times 16 = 48$ sequences were generated per each type. The order of 9 programs were shuffled, and the group being shown was unknown to subjects.

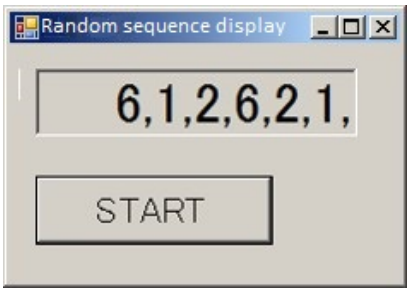


Fig. 6. Screen capture of Random sequences display program.

### C. Evaluated result

The result of the experiment, classified by groups of sequences, were summarized in table III. Please note that the exhibited sequences were given by a standard pseudorandom generator.

[Low rank] were the worst 20% of 300 sequences ordered by value of *err()*. According to values of *err()*, these sequences are far different from randomness in players’ perception. So the feelings of players “this sequence is biased!” was ordinary.

[Optimized] got the best evaluation among 3 groups. Over 50% of the sequences got 4 to 5 score in the experiment. It was concluded that sequences in this group look natural (look like real random sequences) from the players’ point of view. In other words, our *biased* sequences successfully made the players believe that the sequences are *unbiased*.

[Standard] got a better result than [Low rank] but lower than [Optimized]. Because [Standard] provided sequences which have $err()$ from 2.7 to 238.0. This might be interpreted

as “some sequences look natural whereas some sequences did not”.

The experiment exhibited that our designed $err()$ can be used to evaluate and simulate the naturalness of random sequences from the viewpoint of common players. However, there are some questions such as “are all fifteen feature necessary?” or “is there some necessary features missed?”, which need further research and study.

| Group | Evaluated score | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | Average |
| [Standard] | 11 | 19 | 2 | 11 | 5 | 2.58 |
| [Low rank] | 17 | 23 | 2 | 5 | 1 | 1.96 |
| [Optimized] | 7 | 9 | 4 | 23 | 5 | 3.21 |

TABLE III. THE EVALUATION RESULT

## VII. SUGOROKU

In the previous section, the naturalness of random sequences were directly evaluated via display programs. However, in pratical uses the naturalness of sequences might be affected by other factors . In this section, optimized sequences were evaluated by implemented in a simple dice game *Sugoroku*. In addition, player’s dissatisfaction was investigated.

### A. Sugoroku program

Fig. 3 shows a screen capture of the *Sugoroku* program. The experimental procedures are:

1) The token was placed at the start point (left-most cell).
2) “roll the dice” button was pushed by the player, and dice digit was selected and shown, from 1 to 6 by using a certain algorithm.
3) If there was no branch, the token was moved automatically by the showing digit cells.
4) If there were branches, a player selects one of upper/right/lower routes.
5) If the token stops at a red cell (Trap) then the player was lost. if the token reached green seal on the right side, then player win.
6) 10 games per set, and 4 sets were done. After each set (10 games) was played, it was announced to players that the random sequence was changed.

Totally 40 games were played by each player. The digit of dices are controlled by the following 4 algorithms,

1) Using the [Optimized] sequence generated in section VI.A.
2) Using the [Optimized] sequence and “Trap Control” algorithm as shown in the next subsection.
3) Using the [Low Rank] sequence.
4) Using the [Low Rank] sequence and “Trap Control” algorithm.

The order of sets is randomly shuffled, and it is hidden to players. Seventeen subject players were employed in the experiment, and 1000 JPY were rewarded to each subject player to incent them. Each player must decide the path while doing trial and error by themselves, because there optimal policy was not obvious.

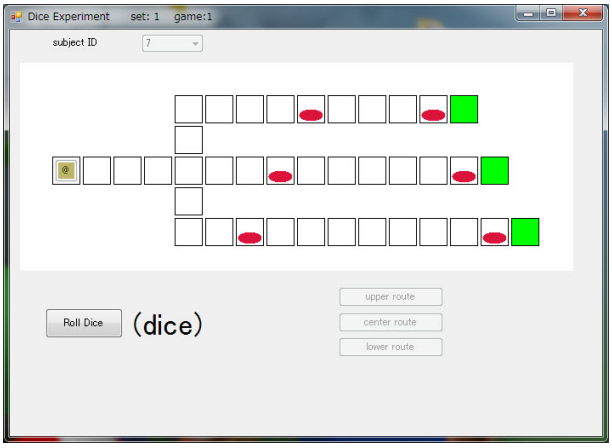


Fig. 7. Screen capture of simple Sugoroku program

### B. Trap control

In the games of *Sugoroku*, the belief of randomness might be divated while playing. The risen of numbers of being traped induce a negative feeling to players. For example, if a player is trapped 4 times in a row, the player will suspect that the dice is intentionally controlled, even if the dice sequence itself seems not to be biased. Then, a method to control the frequency and continuity of being trapped, with keeping the naturalness of the sequence would be proposed.

To control the number of being trapped, “trap sequences” were introduced. These sequences are randomly generated for each player. The following sequence is an example of trap sequences.

- 2,2,2,2,2,1,2,2,2,1,2,2,2,2,1,2,1,2,2,2,2,2,1,2

In the sequences “2” means “by next dice the player **should not** be trapped” and “1” means “by next dice the player **should** be trapped”. Trap sequence is optimized by the optimize algorithm shown on section V.B . The appearance rate of “1” is about 1/6, consecutive part are minimized.

The trap control will be activated when a trap exists in 6 cells ahead and a player is in situation that unable to avoid the trap; after a token passing crosspoint (fig 7). When the trap control is activated, the next trap control number is picked from trap sequence. Note that, $c$ is the distance between token and the next trap which is in the range $1 \leq c \leq 6$ , $t$ is a digit retrieved from a trap sequence, $d_0$ is the last displayed digit of the dice , $d_1$ is the next scheduled digit in the prepared dice sequence, and $d_2$ is a digit the next to $d_1$.

When $t = 2$ and $d_1 \neq c$, or $t = 1$ and $d_1 = c$, the dice schedule is already agreed with the trap controll, in this cases d1 shows just as it is.

When $t = 2$, $d_1 = c$ and $d_0 \neq d_2$, $d_2$ is display instead of $d_1$, to make the player untrapped according to the trap sequence. In fact $d_1$ and $d_2$ are swapped, then next digit is scheduled to $d_1$ instead of $d_2$. The condition $d_0 \neq d_2$ is introduced to avoid the continuation of the same digit.

Finally, when $t = 1$, $d_1 \neq c$ and $d_0 \neq c$, $c$ is shown instead of $d_1$, to make the player trapped according to the trap

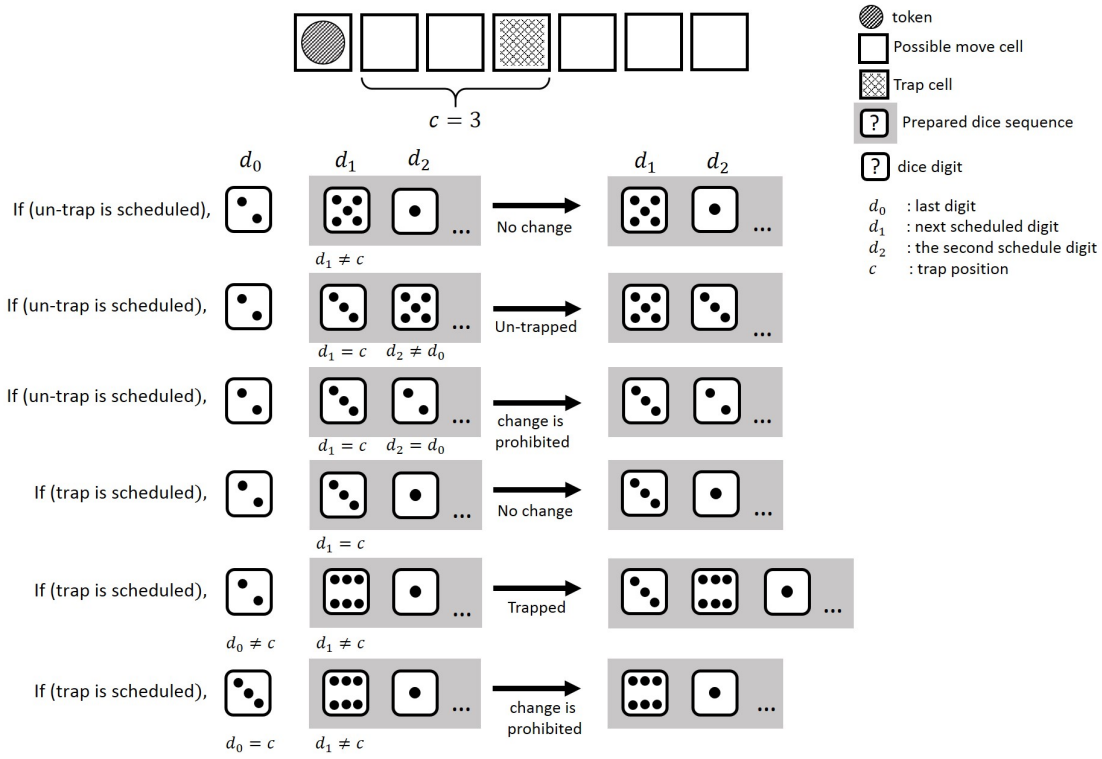


Fig. 8. Example of Trap control method

sequence. Here too, the condition $d_0 \neq c$ is introduced to avoid the continuation of the same digit too.

A trap control is activated for each *chance* to be trapped, not for each *game*. If the trap schedules are setup in each game, the result will be independent of the route selected by the player. Even if player selects an optimal route or a bad route, the result will still be the same. This feature is not suitable and should be avoided.

### C. Evaluation result: naturalness

Similar to the method in section VI, the naturalness of dice sequences which were used in each set of games were evaluated by human subjects. The averaged score of each set are shown in table IV.

| Group | Average score |
|---|---|
| [Optimized] | 3.12 |
| [Optimized] + Trap Control | 3.24 |
| [Low Rank] | 2.71 |
| [Low Rank] + Trap Control | 2.41 |

TABLE IV. THE NATURALNESS EVALUATION RESULT

[Optimized] sequences seemed significantly more natural than [Low Rank] sequences, the same trend was shown as the result in table III, though the difference was smaller. One reason might be that the last 6 digits were displayed in the previous experiment.

One more important result was that, the naturalness was not decreased by using Trap Control, when [Optimized] sequence was used. As the goal was to propose a method to control the frequency and continuity of being trapped, and keeping the naturalness, satisfaction of the latter condition was confirmed.

### D. Evaluation result: frequency and dissatisfaction

The number of the being trapped in each set (in 10 games) was also recorded. Considering the goal, the number is neither too big nor too small.

In the case of using the [Optimized] sequence without trap control, 4 players (of 17) were trapped only twice, and 2 players were trapped as often as 8 times. The average number was 4.23 whereas its standard deviation was 1.97.

In the case of using the [Optimized] sequence with trap control, 1 player was trapped only twice, and no player was trapped over 6 times. The average number was 4.53 that bigger than the previous case, but its standard deviation was 1.14 that fairly smaller. It was concluded that trap control is effective to avoid the “too lucky” or “too unlucky” cases, and keeping the naturalness.

After each set of 10 games question was asked to the subjects, “Do you think that dice was controlled to force you to be trapped?”. Subjects could answer 1-5, 1 is “Strongly yes” and 5 is “Strongly no”. In the case of using the [Optimized] sequence, the number of “strongly yes” is 4 , and in the case of that with trap control only 1. This result confirmed that this method is able to reduce the diviation of random numbers’ neturalness and dissatisfaction.

## VIII. Conclusion

In this research, a method for generation a biased random sequence that making players feel unbiased, according to the common cognitive biases of players, was shown. 15 features were employed to measure the "randomness from the viewpoint of common players", and sequences are optimized to satisfy a given condition. Generated sequences were evaluated and compared to standard random sequences, and it was proved that our method is effective. Also, a trap control algorithm has been proposed and employed to decrease dissatisfaction which can occur while playing dice games like monopoly.

However, this method might not be effective for some kind of players such as scientists or expert game players, because the optimized sequence may be far from the true randomness. To handle this problem, an adaptation for individual cognitive biases is need to be done. This approach will be proposed in near future.

## References


[1] F. James, "A review of pseudorandom number generators," *Computer Physics Communications*, vol. 60, no. 3, pp. 329 – 344, 1990.

[2] T. Preis, P. Virnau, W. Paul, and J. J. Schneider, "GPU accelerated monte carlo simulation of the 2d and 3d ising model," *Journal of Computational Physics*, vol. 228, no. 12, pp. 4468–4477, 2009.

[3] M. Matsumoto and T. Nishimura, "Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator," *ACM Trans. Model. Comput. Simul.*, vol. 8, no. 1, pp. 3–30, Jan. 1998.

[4] Culdcept official site. [Online]. Internet: http://www.culdcept.com/ , May. 31, 2013 [Apr. 22, 2014]

[5] The expaination of dice in culdcept (japanese). [Online]. Internet: http://www.omiyasoft.com/jnote07.html , [Apr. 22, 2014]

[6] Dungeons & dragons online. [Online]. Internet: https://www.ddo.com/forums/showthread.php/437332-Is-the-Daily-Dice-really-random , [May. 12, 2014]

[7] Pseudorandom number generation in pokemon. [Online]. Internet: http://bulbapedia.bulbagarden.net/wiki/Pseudorandom_number_generation_in_Pokemon , Apr. 8, 2013 [May. 19, 2014].

[8] World of warcraft. [Online]. Internet: http://us.battle.net/wow/en/ [June. 4, 2014]

[9] A. Tversky and D. Kahneman, "Judgment under uncertainty: Heuristics and biases," *Science*, vol. 185, no. 4157, pp. 1124–1131, 1974.

[10] P. Bakan, "Response-tendencies in attempts to generate random binary series," *The American Journal of Psychology*, vol. 73, no. 1, pp. 127–131, Mar. 1960.

[11] M. Bar-Hillel and W. A. Wagenaar, "The perception of randomness," *Advances in Applied Mathematics*, vol. 12, no. 4, pp. 428–454, 1991.

[12] M. F. Schilling, "The longest run of heads," *The college mathmetics journal*, vol. 21, no. 3, pp. 428–454, May 1990.

[13] K. Hiraoka, D. Shitamori, M. Ito, Y. Yakuwa, and T. Kawashima , "Game machine," Japan Patent 2011-177 454, Sep. 15, 2011.