Electronic Preprint for Journal of Information Processing Vol.21 No.1

Regular Paper

# The Number of Inequality Signs in the Design of Futoshiki Puzzle

Kazuya Haraguchi[1,a)]

Received: February 9, 2012, Accepted: September 10, 2012

**Abstract:** In this paper, we study how many inequality signs we should include in the design of Futoshiki puzzle. A problem instance of Futoshiki puzzle is given as an $n \times n$ grid of cells such that some cells are empty, other cells are filled with integers in $[n] = \{1, 2, \ldots, n\}$, and some pairs of two adjacent cells have inequality signs. A solver is then asked to fill all the empty cells with integers in $[n]$ so that the $n^2$ integers in the grid form an $n \times n$ Latin square and satisfy all the inequalities. In the design of a Futoshiki instance, we assert that the number of inequality signs should be an intermediate one. To draw this assertion, we compare Futoshiki instances that have different numbers of inequality signs from each other. The criterion is the degree to which the condition on inequality is used to solve the instance. If this degree were small, then the instance would be no better than one of a simple Latin square completion puzzle like Sudoku, with unnecessary inequality signs. Since we are considering Futoshiki puzzle, it is natural to take an interest in instances with large degrees. As a result of the experiments, the Futoshiki instances which have an intermediate number of inequality signs tend to achieve the largest evaluation values, rather than the ones which have few or many inequality signs.

**Keywords:** Puzzle construction, Futoshiki puzzle, Latin square

## 1. Introduction

We deal with Futoshiki puzzle throughout this paper. An *instance* of Futoshiki puzzle is given as an $n \times n$ grid of *cells* such that some cells are empty, other cells are assigned integers in $[n] = \{1, 2, \ldots, n\}$, and some pairs of two adjacent cells have inequality signs. Given an instance, a solver is asked to fill all the empty cells with integers in $[n]$ so that the $n^2$ integers in the grid satisfy the following two conditions.

**Latin square condition:** In each row and in each column, each integer in $[n]$ appears exactly once.

**Inequality condition:** When two adjacent cells have an inequality sign in between, the integers assigned to the two cells satisfy the inequality.

Note that Futoshiki is a Japanese word that means inequality. In **Fig. 1**, we show an example of a Futoshiki instance for $n = 4$, together with its solution.

Let us consider how to generate an instance of Futoshiki puzzle automatically. In our recent work [3], we proposed a framework to generate a puzzle instance of BlockSum puzzle with various levels of difficulty, where BlockSum is a puzzle of a similar kind that deals with a Latin square. Following this framework, we give an overview of an algorithm to generate a Futoshiki instance.

**1.** Decide the level of the solver we assume.
**2.** Generate an $n \times n$ Latin square randomly.
**3.** Decide a partial assignment of integers and the location of inequality signs such that;

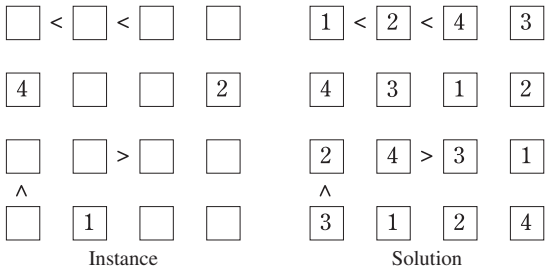


**Fig. 1** A Futoshiki instance and its solution ($n = 4$).

- the instance can be solved by the solver in **1**, and
- the instance has the Latin square in **2** as the unique solution.

In **1**, we implement a certain mechanism to solve a puzzle instance such that the mechanism reflects the level of the solver. The higher the level we assume, the more sophisticated the mechanism should be. In **2**, the Latin square serves as the unique solution of the produced instance. Solution uniqueness is employed in many combinatorial puzzles; e.g., Sudoku [5], [6]. In **3**, the decision can be made as follows. First, assign the value of the Latin square to each cell and embed the inequality sign between any adjacent cells so that the inequality is satisfied by the assigned values. Then repeat removing the assigned value from a cell or the inequality sign from adjacent cells as long as the instance satisfies the above two conditions.

In this paper, we discuss what is a suitable number of inequality signs in **3**. We assume that the solver in **1** and the $n \times n$ Latin square in **2** are given. We also assume that the partial assignment of integers in **3** has been determined in an appropriate way. We can allocate 0 to $2n(n-1)$ inequality signs to the grid, and the question is how many inequality signs are appropriate. To

[1] Faculty of Science and Engineering, Ishinomaki Senshu University, Ishinomaki, Miyagi 986–8580, Japan
[a)] kzyhgc@gmail.com

this question, our answer is an intermediate number. We derive the answer by comparing Futoshiki instances that have different numbers of inequality signs from each other. The criterion is the degree to which the solver uses the Inequality condition until it solves the instance. If the degree were small, the instance would be no better than an instance of "Latin square completion puzzle" like Sudoku, with unnecessary inequality signs. Since we are considering Futoshiki puzzle, it is natural to take interest in an instance that has a large degree.

Let us make an additional assumption on the given solver. We assume that the given solver will use the *candidate list strategy*, one of the traditional strategies for solving combinatorial puzzles. In this strategy, the solver maintains the list of candidate integers for each cell and repeats updating them by eliminating candidates based on certain *inference rules* until all the cells have single candidates. The inference rules are derived either from the Latin square condition or from the Inequality condition. Let us call the former one *L-rule* and the latter one *I-rule*. In our context, an instance seems preferable if, to solve the instance, the solver needs to eliminate many candidates by the I-rule. The number of candidates eliminated by the I-rule is expected to serve as our measure.

However, there must be a vast number of ways to solve a given instance. For example, the solver may encounter such a situation where more than one inference rule can be applied to update the candidate list. In such a case, the choice of inference rules by the solver affects the subsequent solving process and the number of candidate integers eliminated by the I-rule. This number is uniquely determined in one trial to solve the instance, but can be different between trials. Therefore, we observe the average of the numbers by conducting a large number of solving trials. In particular, we are interested in the lowest values for the worst-case analysis; we rate an instance high if even the lowest evaluation values are large. To estimate such values, we employ the "malicious" solver. The malicious solver always applies the L-rule whenever possible. In other words, it never applies the I-rule until the L-rule cannot be applied. It is expected that the malicious solver does not use the I-rule as frequently as ordinary solvers. Thus, we estimate the average of the lowest evaluation values from the behavior of the malicious solver.

We compare the instances that have different numbers of inequality signs by means of the criterion described above. Then we observe that those having an intermediate number of inequality signs achieve the highest evaluation values.

The paper is organized as follows. In Section 2, we introduce the notations and terminologies. In Section 3, we explain how to compare the instances. We present how to generate the instances to be compared and the comparison criterion. Then in Section 4, we present experimental results. Finally we give concluding remarks in Section 5.

## 2. Preliminaries

### 2.1 Latin Square and Futoshiki Instance

Let a cell in the $i$-th row and in the $j$-th column of the grid be denoted by $(i, j) \in [n]^2$. We say that two cells $(i, j)$ and $(i', j')$ are *adjacent* if $|i - i'| + |j - j'| = 1$. Let us denote a partial assignment of integers in $[n]$ to the $n^2$ cells by a partial function $\varphi : [n]^2 \to [n]$. The value $\varphi(i, j)$ represents the integer that is assigned to cell $(i, j)$. We denote the domain of $\varphi$ by $\mathrm{Dom}(\varphi)$, that is, $\mathrm{Dom}(\varphi) = \{(i, j) \in [n]^2 \mid \varphi(i, j) \text{ is defined}\}$. Let us denote $\mathrm{Emp}(\varphi) = [n]^2 \setminus \mathrm{Dom}(\varphi)$. For two partial assignments of integers $\varphi$ and $\psi$, we call $\psi$ an *extension of* $\varphi$, or equivalently, call $\varphi$ a *restriction of* $\psi$ if $\mathrm{Dom}(\varphi) \subseteq \mathrm{Dom}(\psi)$ and $\varphi(i, j) = \psi(i, j)$ holds for any cell $(i, j) \in \mathrm{Dom}(\varphi)$. We call $\varphi$ a *partial Latin square* if, in each row and in each column, any integer appears at most once. If a partial Latin square $\varphi$ is a total function (i.e., $\mathrm{Dom}(\varphi) = [n]^2$), we simply call it a *Latin square*.

Let us denote the set of all the ordered pairs of adjacent cells by $P$;

$$P = \{((i, j), (i', j')) \in [n]^2 \times [n]^2 \mid (i, j) \text{ and } (i', j') \text{ are adjacent}\}.$$

Clearly we have $|P| = 4n(n - 1)$. We call a subset $S$ of $P$ a *sign set* if, for any two adjacent cells $(i, j)$ and $(i', j')$, at most one of $((i, j), (i', j'))$ and $((i', j'), (i, j))$ appears in $S$. Any sign set $S$ satisfies $|S| \leq 2n(n - 1)$.

We denote a Futoshiki instance by $I$. We define $I$ as a pair $I = (\varphi, S)$, where $\varphi : [n]^2 \to [n]$ is a partial Latin square and $S$ is a sign set. The cells of $\mathrm{Dom}(\varphi)$ are assigned integers $\varphi(i, j)$'-s that are given as hints to the solver, whereas the cells of $\mathrm{Emp}(\varphi)$ are the empty cells that the solver should complete. The sign set $S$ represents the set of inequality signs in the instance, i.e., $((i, j), (i', j')) \in S$ indicates that the solver needs to assign a smaller integer to $(i, j)$ than $(i', j')$. Since $S$ is a sign set, $((i, j), (i', j')) \in S$ implies $((i', j'), (i, j)) \notin S$. Then we have no contradicting inequality sign. A *partial solution to* $I$ is a partial Latin square $\psi$ that is an extension of $\varphi$ and that satisfies the inequality $\psi(i, j) < \psi(i', j')$ if $((i, j), (i', j')) \in S$ and $(i, j), (i', j') \in \mathrm{Dom}(\psi)$. A partial solution is simply called a *solution* if it is a Latin square.

### 2.2 Candidate List Strategy

The candidate list strategy refers to an algorithm that solves a given instance as follows. Let us denote a given instance by $I = (\varphi, S)$. A *candidate list* is defined as a mapping $C : [n]^2 \to 2^{[n]}$, where $2^{[n]}$ denotes the power set of $[n]$. The set $C(i, j) \subseteq [n]$ represents integers that can be assigned to $(i, j)$. The candidate list strategy starts with the following mapping;

$$C(i, j) = \begin{cases} \{\varphi(i, j)\} & \text{if } (i, j) \in \mathrm{Dom}(\varphi), \\ [n] & \text{otherwise.} \end{cases}$$

At this time, the number of all candidate integers over the $n^2$ cells is;

$$\sum_{(i,j)\in[n]^2} |C(i, j)| = |\mathrm{Dom}(\varphi)| + n|\mathrm{Emp}(\varphi)| = n^2 + (n - 1)|\mathrm{Emp}(\varphi)|.$$

Then we repeat choosing an appropriate cell $(i, j)$ and updating $C(i, j)$ by eliminating some of the integers in it based on inference rules that we describe later. During the process, if $C(i, j)$ is reduced to a singleton $\{v\}$, which means that there is no candidate other than $v$, then the integer $v$ can be assigned to the cell $(i, j)$. In other words, we can construct a partial solution $\psi$ as follows;

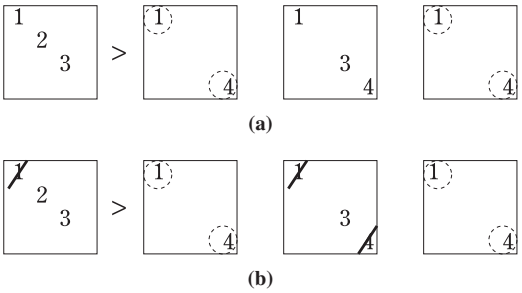


**Fig. 2** Example of Naked Pair: (a) The candidate list before the rule is applied. (b) The candidate list after the rule is applied.

$$\psi(i,j) = \begin{cases} v & \text{if } C(i,j) \text{ is a singleton } \{v\}, \\ \text{(not defined)} & \text{otherwise.} \end{cases}$$

Then we have solved $I$ if $C(i,j)$ is reduced to a singleton for every $(i,j) \in [n]^2$. When $I$ is solved, there are $n^2$ candidate integers over the $n^2$ cells. Hence, the number of candidate integers that have been eliminated is;

$$n^2 + (n-1)|\mathrm{Emp}(\varphi)| - n^2 = (n-1)|\mathrm{Emp}(\varphi)|. \quad (1)$$

Hence, if we cannot reduce $(n-1)|\mathrm{Emp}(\varphi)|$ candidate integers in any way, the instance cannot be solved by the candidate list strategy.

What we call inference rules are logical techniques that are used to eliminate the $(n-1)|\mathrm{Emp}(\varphi)|$ candidate integers. We use two types of inference rules, that is L-rule and I-rule, where the former rule (resp., the latter rule) is derived from the Latin square condition (resp., the Inequality condition). For the L-rule, we take inference rules that are prevalent in Sudoku. Before we derive their generalized formulation, let us illustrate Naked Pair [1], [4] for example. See **Fig. 2**. The example assumes $n = 4$. The figure (a) shows the candidate list of a certain row before Naked Pair is applied. The digits in a cell indicate the candidate integers of the cell. Naked Pair updates the candidate list as follows; since the two cells in the 2nd and 4th columns contain the same 2 candidates, that is 1 and 4, these two cells can have 1 and 4 respectively, or 4 and 1 respectively. Then none of the other cells in this row can take 1 or 4. We can eliminate 1 and 4 from the candidate lists of the other cells, as shown in (b). In this case, three candidate integers are eliminated.

We formulate the L-rule by generalizing Naked Pair as follows. Assume that a candidate list $C$ is given. For an arbitrary row, say the $i$-th row ($i \in [n]$), we construct an $n \times n$ 0-1 matrix. We denote the matrix by $M$ and the entry in the $x$-th row and in the $y$-th column by $M_{x,y}$ ($x, y \in [n]$). We define $M_{x,y}$ so that it indicates whether the integer $x$ is contained in the list $C(i,y)$;

$$M_{x,y} = \begin{cases} 1 & \text{if } x \in C(i,y), \\ 0 & \text{otherwise.} \end{cases}$$

Let $Y \subseteq [n]$ denote a column subset of $M$, and $X(Y) \subseteq [n]$ denote the row subset of $M$ such that $X(Y) = \{x \in [n] \mid M_{x,y} = 1 \text{ for some } y \in Y\}$. Let $k \in [n]$ denote a natural number. Then if $M$ has a column subset $Y$ such that $|Y| = |X(Y)| \leq k$, all the entries whose rows are in $X(Y)$ and whose columns are out of $Y$ can be turned into 0. In other words, we can eliminate the integer $x$ from the list of cell $(i,y)$, i.e., $C(i,y) \leftarrow C(i,y) \setminus \{x\}$, for any $(x,y) \in X(Y) \times ([n] \setminus Y)$. The number of candidate integers eliminated becomes;

$$\sum_{(x,y) \in X(Y) \times ([n] \setminus Y)} M_{x,y}.$$

One sees that Naked Pair is a special case of this rule for $k = 2$.

In the above, we constructed the matrix $M$ for a fixed row of the grid. We associated a candidate integer with each row of $M$, and a column of the grid with each column of $M$. Interestingly, we can update the candidate list as above even if we exchange the roles of grid row, grid column and candidate integer. Some of the prevalent rules can be explained in this way, and we show the relationship between our formulation and the prevalent rules in **Table 1**. In the rest of the paper, we assume that $k$ is a given constant.

**Table 1** Relationship between our formulation and the prevalent inference rules (R: grid row, C: grid column, I: candidate integer).

| Fix | Row of $M$ | Column of $M$ | $k=1$ | $k=2$ | $k=3$ |
|---|---|---|---|---|---|
| R | I | C | Naked | Naked | Naked |
| C | I | R | Single | Pair | Triple |
| R | C | I | Hidden | Hidden | Hidden |
| C | R | I | Single | Pair | Triple |
| I | R | C | - | X-wing | Sword-fish |
| I | C | R | | | |

Next we introduce the I-rule. For any pair $((i,j),(i',j')) \in S$ of adjacent cells that have an inequality sign, some of the candidate integers in $C(i,j)$ and $C(i',j')$ can be eliminated from the Inequality condition in the following way. Let $\mu = \min\{v \in C(i,j)\}$ and $\nu = \max\{v' \in C(i',j')\}$. Since the integer assigned to $(i,j)$ cannot be larger than or equal to $\nu$, we can eliminate $\nu, \nu+1, \ldots, n$ from $C(i,j)$, i.e., $C(i,j) \leftarrow C(i,j) \setminus \{\nu, \nu+1, \ldots, n\}$. Similarly, since the integer assigned to $(i',j')$ cannot be smaller than or equal to $\mu$, we can eliminate $1, 2, \ldots, \mu$ from $C(i',j')$, i.e., $C(i',j') \leftarrow C(i',j') \setminus \{1, 2, \ldots, \mu\}$. The number of integers eliminated is;

$$|C(i,j) \cap \{\nu, \nu+1, \ldots, n\}| + |C(i',j') \cap \{1, 2, \ldots, \mu\}|. \quad (2)$$

For example, in Fig. 2 (a), an inequality sign exists between the cells of the 1st and 2nd columns. Then the candidate integer 1 in the cell of the 1st column can be eliminated, and the candidate integer 4 in the cell of the 2nd column can be also eliminated.

## 3. How to Compare Futoshiki Instances

In this section, we explain how to compare Futoshiki instances having different numbers of inequality signs from each other. In Section 3.1, we present how to generate the instances to be compared. In Section 3.2, we explain the criterion for the comparison.

### 3.1 How to Generate Instances

For the comparison, we would like to generate several instances so that they have different numbers of inequality signs from each other. Besides, they need to satisfy the following conditions;

**(i)** the instances can be solved by the candidate strategy,

**(ii)** the instances have the given Latin square as the unique solution, and

**(iii)** the instances have the same partial Latin square as the hints which are given to the solver.

We need some preparatory procedures before presenting the algorithm. We denote the given Latin square by $\ell$. Due to (ii), we restrict ourselves to the instances that have $\ell$ as a solution. Let us denote by $\mathcal{I}$ the set of all the instances that have $\ell$ as a solution. We define $S_\ell$ as the sign set induced by $\ell$, i.e.,

$$S_\ell = \{((i, j), (i', j')) \in P \mid \ell(i, j) < \ell(i', j')\}.$$

**Proposition 1** An instance $(\varphi, S)$ has $\ell$ as a solution if and only if $\varphi$ is a restriction of $\ell$ and $S$ is a subset of $S_\ell$.

Using Proposition 1, $\mathcal{I}$ can be expressed as;

$$\mathcal{I} = \{(\varphi, S) \mid \varphi \text{ is a restriction of } \ell \text{ and } S \subseteq S_\ell\}.$$

We introduce a binary relation on the instances in $\mathcal{I}$. For any two instances in $\mathcal{I}$, say $(\varphi, S)$ and $(\varphi', S')$, we write $(\varphi, S) \leqslant (\varphi', S')$ if $\varphi$ is a restriction of $\varphi'$ and $S \subseteq S'$. Clearly the relation $\leqslant$ is a partial order. Then $\mathcal{I}$ has the top element and the bottom element with respect to $\leqslant$. The top is $I_\top = (\ell, S_\ell)$ and the bottom is $I_\bot = (\varphi_{\text{emp}}, \emptyset)$, where $\varphi_{\text{emp}}$ denotes an empty function (i.e., $\text{Dom}(\varphi_{\text{emp}}) = \emptyset$). Since $(\mathcal{I}, \leqslant)$ is a partially ordered set, it can be represented by means of the Hasse diagram. In the Hasse diagram, each node corresponds to an instance in $\mathcal{I}$. An edge is drawn between $(\varphi, S)$ and $(\varphi', S')$ if $(\varphi, S) \leqslant (\varphi', S')$ and exactly one of the following two equalities holds: $|\text{Dom}(\varphi') \setminus \text{Dom}(\varphi)| = 1$ or $|S' \setminus S| = 1$. The nodes are arranged so that $(\varphi, S)$ is placed under $(\varphi', S')$ if $(\varphi, S) \leqslant (\varphi', S')$. In particular, $I_\top$ is placed at the top and $I_\bot$ is placed at the bottom.

We call an instance *solvable* if it can be solved by the candidate list strategy anyhow. The strategy solves an instance by updating the candidate list based on inference rules, that is logic. Hence, any solvable instance has a unique solution. To meet (i) and (ii), we search for solvable instances in $\mathcal{I}$.

**Proposition 2** Let $(\varphi, S)$ and $(\varphi', S')$ denote instances such that $(\varphi, S) \leqslant (\varphi', S')$. If the instance $(\varphi, S)$ is solvable, $(\varphi', S')$ is also solvable.

Proposition 2 means that solvability is upward monotone with respect to $\leqslant$.

Now we are ready to present the algorithm to generate instances. We illustrate the Hasse diagram of $\mathcal{I}$ and the trajectory of the algorithm in **Fig. 3**. The algorithm starts the search from the *initial instance*, denoted by $I_{\text{init}}$. We define the initial instance as the one such that all the $n^2$ cells are empty and any two adjacent cells have an inequality sign, that is, $I_{\text{init}} = (\varphi_{\text{emp}}, S_\ell)$. In Fig. 3, the initial instance is represented by the black point. Note that the initial instance is not necessarily solvable. First, the algorithm gives appropriate integers to some empty cells so that the instance is solvable. In other words, the algorithm finds a certain restriction of $\ell$. This corresponds to climbing upward along edges of the Hasse diagram. Suppose that the algorithm eventually finds a restriction $\varphi$ of $\ell$ such that $(\varphi, S_\ell)$ is solvable. Let us denote this instance by $I^0 = (\varphi, S)$, where $S = S_\ell$. The instance $I^0$ is represented by the shaded point in Fig. 3. In the sequel, we may write $I^0$ as $I_{\max}$ since it has $2n(n-1)$ inequality signs, which is the maximum. Then, the algorithm repeats

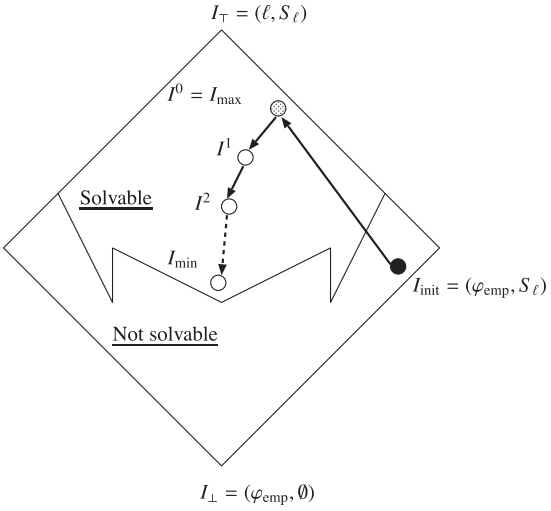


**Fig. 3** Hasse diagram of the instance set $\mathcal{I}$ and the trajectory of the algorithm.

**1.** Find a restriction $\varphi$ of the given Latin square $\ell$ such that $(\varphi, S_\ell)$ is solvable.
**2.** Let $\mathcal{H} \leftarrow \emptyset$ and $S \leftarrow S_\ell$.
**3.** Repeat the followings;
  **3-1.** Let $\mathcal{H} \leftarrow \mathcal{H} \cup \{(\varphi, S)\}$.
  **3-2.** If there exists $e = ((i, j), (i', j')) \in S$ such that $(\varphi, S \setminus \{e\})$ is solvable, then $S \leftarrow S \setminus \{e\}$. Otherwise, output $\mathcal{H}$ and halt.

**Fig. 4** Algorithm to generate instances for the comparison experiment.

removing an inequality sign as long as the instance is solvable. In other words, the algorithm lets $S \leftarrow S \setminus \{((i, j), (i', j'))\}$ for a certain $((i, j), (i', j')) \in S$. The removal of inequality signs corresponds to descending downward along edges. Finally, the algorithm halts if it encounters a *minimal solvable instance*, i.e., an instance such that the removal of any inequality sign results in an unsolvable instance. During this process, we find the solvable instances $I^0 = I_{\max}, I^1, \ldots, I_{\min}$, where $I_{\min}$ denotes a minimal solvable one. Clearly these instances have different numbers of inequality signs from each other and also satisfy the above conditions (i) to (iii). We use them for the comparison experiment.

We summarize the algorithm in **Fig. 4**. In **1**, we can obtain $\varphi$ by extending the domain of an empty function repeatedly; starting with $\varphi = \varphi_{\text{emp}}$, repeat letting $\varphi(i, j) \leftarrow \ell(i, j)$ for a certain $(i, j) \in \text{Emp}(\varphi)$ until $(\varphi, S_\ell)$ is solvable. In the experiment, however, we borrow the given Latin square $\ell$ and its restriction $\varphi$ from instances on the Internet. Whether an instance is solvable or not can be recognized by applying the candidate list strategy. The detailed implementation of the candidate list strategy is explained in the next subsection, together with the criterion for comparing the instances.

### 3.2 Criterion for Comparing Instances

Basically, our criterion for comparing instances is the number of candidate integers that are eliminated by the I-rule when the instance is solved by the candidate list strategy. In the course of solving the instance, when the I-rule is applied, the number of eliminated candidates is counted by Eq. (2). The evaluation value

is the sum of Eq. (2) over a trial to solve the instance. Hence, it is unique in one trial, but can be different between trials. For example, the given solver may encounter such a situation where more than one inference rule can be applied. Figure 2 (a) is a good example. We can use not only the L-rule (Naked Pair) but also the I-rule to eliminate the integer 1 from the cell in the 1st column. The choice of inference rules should affect the evaluation value.

To observe the tendency of the evaluation values, we sample a sufficiently large number of solving trials. Let $T$ denote a natural number. We solve a given instance $I$ by applying the candidate list strategy $T$ times. Let $\rho_t(I)$ ($t \in [T]$) denote the number of candidate integers that are eliminated by the I-rule in the $t$-th trial. Then we compute the average;

$$\hat{\rho}(I) = \frac{1}{T}\sum_{t=1}^{T}\rho_t(I). \tag{3}$$

To solve an instance, we need to eliminate $(n-1)|\mathrm{Emp}(\varphi)|$ candidate integers, as seen in Eq. (1). We normalize the average $\hat{\rho}(I)$ by this number. We call the normalized value the *evaluation ratio*, denoted by $r(I)$, and use it as the criterion.

$$r(I) = \frac{\hat{\rho}(I)}{(n-1)|\mathrm{Emp}(\varphi)|}. \tag{4}$$

The larger $r(I)$ is, the higher we rate $I$.

Among all the possible evaluation ratios of the instance, we are interested in the lowest ones for the worst-case analysis; we can rate the instance high if even the lowest evaluation ratios are large. To estimate the lowest evaluation ratios, we employ the "malicious" solver that never uses the I-rule as long as the L-rule can be applied. Formally, the solver works as follows;

**1.** As long as there is a non-empty subset $A \subseteq [n]^2$ of cells to which the L-rule can be applied, update $C(i, j)$ for any cell $(i, j) \in A$ by the L-rule.

**2.** Choose a pair $((i, j), (i', j')) \in S$ arbitrarily such that the I-rule can be applied to $(i, j)$ and $(i', j')$. Update $C(i, j)$ and $C(i', j')$ by the I-rule. The number of eliminated integers is counted by Eq. (2).

**3.** If the instance is not solved, return to **1**.

In **2**, we have the freedom to choose a pair of adjacent cells among the ones to which the I-rule can be applied. We choose one pair at random. Then, the observed values $\rho_t(I)$'-s can be different between trials.

## 4. Computational Experiments

In this section, we compare Futoshiki instances that have different numbers of inequality signs from each other.

### 4.1 Experimental Settings

In the candidate list strategy we assume, the L-rule is the collection of Naked Single, Naked Pair, Hidden Single and Hidden Pair in Table 1, and the I-rule is the one described in Section 2.2. We decide the Latin square $\ell$ of the solution of the generated instances in the following way; We pick up 18 instances from sudoku-puzzles.net [2]. Let us call them *original instances*. We solve each original instance to find its solution, and use the solution as $\ell$. We find that each original instance can be solved by the candidate list strategy, and thus $\ell$ is the unique solution. Let us denote the original instance by $I_{\mathrm{orig}} = (\varphi, S_{\mathrm{orig}})$. We generate instances that have $\ell$ as the unique solution by the algorithm in Fig. 4. We need to specify a restriction of $\ell$ in **1**. For this, we use the one $\varphi$ of the original instance. The instance $I^0 = (\varphi, S_\ell)$ is surely solvable since $I_{\mathrm{orig}} \leqslant I^0$ and $I_{\mathrm{orig}}$ is solvable. In **3-2**, we employ the greedy method for selecting a pair of adjacent cells to be removed, that is, we remove the pair of adjacent cells so that the evaluation ratio achieves the highest. To compute the evaluation ratio of an instance $I$, we solve $I$ by the candidate list strategy 100 times, that is, we use $T = 100$ in Eq. (3).

**Table 2** Summary of the original instances from sudoku-puzzles.net [2].

| $n$ | ID | Empty cells | Signs |
|---|---|---|---|
| 5 | 321860229 | 19 | 5 |
| 5 | 323730102 | 19 | 3 |
| 5 | 326429168 | 23 | 7 |
| 5 | 327451171 | 23 | 11 |
| 5 | 325826019 | 25 | 10 |
| 5 | 326412689 | 25 | 10 |
| 7 | 341198303 | 33 | 6 |
| 7 | 342599334 | 33 | 7 |
| 7 | 341189239 | 41 | 19 |
| 7 | 341579254 | 41 | 17 |
| 7 | 341240325 | 45 | 17 |
| 7 | 341819854 | 45 | 18 |
| 9 | 362803405 | 57 | 18 |
| 9 | 365807617 | 57 | 16 |
| 9 | 366299530 | 69 | 35 |
| 9 | 366364349 | 69 | 34 |
| 9 | 361309814 | 75 | 41 |
| 9 | 364108856 | 75 | 47 |

### 4.2 Results

We show the summary of the 18 original instances in **Table 2**. Each original instance is given its ID number. One can see the instance and play with it by entering the ID number at sudoku-puzzles.net. The column "Empty cells" represents the number of empty cells, that is $|\mathrm{Emp}(\varphi)|$, and the one "Signs" represents the number of inequality signs, that is $|S_{\mathrm{orig}}|$.

For each original instance $(\varphi, S_{\mathrm{orig}})$, we generate the instance set $\mathcal{H} = \{I^0 = I_{\max}, I^1, \ldots, I_{\min}\}$ in the way described above. Among the instances in $\mathcal{H}$, $I_{\max}$ has the maximum number of inequality signs, that is $2n(n-1)$, and $I_{\min}$ has the minimum number of inequality signs. We denote by $I_*$ the instance that achieves the highest evaluation ratio among the ones in $\mathcal{H}$.

In **Table 3**, we show the numbers of inequality signs and the evaluation ratios of $I_{\mathrm{orig}}$, $I_{\min}$, $I_*$ and $I_{\max}$. By definition, $I_*$ achieves higher evaluation ratios than $I_{\min}$ and $I_{\max}$. It is also better than $I_{\mathrm{orig}}$. Judging from the number of inequality signs and the evaluation ratio, $I_{\mathrm{orig}}$ is close to $I_{\min}$ rather than $I_*$ or $I_{\max}$. The original instances and the minimal solvable instances are not good in our context.

It appears that, if an instance has few inequality signs, the evaluation ratio would be low since the solver should meet few chances to apply the I-rule. This is supported by the following fact; Let us denote an instance by $I = (\varphi, S)$. Clearly we have $\hat{\rho}(I) \leq 2(n-1)|S|$, where $|S|$ denotes the number of inequality signs. From Eq. (4), we have;

$$r(I) = \frac{\hat{\rho}(I)}{(n-1)|\mathrm{Emp}(\varphi)|} \leq \frac{2|S|}{|\mathrm{Emp}(\varphi)|}. \tag{5}$$

**Table 3** The numbers of inequality signs and the evaluation ratios.

| $n$ | ID | $I_{\text{orig}}$ | | $I_{\min}$ | | $I_*$ | | $I_{\max}$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | Signs | Ratio | Signs | Ratio | Signs | Ratio | Signs | Ratio |
| 5 | 321860229 | 5 | .133 | 5 | .176 | 15 | .234 | 40 | .178 |
| 5 | 323730102 | 3 | .072 | 3 | .103 | 21 | .175 | 40 | .112 |
| 5 | 326429168 | 7 | .223 | 7 | .270 | 33 | .372 | 40 | .340 |
| 5 | 327451171 | 11 | .299 | 8 | .263 | 22 | .400 | 40 | .373 |
| 5 | 325826019 | 10 | .304 | 12 | .412 | 22 | .503 | 40 | .431 |
| 5 | 326412689 | 10 | .339 | 14 | .472 | 18 | .502 | 40 | .442 |
| 7 | 341198303 | 6 | .070 | 8 | .114 | 36 | .175 | 84 | .120 |
| 7 | 342599334 | 7 | .050 | 8 | .089 | 24 | .116 | 84 | .076 |
| 7 | 341189239 | 19 | .222 | 17 | .224 | 57 | .337 | 84 | .249 |
| 7 | 341579254 | 17 | .184 | 20 | .240 | 52 | .344 | 84 | .268 |
| 7 | 341240325 | 17 | .255 | 25 | .353 | 62 | .453 | 84 | .379 |
| 7 | 341819854 | 18 | .234 | 23 | .314 | 57 | .446 | 84 | .380 |
| 9 | 362803405 | 18 | .101 | 29 | .160 | 97 | .220 | 144 | .157 |
| 9 | 365807617 | 16 | .097 | 20 | .131 | 98 | .222 | 144 | .146 |
| 9 | 366299530 | 35 | .245 | 37 | .306 | 90 | .411 | 144 | .321 |
| 9 | 366364349 | 34 | .241 | 36 | .272 | 109 | .391 | 144 | .311 |
| 9 | 361309814 | 41 | .319 | 43 | .375 | 122 | .485 | 144 | .423 |
| 9 | 364108856 | 47 | .352 | 43 | .358 | 103 | .500 | 144 | .409 |

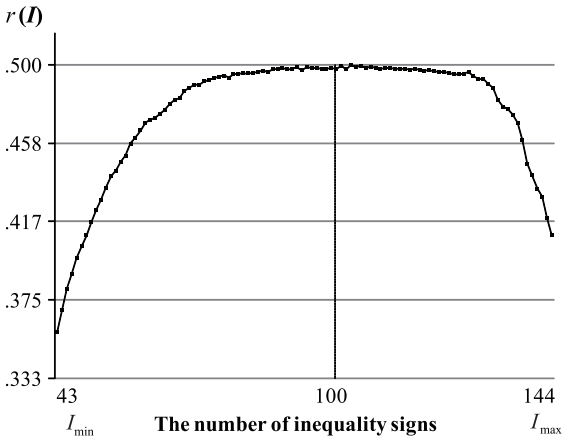


**Fig. 5** Evaluation ratios of the instances generated from the original instance with the ID number 364108856 ($n = 9$).

If $I$ has few inequality signs, that is, if $|S|$ is small, the above upper bound becomes small. Then $r(I)$ is expected to achieve a small value. In fact, $r(I_{\min}) \leq r(I_{\max})$ holds for 15 out of the 18 cases; the 3 exceptional cases are the original instances with the ID numbers 326412689 ($n = 5$), 342599334 ($n = 7$), and 362803405 ($n = 9$). However, we have $r(I_{\max}) \leq r(I_*)$ although $I_*$ has a significantly smaller number of inequality signs than $I_{\max}$. A reason is described as follows. Among the inequality signs in $I_{\max}$, there must be ones such that applying the I-rule to them immediately leads us to a situation where the L-rule can be applied. Even if we remove such inequality signs from $I_{\max}$, the evaluation ratio must not be decreased, or may be even increased.

In **Fig. 5**, we illustrate this observation by showing the evaluation ratios of all the instances in $\mathcal{H} = \{I^0 = I_{\max}, \ldots, I_*, \ldots, I_{\min}\}$. The instances are generated from the original instance with the ID number 364108856 ($n = 9$). In this generation process, the algorithm starts from $I_{\max}$ and repeats removing an inequality sign until it reaches a minimal solvable instance $I_{\min}$. Let us follow how the evaluation ratio changes as the number of inequality signs is decreasing. Starting from .409, the evaluation ratio first increases up to almost .500. It remains to be so high values as .500 for a while, and dramatically decreases when the number of inequality signs is smaller than 100. We observe a similar tendency in other $\mathcal{H}$'-s that are generated from other original instances; as shown in Fig. 5, the instances that have an intermediate number of inequality signs achieve the highest evaluation ratios.

Let us observe these results. For each number of inequality signs, the algorithm should find an instance that attains approximately the largest evaluation ratio among those having the same number of inequality signs since we employ the greedy method in the choice of the inequality sign to be removed (see Section 4.1). Then, among all the found instances, the one $I_*$ with an intermediate number of inequality signs achieves the largest evaluation ratio in all the 18 test cases. Thus, in the instances that have the largest evaluation ratios, the number of inequality signs should be intermediate rather than small or large. Based on these, we claim that one should include an intermediate number of inequality signs in the design of a Futoshiki instance.

## 5. Concluding Remarks

In this paper, we started to consider how to design a Futoshiki instance automatically. Employing the algorithmic framework presented in our recent work on BlockSum puzzle [3], we have considered what is a suitable number of inequality signs. To compare instances having different numbers of inequality signs, we introduced the evaluation ratio as the criterion. The evaluation ratio estimates the degree to which the I-rule is needed to solve the instance. Since the puzzle is "Futoshiki puzzle," the larger the evaluation ratio is, the higher we rate the instance. In the experimental studies, we observed that the instances with an intermediate number of inequality signs achieve the largest evaluation ratios.

Given the number of inequality signs, it is another task to decide their location so that the evaluation ratio is large. Our future work includes development of a fast algorithm to do this. The algorithm we used to generate the instances is not practical in terms of computation time. For example, our implementation takes about $5.7 \times 10^3$ seconds to generate the instance set $\mathcal{H}$ from an original instance of $n = 9$. This is mainly because the algorithm searches too many instances and solves each found instance by the candidate list strategy many times to compute its evaluation ratio. Then we need not only to reduce the search space but

also to develop a fast implementation of the candidate list strategy.

We introduced the evaluation ratio $r$ only to compare the instances that have different numbers of inequality signs from each other. However, this criterion can be used to compare any two instances. In Table 3, the ranges of the evaluation ratios are different between original instances even if they have the same $n$. We conjecture that the range should be strongly affected by the Latin square of the unique solution. We are interested in what kind of Latin squares induce a highly rated Futoshiki instance. We are also interested in the relationship between the evaluation ratio and the difficulty level. These are left for future work.

## References


[1] Davis, T.: The Mathematics of Sudoku (online), available from ⟨http://www.geometer.org/mathcircles/sudoku.pdf⟩ (accessed 2011-06-01).

[2] Flueckiger, M.: Sudoku-Puzzles.net (online), available from ⟨http://www.sudoku-puzzles.net/⟩ (accessed 2011-06-01).

[3] Haraguchi, K., Abe, Y. and Maruoka, A.: How to Produce BlockSum Instances with Various Levels of Difficulty, *Journal of Information Processing*, Vol.20, No.3, pp.727–737 (2012).

[4] Johnson, A.: Simple Sudoku (online), available from ⟨http://www.angusj.com/sudoku/⟩ (accessed 2011-06-01).

[5] Lewis, R.: Metaheuristics can solve sudoku puzzles, *Journal of Heuristics*, Vol.13, pp.387–401 (2007).

[6] Simonis, H.: Sudoku as a Constraint Problem, *Proc. 4th International Workshop of Modelling and Reformulating Constraint Satisfaction Problems*, pp.13–27 (2005).


**Kazuya Haraguchi** received his B.E., Master of Informatics, and Doctor of Informatics from Kyoto University, in 2001, 2003 and 2007, respectively. He is currently with the Department of Information Technology and Electronics, Faculty of Science and Engineering, Ishinomaki Senshu University. His interests include algorithms, optimization, and their application to artificial intelligence and operations research.