近畿能開大ジャーナル
第22号（2014年11月）

Journal of Kinki Polytechnic
College.No.22.November.（2014）

A Study on Solving Polyomino Packing Puzzle Using Genetic Algorithm

Michio INNAMI*

Abstract

This paper attempts to investigate the search ability for solutions of a polyomino packing puzzle using genetic algorithm (GA). The puzzle pieces are formed by joining several squares along the edges. Polyomino packing puzzles are NP-complete. It is rather difficult for GA to solve this type of problems in terms of the building block hypothesis because the partial solutions interfere with each other.

In this paper, genetic coding methods are proposed, that makes the search space narrower. Therefore it is expected to improve search performance. Also the effects of niching operators such as sharing and clearing are investigated. A 9-piece puzzle of which the board has 6 rows and 6 columns is taken as a sample problem.

The computer experiments were executed by above methods. The results indicate that the proposed approach efficiently find the solutions. In addition, it is shown that the niching methods operate sufficiently upon the exploration.

1 Introduction

The polyomino packing puzzle game is a two-dimensional combinatorial puzzle. The puzzle pieces are formed by joining several squares along the edges. The game has been played since the mid-18th century [1]. This kind of puzzle is NP-complete [2]. A puzzle consisting of many pieces has a huge number of combination to arrange them. In such a case it is a quite complicated problem.

This paper tries to find the solution for a polyomino packing puzzle using genetic algorithm (GA). It is rather difficult for GA to solve this type of problems in terms of the building block hypothesis [3] because the partial solutions interfere with each other. That is to say, it is easy to trap into local optima. It costs much to get an optimal solution.

This study evaluates proposed genetic coding methods that may be efficient for solving this type of puzzles. Furthermore, niching methods are applied in order to maintain multiple solutions, and to avoid convergence to local optima.

This paper is structured as follows. Section 2 overviews GA and scaling methods, while Section 3 describes niching methods. Section 4 presents a sample problem. Section 5 shows proposed genetic coding methods. Section 6 discusses experimental results. Finally Section 7 concludes the paper.

2 Genetic algorithm

Genetic algorithm is one of the population-based optimization techniques. It has been applied to a number of optimization problems in various fields.

2.1 Simple genetic algorithm

The simple genetic algorithm (SGA) consists of following steps.

1. Randomly create initial individuals in population.
2. Evaluate the fitness value of each individual.
3. Select individuals for next generation according to their fitness values.

*Department of Advanced Information Technology for Production System

4. Randomly form pairs of individuals then apply crossover operation with certain crossover rate, to generate offspring individuals.
5. Mutate the offspring individuals with certain mutation rate.
6. Go to step 2 until the solution has converged.

2.2 Elite selection

Several individuals with the best fitness of each generation are carried over to the next generation without any operation. This strategy is considered to accelerate the convergence of solutions.

2.3 Scaling functions

Scaling functions convert fitness values to make the range suitable for selection. Linear scaling methods are one of the most common scaling functions. This function scales fitness values using the following equations.

$$f_s = af + b \tag{1}$$

$$a = \frac{(c-1)f_{\mathrm{avg}}}{f_{\max} - f_{\mathrm{avg}}} \tag{2}$$

$$b = \frac{f_{\mathrm{avg}}(f_{\max} - cf_{\mathrm{avg}})}{f_{\max} - f_{\mathrm{avg}}} \tag{3}$$

where $f_{\max}$ and f_{avg} are the maximum and average values of the fitness function, respectively.

3 Niching methods

Many niching methods have been proposed to maintain diversity of solutions. This section describes sharing and clearing methods used in this study.

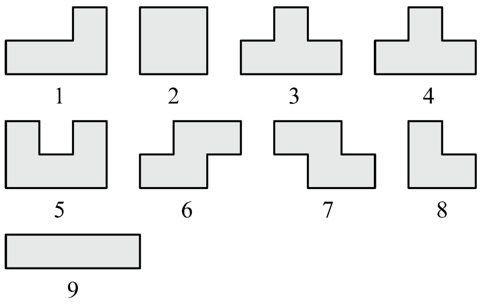
3.1 Sharing

Sharing methods convert fitness values according to the equation below [4].

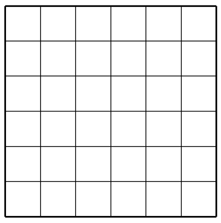
$$f_i' = \frac{f_i}{m_i} \tag{4}$$

where f_i is fitness value. m_i is given by

$$m_i = \sum_{j=1}^{N} sh(d_{ij}) \tag{5}$$



(a) Pieces



(b) Board

Fig. 1 Polyomino packing puzzle

N is the population size and d_{ij} is the distance between individual i and j. $sh(d_{ij})$ is defined by

$$sh(d_{ij}) = \begin{cases} 1 - \left(\dfrac{d_{ij}}{\sigma}\right)^{\alpha} & \text{if } d_{ij} < \sigma \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where σ and α are constants.

In sharing methods, f_i' is used instead of f_i for selection. That converts the fitness value of individual i much smaller if there exist many other individuals in the vicinity.

3.2 Clearing

In clearing methods [5], the individuals that exist within distance σ of individual i are considered to belong to the same group. The fitness values of the individuals belonging to the group are set to zero except the top κ individuals. Therefore the maintenance of multiple solutions is expected.

4 Problem setting

This paper takes up a polyomino packing puzzle shown in Fig. 1. It is a 9-piece puzzle of which the board has 6 rows and 6 columns. Each piece can be rotated every 90 degrees, whereas turning over is not allowed.

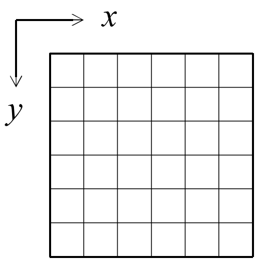


Fig. 2 Coordinate system for cells

5 Experimental setting

5.1 Genetic coding methods

The explorations are executed by two genetic coding methods as described below.

Method I

The positions of cells in the board are designated by coordinates (x, y), as shown in Fig. 2. In this case, three values, *i.e.*, rotation code, x and y coordinates, are stored in a chromosome per piece. The coordinates indicate the position of upper left part of the piece. The methods require 27 genes in a chromosome.

Method II

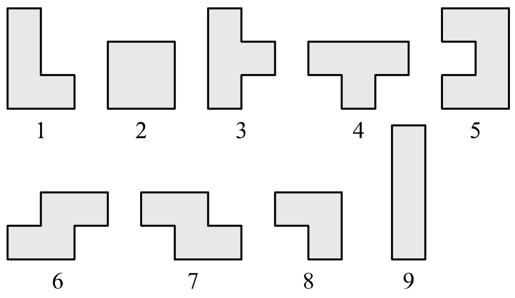
This paper proposes the genetic coding methods as follows.

Every cell in the board is numbered as shown in Fig. 3. We trace the cells in order of their numbers. When we come up with a cell not occupied by any pieces so far, we allocate to the cell the upper left part of the piece whose identifier is stored in the next gene. We repeat the above steps until all the pieces are allocated. For example, if genes in a chromosome describes the order of the pieces as (5, 3, 4, 9, 6, 7, 2, 1, 8) and their directions as illustrated in Fig. 4(a), the placement of each piece is shown in Fig. 4(b). The number of genes in a chromosome is 18.

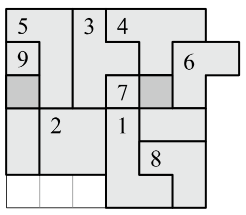
The number of combinations is less than that of method I. The search space is narrower. Therefore faster conversion to the optimal solutions is expected.

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

Fig. 3 Numbered cells in the board



(a) Direction of each piece



(b) Placement of the pieces

Fig. 4 Example of placement of pieces on a board

5.2 Definition of fitness function

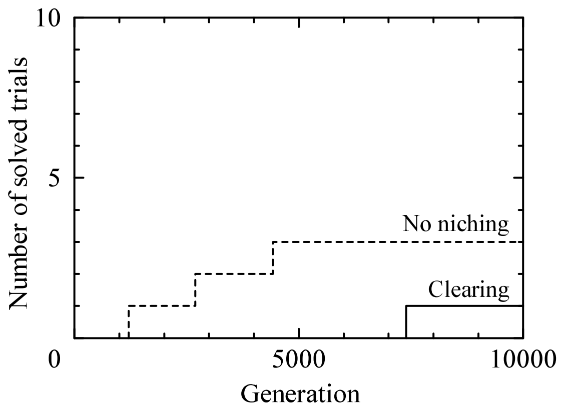
The total number of cells in the board which is not occupied by any pieces is defined as fitness function. That means the puzzle is solved when the fitness value reaches 36.

5.3 Parameter setting

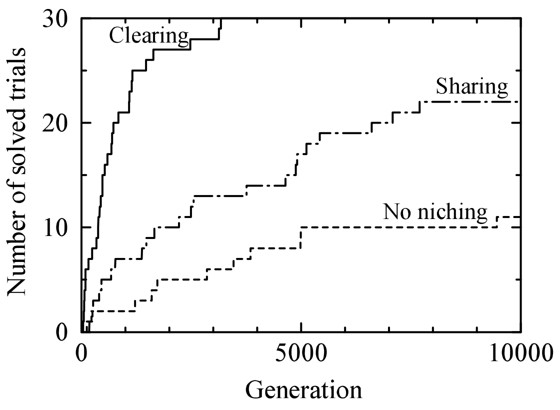
The GA parameters are set as follows: population size 100, crossover rate 0.5, mutation rate 0.04, number of elite individuals 3, and maximum number of generations 10000.

The niching parameters are as follows: $\sigma = 30$, and $\alpha = 1$ for sharing, $\kappa = 5$ for clearing.

The scaling parameter is set to $c = 2$ in equations (2) and (3).



(a) Method I



(b) Method II

Fig. 5 Transition of number of solved trials

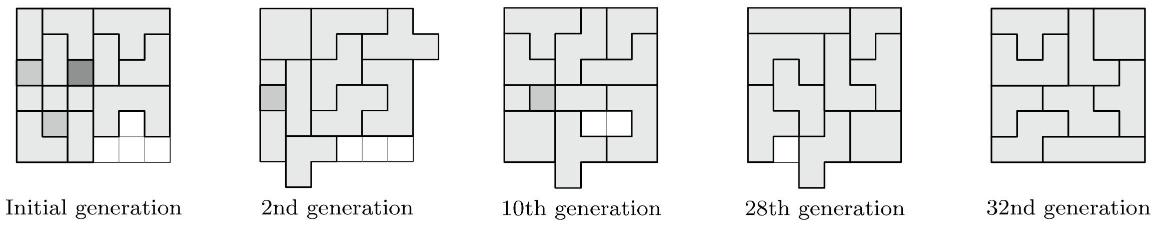


Fig. 6 Sample of process on piece placement

6 Results

The searches were conducted both by Method I and II. In addition, sharing and clearing methods were adopted. Each trial was carried out for 30 times.

Fig. 5(a) and (b) show the relationships between the number of the solved trials and the generation by Method I and II, respectively. Only at most 10% of the trials reached a solution by Method I. In this case, sharing methods did lead no trials to the optimal solution. Method II, however, improved search performance. Niching methods are proved to be extremely effective. Especially clearing method made all the trials solved within 3200th generation. On the contrary, niching is not effective in case of Method I. Fig. 6 shows the process of a trial by Method II, which has reached the optimal solution fastest of all the trials.

7 Conclusion

This paper has attempted to solve efficiently a polyomino packing puzzle using GA. It is considered difficult to form building blocks for this type of problem. Nevertheless the proposed genetic coding methods, that narrows the search space, improves the search performance. Also considering the results, it is possible that niching methods are more effective if the search space is narrower.

References

1) S. Coffin, and J. Slocum, "What's New in Polyomino Puzzles and Their Design," *Mathematical Properties of Sequences and Other Combinatorial Structures*, Springer US, pp. 113–119 (2003)

2) E. Demaine, and M. Demaine, "Jigsaw Puzzles, Edge Matching, and Polyomino Packing: Connections and Complexity," *Graphs and Combinatorics*, 23(s1), pp. 195–208 (2007)

3) S. Forrest, and M. Mitchell, "Relative Building-Block Fitness and the Building-Block Hypothesis", in Whitley, D, ed. FOGA 2, Morgan Kaufmann, San Mateo, CA.

4) B. Sareni B, and L. Krähenbühl, "Fitness Sharing and Niching Methods Revisited", *IEEE Transactions on Evolutionary Computation*, Vol.2, No.3, pp.97–106 (1998)

5) A. Pétrowski, "A Clearing Procedure as a Niching Method for Genetic Algorithms", *Proceedings of 1996 IEEE International Conference on Evolutionary Computation*, pp.798–803 (1996)