Trans. of the Society of Instrument
and Control Engineers
Vol.E-4, No.1, 1/8 (2005)

A New Approach to Rectangle Packing Problem Based on Stochastic Tabu Search

Yuji SHIGEHIRO*, Seiji KOSHIYAMA** and Tatsuya MASUDA*

A rectangle packing is a non-overlapping placement of given rectangles. The algorithm to optimize the rectangle packing can be required in a variety of application fields. However, since the optimization problem of rectangle packing is a combinatorial optimization problem and NP-hard, it is hard to solve it exactly in practical applications. To cope with this difficulty, a new coding scheme, sequence-pair, has been proposed, by means of which any possible packing can be represented. Tabu search is one of the powerful meta-heuristics, which can explore the solution space effectively in an intelligent manner. Thus a tabu search approach for rectangle packing problem based on sequence-pair should be powerful in practical applications, however any such approaches have not been reported. The purpose of this paper is to apply tabu search to rectangle packing based on sequence-pair, and evaluate this approach. First, rectangle packing problem is formulated, and the representation of each solution is defined. And then, the proposed method is described in detail, where first admissible move strategy and a concept of stochastic tabu restrictions are employed. The experimental result shows that the proposed method keeps providing good performance through the search process.

Key Words: tabu search, meta heuristics, rectangle packing, stochastic tabu restrictions

1. Introduction

A *rectangle packing* is a non-overlapping placement of given rectangles. The algorithm to optimize the rectangle packing can be required in a variety of application fields, such as VLSI layout design, facility layout problem, etc. However, since the optimization problem of rectangle packing is a combinatorial optimization problem and NP-hard[1], it is hard to solve it exactly in practical applications. Thus heuristic algorithms have been widely used.

Until now, a variety of approaches based on simulated annealing or genetic algorithm have been reported in the literature for rectangle packing. Most of them could be classified into two categories according to their coding schemes, namely the representation of each packing. In one scheme, a packing is defined by a tree structure (and in many cases, represented in terms of "Polish expression")[2)~4)], in the other scheme, a packing is defined by a permutation of rectangles[5)~7)]. However in each of these schemes, some kind of packing can not be represented, and consequently, we could never obtain such kind of packing using these approaches.

To cope with this difficulty, several new coding schemes, such as sequence-pair[1], BSG-structure[8], O-tree[9], and so on, have been proposed, by means of which any possible packing can be represented, and good packing can be obtained. Among them, the sequence-pair coding has been widely researched[10)~12)], since it can be implemented easily and effectively. It is also thought that the sequence-pair coding is fit to use in most of meta-heuristics[13].

Tabu search[14), 15)] is one of the powerful meta-heuristics, which can explore the solution space effectively in an intelligent manner by means of "flexible memory"[13]. A tabu search approach for rectangle packing problem based on sequence-pair should be powerful in practical applications, however any such approaches have not been reported. The purpose of this paper is to present a new approach for rectangle packing based on tabu search heuristic[16], and evaluate it.

In this paper, rectangle packing problem is formulated at first. Next, the representation of each solution is defined, and then tabu search is applied. A concept of stochastic tabu restrictions[14] is also introduced. In addition, a part of experimental results is shown in order to demonstrate the effectiveness of the proposed method.

2. Rectangle packing problem

A packing is a non-overlapping placement of given rectangles. A rectangle packing problem is to seek a packing with minimum (bounding) area[1]. The problem can be formulated as follows:

Definition (Rectangle packing problem). Assume there exist n rectangles r_i $(i = 1, \ldots, n)$. Let w_i and h_i be width and height of r_i, and (x_i, y_i) be coordinate of

* Department of Electrical and Electronic Systems Engineering, Osaka Institute of Technology, Osaka
** SANYO Electric Co., Ltd., Osaka

the center of r_i. Given the width w_i and height h_i of each rectangle r_i $(i = 1, \ldots, n)$, seek a set of coordinates (x_i, y_i) of all given rectangles r_i $(i = 1, \ldots, n)$ which minimizes the area s of the bounding rectangle of all given rectangles

$$s = \left\{ \max_i \left(x_i + \frac{w_i}{2} \right) - \min_i \left(x_i - \frac{w_i}{2} \right) \right\} \times \left\{ \max_i \left(y_i + \frac{h_i}{2} \right) - \min_i \left(y_i - \frac{h_i}{2} \right) \right\} \qquad (1)$$

under the following "non-overlapping" condition

$$|x_i - x_j| \geq \frac{w_i + w_j}{2} \quad \text{or} \quad |y_i - y_j| \geq \frac{h_i + h_j}{2}$$
$$\text{for any } i, j \quad (1 \leq i \leq n, 1 \leq j \leq n, i \neq j). \qquad (2)$$

3. Tabu search for rectangle packing

Tabu search [14), 15)] is one of the powerful metaheuristics, where the optimal solution is searched in the solution space, moving from one trial solution to another, in the same way as local neighborhood search. In tabu search, as distinct from local neighborhood search, the exploration can be continued even at a locally optimal solution. In order not to fall back into a locally optimal solution previously emerged, some of possible moves are regarded as forbidden, on the basis of historical information from the search process.

This historical information is stored in a queue, called *tabu list*, and the element of tabu list is called *attribute*. The length of the tabu list is limited, and this limit length is called *tabu length*. When moving from one trial solution to another in the search process, a set of attributes corresponding to the move is added to the tabu list, and the oldest set of attributes in the tabu list is discarded when the length of the tabu list exceeds the tabu length.

Now, consider how to apply tabu search to rectangle packing problem.

3.1 Representation of solution

In this paper, each solution of rectangle packing problem is represented by a pair of permutations of rectangles called sequence-pair, and decoded to a corresponding packing by means of the procedure described in 10). The two sequences of sequence-pair are denoted by Γ_+ and Γ_-, respectively.

The decoded packing by 10) is either the same as one obtained by the procedure described in 1) or more compacted packing obtained by reducing redundant constraints between rectangles taken into account in 1). Refer to 10) for more detailed discussion.

For example, given a set of rectangles shown in **Fig. 1** (a) and a sequence-pair

$$(\Gamma_+, \Gamma_-) = ((r_4, r_6, r_5, r_1, r_2, r_3, r_7), (r_2, r_7, r_5, r_3, r_4, r_1, r_6)) \qquad (3)$$

the decoding process is illustrated in Fig. 1 (b)–(e).

The bold line L in Fig. 1 shows the boundary of the region where any (large) rectangle can be placed without overlap. In the decoding process, each rectangle is placed onto L, one after another, based on the following guidelines:

(1) Rectangles are placed in the same order as Γ_-.

(2) All rectangles on L are always aligned in the same order as Γ_+, from upper-left to lower-right.

In Fig. 1, r_2 is placed at first and then $r_7, r_5, \ldots$, since r_2 is at first and then $r_7, r_5, \ldots$ on Γ_-. In Fig. 1 (d), r_1 is placed between r_5 and r_3, since r_1 is between r_5 and r_3 on Γ_+.

In the same way, any sequence-pair can be decoded to a corresponding packing.

3.2 Outline of the proposed method

We have applied tabu search to rectangle packing problem as outlined in **Fig. 2**. Procedure TABUPACKING is the main routine, which repeatedly calls procedure NEXTSOLUTION at line 9 for iterative improvement.

x is the trial solution, which is represented by a sequence-pair as described above. At first, x is initialized as a random sequence-pair at line 5. And then, an adjacent solution of x is selected through lines from 13 to 34, and x is updated with it at line 37, in order to move from one solution to another, in the same way as local neighborhood search.

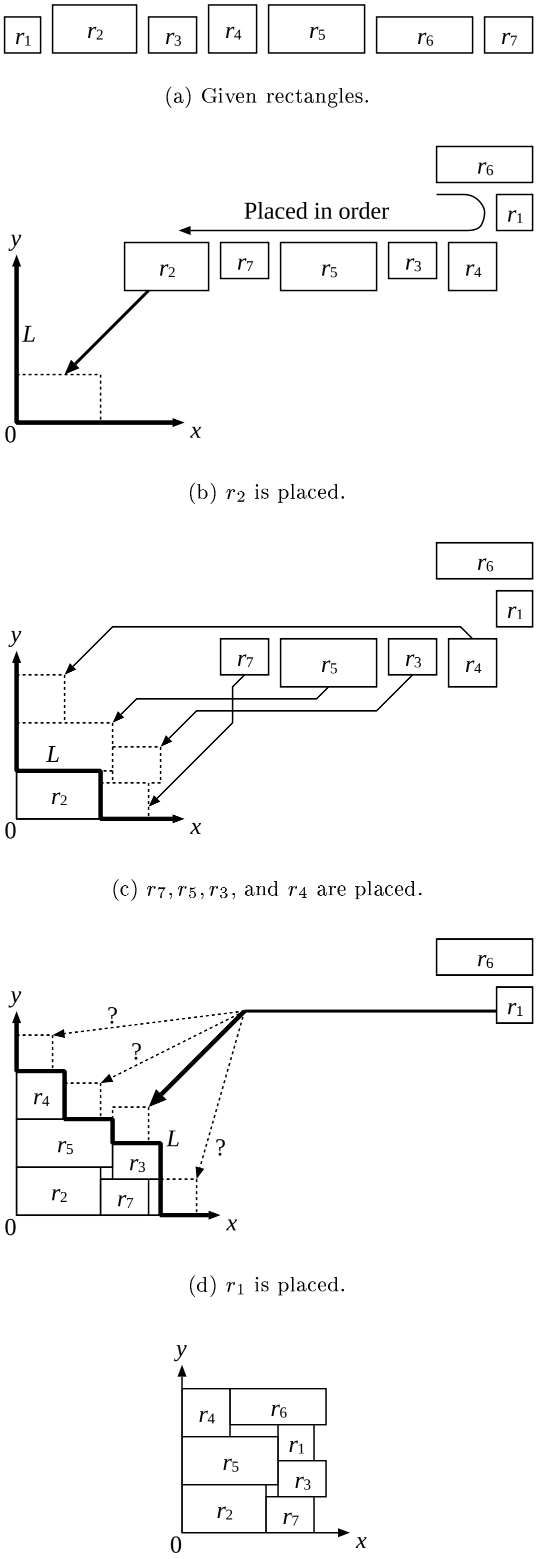
Whenever a new sequence-pair is generated, it is decoded to a corresponding packing as described above, and the area of the packing is evaluated, as shown at line 5, 6, 16 and 17. The area of the packing is stored, and referred when it is required. The notation $f(x)$ means the area of the packing decoded from x.

n_e is the number of evaluations of solutions. Initially, n_e is cleared at line 4. Whenever a sequence-pair is evaluated, n_e is incremented, at line 6 and 17. When n_e is larger than a given number MAXEVAL, the algorithm is terminated, as seen at line 8.

x_b keeps the best solution through the search process, which is initialized at line 7, and updated at line 18 and 20.

3.3 Strategy of move

As for move, not "best admissible move strategy" but "first admissible move strategy" is employed, as follows. In the literature [17)], it is reported that the latter is more effective than the former.



(a) Given rectangles.

(b) r_2 is placed.

(c) r_7, r_5, r_3, and r_4 are placed.

(d) r_1 is placed.

(e) Result of decoding.

Fig. 1 Process of decoding based on 10).

```
 1: procedure TABUPACKING:
 2:   begin
 3:     initialize tabu list $T$;
 4:     $n_e \leftarrow 0$;
 5:     let $x$ be an initial solution;
 6:     evaluate $x$; $n_e \leftarrow n_e + 1$;
 7:     $x_b \leftarrow x$;
 8:     while $n_e <$ MAXEVAL do
 9:       NEXTSOLUTION($x, T, x_b, n_e$);
10:   end.
11: procedure NEXTSOLUTION($x, T, x_b, n_e$):
12:   begin
13:     $x_g \leftarrow$ nil;
14:     for $i \leftarrow 1$ to MAXNEI do
15:       begin
16:         let $x_a$ be an adjacent solution of $x$;
17:         evaluate $x_a$; $n_e \leftarrow n_e + 1$;
18:         if $f(x_a) < f(x_b)$ then
19:           begin
20:             $x_b \leftarrow x_a$;
21:             goto MOVE;
22:           end
23:         if the move from $x$ to $x_a$ is not forbidden
                by the tabu restrictions specified by
                $T$ then
24:           begin
25:             if $f(x_a) \leq f(x)$ then
26:               goto MOVE;
27:             if $x_g =$ nil or $f(x_a) < f(x_g)$ then
28:               $x_g \leftarrow x_a$;
29:           end
30:       end
31:     if $x_g =$ nil then
32:       goto NOMOVE;
33:     else
34:       $x_a \leftarrow x_g$;
35: label MOVE:
36:     update $T$ with a set of attributes corresponding
            to the move from $x$ to $x_a$;
37:     $x \leftarrow x_a$;
38: label NOMOVE:
39:   end.
```

Fig. 2 Outline of the proposed method.

An adjacent solution of x is selected randomly and assigned to x_a, at line 16. At first, the tabu restrictions are considered. Only if the move to x_a is not forbidden by the tabu restrictions, the move can be accepted, as described at line 23. Then, the area of x_a is examined, and if x_a is better than x, the move is chosen, without evaluating the other adjacent solutions, as described at line 25 and 26.

In the case when x_a is better than x_b (the best solution previously emerged), *aspiration level*[14] is considered to be attained, and the move is chosen whether the move is forbidden or not, as described at line 18 and 21.

Only in these two cases, the move to this adjacent solution is chosen. In the other cases, this adjacent solution is discarded. Another adjacent solution is selected randomly, and tried again. If no move is chosen after a given number MAXNEI of adjacent solutions are tried, we think (from the standpoint of efficiency) that x is a locally optimal solution, and the best of adjacent solutions already tried and not forbidden by the tabu restrictions is chosen. The MAXNEI adjacent solutions are tried in the loop specified at line 14. The best adjacent solution is kept in x_g through lines 13, 27, and 28, and chosen at line 34.

3.4 Generation of adjacent solution

In the proposed method, an adjacent solution x_a of the trial solution x is generated by means of one of the following three operations randomly selected:

(1) Swap two rectangles on Γ_+ of x.

(2) Swap two rectangles on Γ_- of x.

(3) Swap two rectangles on both Γ_+ and Γ_- of x.

In each operation, the two rectangles are selected randomly.

As is illustrated in **Fig. 3**, these operations correspond to the following transformations of the packing, respectively:

(1) Move two rectangles toward upper-left and lower-right.

(2) Move two rectangles toward upper-right and lower-left.

(3) Swap two rectangles.

3.5 Stochastic tabu restrictions

In applying tabu search heuristic, the tabu restrictions greatly affect the efficiency of the search process. After careful consideration and some computational experiments, the following procedure is constructed:

Let $r_a(m)$ and $r_b(m)$ be two rectangles which are swapped in the move m. Let $s(m)$ be sequence(s) (Γ_+, Γ_-, or both) where rectangles are swapped in the move m. Assume the tabu length is TL, and attributes of the last TL moves are stored in the tabu list. Denote the last TL moves by $m(1), m(2), \ldots, m(\mathrm{TL})$ from newest to oldest (ex. $m(1)$ denotes the last move).

When a new adjacent solution x_a is generated, the corresponding move m_a is tested with each element of the tabu list. If both of the following conditions:

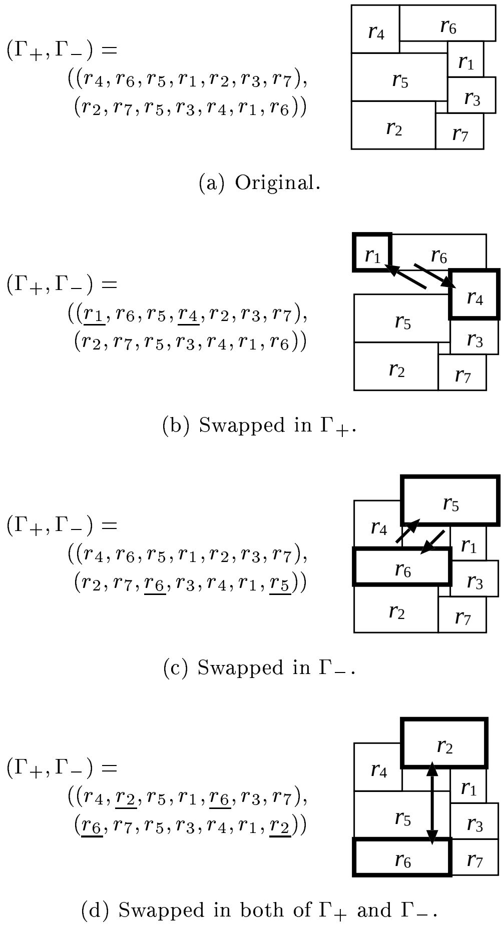
(1) $s(m_a)$ is the same as $s(m(i))$

(2) $r_a(m_a)$ or $r_b(m_a)$ is the same as $r_a(m(i))$ or $r_b(m(i))$

are satisfied, m_a is rejected with probability

$$p(i) = \frac{\mathrm{TL} - i + 1}{\mathrm{TL}}. \quad (4)$$

This procedure can be described more formally in **Fig. 4**.



$(\Gamma_+, \Gamma_-) =$
$((r_4, r_6, r_5, r_1, r_2, r_3, r_7),$
$(r_2, r_7, r_5, r_3, r_4, r_1, r_6))$

(a) Original.

$(\Gamma_+, \Gamma_-) =$
$((\underline{r_1}, r_6, r_5, \underline{r_4}, r_2, r_3, r_7),$
$(r_2, r_7, r_5, r_3, r_4, r_1, r_6))$

(b) Swapped in Γ_+.

$(\Gamma_+, \Gamma_-) =$
$((r_4, r_6, r_5, r_1, r_2, r_3, r_7),$
$(r_2, r_7, \underline{r_6}, r_3, r_4, r_1, \underline{r_5}))$

(c) Swapped in Γ_-.

$(\Gamma_+, \Gamma_-) =$
$((r_4, \underline{r_2}, r_5, r_1, \underline{r_6}, r_3, r_7),$
$(\underline{r_6}, r_7, r_5, r_3, r_4, r_1, \underline{r_2}))$

(d) Swapped in both of Γ_+ and Γ_-.

Fig. 3 Operations to generate adjacent solutions.

```
procedure CANMOVE($m_a$):
  begin
    for $i \leftarrow 1$ to TL do
      if $s(m_a) = s(m(i))$ then
        if $r_a(m_a) = r_a(m(i))$
            or $r_a(m_a) = r_b(m(i))$
            or $r_b(m_a) = r_a(m(i))$
            or $r_b(m_a) = r_b(m(i))$ then
          begin
            let $r$ be a random number in $[0, 1)$;
            $p \leftarrow \frac{\mathrm{TL} - i + 1}{\mathrm{TL}}$;
            if $r < p$ then
              begin
                CANMOVE ← "no";
                goto EXIT;
              end
          end
    CANMOVE ← "yes";
label EXIT:
  end.
```

Fig. 4 Procedure to check tabu restrictions.

4. Experimental results

The proposed method has been programmed in C++ language, and run on a VT-Alpha600H/U (18.0

SPECint95) workstation. We have also implemented the following 4 methods in order to evaluate the proposed method. In all of them, adjacent solutions are generated in the same way as described in 3. 4.

LB (multi-start Local search with Best admissible move strategy)

In this method, all adjacent solutions are generated for each trial solution x, and evaluated in random order. Then, the best solution x_b of them is selected as the candidate of next trial solution. In the case when x_b is not better than x (that is, x is a locally optimal solution), the exploration is restarted from another trial solution randomly generated.

LF (multi-start Local search with First admissible move strategy)

In this method, all adjacent solutions are generated for each trial solution x, and evaluated in random order, in the same way as LB. However, in case when an adjacent solution x_a is evaluated and it is better than x, x_a is selected as the next trial solution immediately, without evaluating the other adjacent solutions. If no solution is selected after all adjacent solutions are evaluated (that is, x is a locally optimal solution), the exploration is restarted from another trial solution randomly generated.

SA (Simulated Annealing)

In this method[10], the geometric cooling is employed for cooling schedule. Parameters, such as initial temperature and cooling ratio, are decided based on exhaustive preliminary computational experiments.

TS (Tabu Search without stochastic tabu restrictions)

A simplified version of the proposed method has been also programmed, which is the same as the proposed method except that the tabu restrictions are deterministic ("$p(i) = 1$" is adopted instead of Eq. (4)).

We applied all of above 5 methods to MCNC benchmark "ami49"[18], which is a problem in VLSI layout design including 49 building blocks. In all computational experiments, the number of evaluations (MAXEVAL) was limited to 1,000,000. Many preliminary computational experiments were carried out in order to decide parameters required in SA, TS, and the proposed method. The parameters for the proposed method was decided as follows:

MAXNEI : 200
TL : 5

Each method was executed 100 times for "ami49". The CPU time of a trial of the proposed method in our implementation is about 8 minutes. The average behavior over 100 trials of each method is summarized in **Fig. 5**. For each number of evaluations (indicated at horizontal axis in Fig. 5), the area of the best solution obtained in the trial *until that moment* was kept, and the average of them over 100 trials is plotted in Fig. 5, for each method.

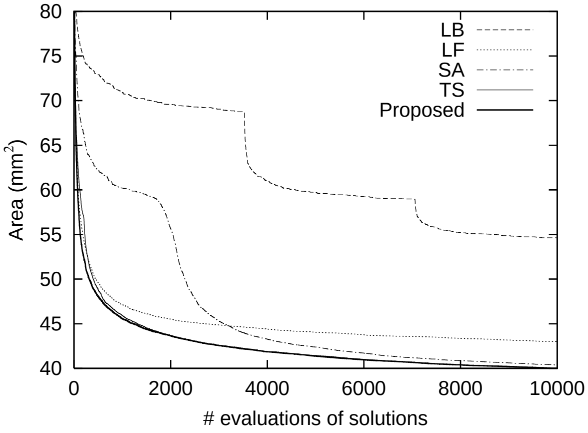


Fig. 5 Behaviors of methods.

At first, the following should be noticed hear:

(1) In LB, the first move from the initial solution to another solution is carried out *after* all of the adjacent solutions of the initial solution are evaluated. The number of adjacent solutions is 3,528 (${}_{49}C_2 \times 3$).

In each of 100 trials in the experiment, at first, 3,528 adjacent solutions of the *initial solution* were evaluated. In the meantime, solutions which were better than ones previously emerged in the trial were found (evaluated), and the curve of LB in Fig. 5 is plotted downward.

Just after 3,528 evaluations, the move from the initial solution to another was carried out, and after that, adjacent solutions of the *new solution* were evaluated. Thus, the curve of LB in Fig. 5 has a corner at 3,528 evaluations.

(2) In each of 100 trials for SA in the experiment, at first, the temperature was high, and the search process was rather in "random walk" fashion. In the meantime, solutions which were better than ones previously emerged in the trial were found, and the curve of SA in Fig. 5 is plotted downward.

As the temperature fell and gradually the move toward a bad solution became unaccepted, better solutions found one after another, and the curve of SA in Fig. 5 has a steep slope at about 1,800–2,500 evaluations.

From Fig. 5 and these notices, the performances in the early stage of the search processes are summarized as follows:

(1) The performance in the early stage of the search process of LB is obviously inferior to the other methods. For "ami49", the number of adjacent solutions is large, and the evaluations of all of them degrade the performance.

(2) As for SA, the performance in the early stage (more specifically, less than about 3,000 evaluations) is not good compared to LF, TS, and the proposed method. If the temperature is lowered rapidly, the performance in the early stage may be improved however the solution obtained after MAXEVAL (1,000,000) evaluations should be worse. For SA, the "random walk" phase is essential and degradation of the performance in the early stage is unavoidable.

(3) LF, TS, and the proposed method provided better performance than LB and SA at less than about 3,000 evaluations. However, the performance of LF gradually went down.

In each trial of LF, the exploration was trapped at a locally optimal solution and restarted. The first restart in each trial in the experiments was at about 4,000–27,000 evaluations, however, TS and the proposed method outperform LF even at 1,000 evaluations. The reason should be that the exploration of TS or the proposed method is more diversified because of tabu restrictions.

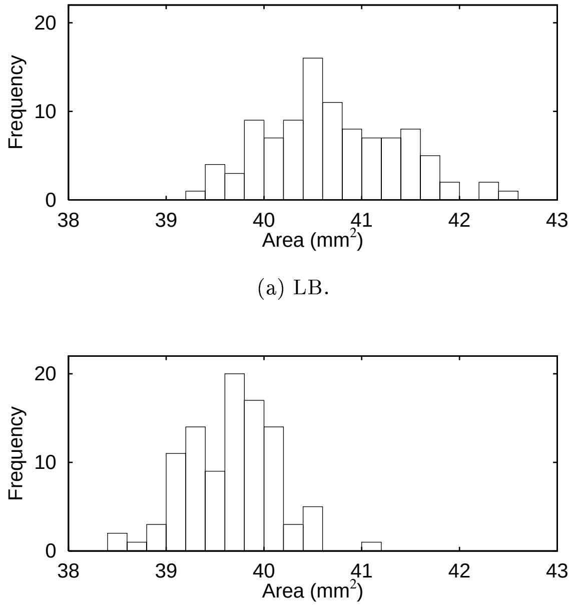
(4) At about 0–1,000 evaluations, the proposed method (bold line in Fig. 5) was better than TS (solid thin line in Fig. 5). Stochastic tabu restrictions improve the performance in the early stage.

The performance of TS and the proposed method in the early stage of the search process is good. This seems to be preferable for practical applications where it takes a good amount of time to evaluate each solution. For example, in the layout design of VLSI's, the objective function may depend on not only the area of the bounding rectangle of all given rectangles (in the case of our computational experiments) but also the result of global or detailed routing, circuit's performance evaluation, and so on.

Table 1, **Fig. 6**, and **Fig. 7** show the area of the best solution obtained in each trial. It should be noticed here that horizontal scales of graphs in Fig. 6 and Fig. 7 are different.

Table 1 Statistics of 100 trials.

	Area (mm^2)			
	Average	Standard deviation	Best	Worst
LB	40.70	0.67	39.21	42.43
LF	39.66	0.46	38.42	41.01
SA	36.68	0.24	36.18	37.66
TS	36.71	0.22	36.24	37.54
Proposed	36.66	0.20	36.26	37.24



(a) LB.

(b) LF.

Fig. 6 Frequency distribution of 100 trials (1).

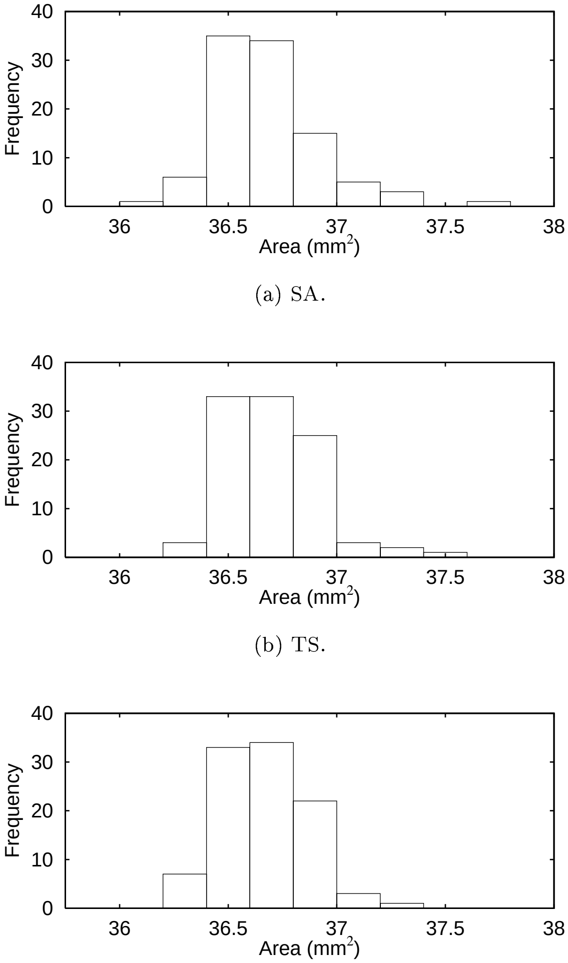
The results are summarized as follows:

(1) LB and LF are much worse than SA, TS, and the proposed method. The reason should be that the exploration is stopped at a locally optimal solution, and solutions around it cannot be explored, in LB and LF.

(2) The areas of the solutions obtained by SA, TS, and the proposed method are almost same. As for the worst areas, the proposed method is slightly smaller, and SA is slightly larger. If the execution time of the program is very long as in the case described above, and we can run the program only once, the proposed method should be preferable.

Figure 8 shows the rectangle packing of ami49 obtained by the proposed method. As can be seen from the figure, a very tight packing can be obtained.

We also generated randomly two large problems "rand1000" and "rand3000" including 1,000 and 3,000 rectangles respectively, and applied SA, TS, and the proposed method. The numbers of evaluations were limited to 10,000 and 1,000 for "rand1000" and "rand3000" respectively, so that CPU time of each execution of the proposed method would be about 15 minutes. The cooling schedule of SA was adjusted to each problem, on the other hand, the same parameters as for "ami49" were used



(a) SA.

(b) TS.

(c) The proposed method.

Fig. 7 Frequency distribution of 100 trials (2).

in TS and the proposed method. Each method was executed 50 times, and **Table 2** shows the statistics of the area of the best solution obtained in each trial. The average behaviors over 50 trials of each method applied to "rand3000" are also summarized in **Fig. 9**. The proposed method provided better performance also in this experiments.

Table 2 Statistics of 50 trials for large problems.

	Area			
	Average	Standard deviation	Best	Worst
rand1000				
SA	42.63	0.90	40.74	45.10
TS	42.37	1.05	40.44	45.04
Proposed	42.26	0.97	40.18	44.62
rand3000				
SA	196.1	7.8	180.8	213.3
TS	191.7	6.8	178.4	208.4
Proposed	190.7	6.5	174.3	205.5

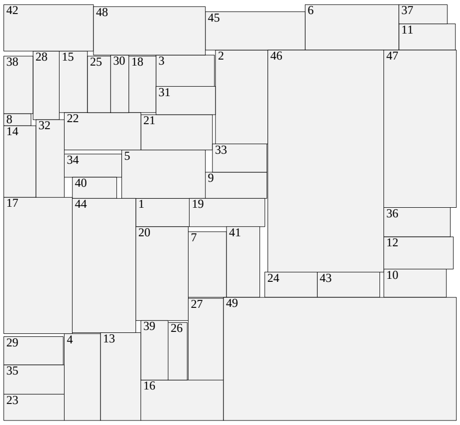


Fig. 8 A rectangle packing obtained by the proposed method.

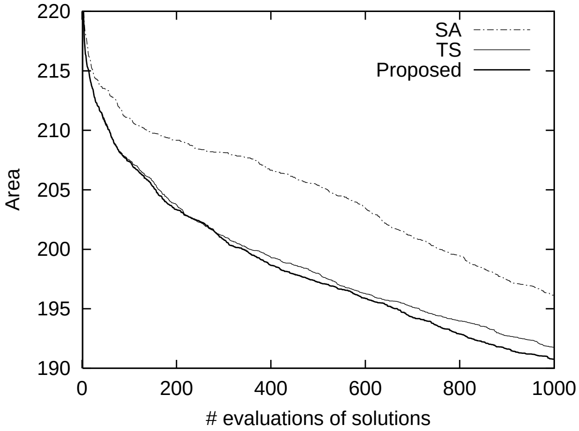


Fig. 9 Behaviors of methods applied to "rand3000".

5. Concluding remarks

In this paper, a new approach for the rectangle packing problem based on tabu search heuristic has been presented. Each solution is defined by means of a sequence-pair, and then any possible solution can be explored. As far as the authors know, any approaches based on tabu search utilizing sequence-pair have not been reported previously. A concept of stochastic tabu restrictions has been also introduced.

The experimental result shows that tabu search provides better performance than the other methods in the early stage of the search process, and also provides good results comparable to simulated annealing at the end of the search process. Stochastic tabu restrictions improve the performance of tabu search.

References

1) H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani: VLSI module placement based on rectangle-packing by the sequence-pair, *IEEE Trans. Computer Aided Design*, **15**-12, 1518/1524 (1996)

2) T. Yamanouchi, K. Tamakashi, and T. Kambe: Hybrid floorplannig based on partial clustering and module

restructuring, *Proc. Design Automation Conf.*, 478/483 (1996)

3) V. Schnecke and O. Vornberger: Hybrid genetic algorithm for constrained placement problem, *IEEE Trans. Evolutionary Computation*, **1**-4, 266/277 (1997)

4) W. Sessomboon, H. Muramatsu, T. Irohara, and K. Yoshimoto: Integrated approach for solving cell formation and layout problems using genetic algorithm, *Trans. Japan Society of Mechanical Engineers C*, **64**-617, 362/369 (1998) (in Japanese).

5) W. Sessomboon, H. Nakano, T. Irohara, and K. Yoshimoto: A technique to solve dynamic layout problem with fixed and rectangular departments, *Trans. Japan Society of Mechanical Engineers C*, **63**-615, 4050/4056 (1997) (in Japanese).

6) N. Nakaya, A. Kanasugi, and K. Kondo: A placement method for arbitrary shape blocks, *J. Japan Institute of Electronics Packaging*, **1**, 6, 476/482 (1998) (in Japanese).

7) Y. Shirai and N. Matsumoto: Performance evaluation of ES type genetic algorithms for solving block layout problems with floor constraints, *Trans. Japan Society of Mechanical Engineers C*, **65**-634, 2456/2464 (1999) (in Japanese).

8) S. Nakatake, K. Fujiyoshi, H. Murata, and Y. Kajitani: Module packing based on the BSG-structure and IC layout applications, *IEEE Trans. Computer Aided Design*, **17**-6, 519/530 (1998)

9) P. N. Guo, T. Takahashi, C. K. Cheng, and T. Yoshimura: Floorplanning using a tree representation, *IEEE Trans. Computer Aided Design*, **20**-2, 281/289 (2001)

10) A. Nagao, T. Sawa, Y. Shigehiro, I. Shirakawa, and T. Kambe: A new approach to rectangle-packing, *Trans. Institute of Electronics, Information and Communication Engineers A*, **J81-A**-10, 1362/1371 (1998) (in Japanese).

11) K. Hatta, S. Wakabayashi, and T. Koide: Solving the rectangular packing problem by an adaptive GA based on sequence-pair, *Proc. Asia South Pacific Design Automation Conf.*, 181/184 (1999)

12) Y. Shigehiro, S. Yamaguchi, M. Inoue, and T. Masuda: A genetic algorithm based on sequence-pair for floorplan design, *Trans. Institute of Electrical Engineers of Japan*, **121-C**-3, 601/607 (2001) (in Japanese).

13) C. R. Reeves: Modern Heuristic Techniques for Combinatorial Problems, Halsted Press (1993)

14) F. Glover: Tabu search-part I, *ORSA J. Computing*, **1**-3, 190/206 (1989)

15) F. Glover: Tabu search-part II, *ORSA J. Computing*, **2**-1, 4/32 (1990)

16) Y. Shigehiro, S. Koshiyama, and T. Masuda: Stochastic tabu search for rectangle packing, *Proc. Intl. Conf. on Systems, Man and Cybernetics*, 2753/2758 (2001)

17) M. Yagiura and T. Ibaraki: On metaheuristic algorithms for combinatorial optimization problems, *Trans. Institute of Electronics, Information and Communication Engineers D-I*, **J83-D-I**-1, 3/25 (2000) (in Japanese).

18) http://www.cbl.ncsu.edu/pub/Benchmark_dirs/LayoutSynth90/

Yuji **Shigehiro** (Member)

Yuji Shigehiro received the B.E., M.E., and D.E. degrees in electronic engineering from Osaka University, Japan, in 1987, 1989, and 1994, respectively. From 1992 to 1998, he was with the Department of Information Systems Engineering, Osaka University. Since 1998, he has been with the Department of Electrical and Electronic Systems Engineering, Osaka Institute of Technology, Japan, where he is currently an Associate Professor. His research interests include layout design automation of large scale integrated circuits, meta-heuristics, and decentralized autonomous systems.

Seiji **Koshiyama**

Seiji Koshiyama received the B.E. and M.E. degrees in electrical engineering from Osaka Institute of Technology, Japan, in 1999 and 2001, respectively. In 2001, he joined the SANYO Electric Co., Ltd., Japan, as a researcher. His research interests include layout design automation of large scale integrated circuits and meta-heuristics.

Tatsuya **Masuda** (Member)

Tatsuya Masuda received the B.E., M.E., and D.E. degrees in electrical engineering from Osaka University, Japan, in 1973, 1975, and 1981, respectively. From 1981 to 1987, he was with the Department of Electrical Engineering, Osaka University. Since 1987, he has been with the Department of Electrical and Electronic Systems Engineering, Osaka Institute of Technology, Japan, where he is a Professor. His research interests include neural networks, fuzzy systems, evolutionary computation, metaheuristics, and artificial life.

Reprinted from Trans. of the SICE
Vol. 40 No. 7 747/754 2004