CHAPTER 24

Multi-Objective Rectangular Packing Problem

Shinya Watanabe
Department of Human and Computational Intelligence,
Ritsumeikan University,
1-1-1 Nojihigashi, Kusatsu, Shiga 525-8577, Japan.
E-mail: sin@sys.ci.ritsumei.ac.jp

Tomoyuki Hiroyasu
Doshisha University Department of Engineering
1-3 Tatara Miyakodani,Kyo-tanabe, Kyoto, 610-0321, JAPAN
E-mail: tomo@is.doshisha.ac.jp

This chapter describes an implementation of a Multi-Objective Genetic Algorithm (MOGA) for the Multi-Objective Rectangular Packing Problem (RP). RP is a well-known discrete combinatorial optimization problem arising in many applications, such as a floor-planning problem in the LSI problem, truck packing problem, etc. Over the last 20 years, Evolutionary Algorithms (EAs), including Genetic Algorithms (GA), have been applied to RP, as EAs are adapted for pattern generation. On the other hand, many cases of RP have become multi-objective optimization problems. For example, floor-planning problems should take care of the minimum layout area, the minimum length of wires, etc. Therefore, RP is a very important problem as an application of MOGA. In this chapter, we describe the application of MOGA to Multi-Objective RP. We treat RP as two objective optimization problems to archive several critical layout patterns, which have different aspect ratios of packing area. We used the Neighborhood Cultivation GA (NCGA) as a MOGA algorithm. NCGA includes not only the mechanisms of effective algorithms, such as NSGA-II and SPEA2, but also the mechanism of neighborhood crossover. The results were compared to those obtained using other methods. Through numerical examples, we found that MOGA is a very effective method for RP. Especially, NCGA can provide the best solutions as compared to other methods.

1. Introduction

In this chapter, we describe the implementation of a Multi-Objective Genetic Algorithm (MOGA) for the Multi-Objective Rectangular Packing(RP). RP is a well-known discrete combinatorial optimization problem arising in many applications, such as the VLSI layout problem[2,4,10,12,15,16],the truck packing problem [14], etc. As RP is a well-known NP-hard and discrete problem, good heuristic methods such as Genetic Algorithms (GA) or Simulated Annealing (SA), are generally applied.

The VLSI layout problem is one of the most important RP because there are many VLSI layout problems such as chip floor planning, standard cell, macro cell digital placement, and analog placement which have the same goal of optimally packing arbitrarily sized blocks. In addition, layout complexity is becoming an important design consideration as VLSI device integration is doubling every two to three years. In addition, a floor-layout problem is essentially a multi-objective optimization problem involving the minimum layout area, the minimum length of wires, the minimum overlapping area, etc. Therefore, RP is a very important problem as an application of a MOGA.

As the variety of packing is infinite, the important key for successful optimization is the introduction of a finite solution space that includes an optimal solution. We used a sequent-pair to represent the solution of rectangular packing. Sequence-pair schemes can represent not only slicing structures but also non-slicing structures.

In this chapter, we describe the application of MOGA to Multi-Objective RP. We treat the RP as two objective optimization problems to achieve several critical layout patterns, which have different aspect ratios of packing area. We used Neighborhood Cultivation GA (NCGA) as a MOGA algorithm [17]. NCGA includes not only the mechanisms of effective algorithms, such as NSGA-II and SPEA2, but also the mechanism of neighborhood crossover. This model can be used to derive good nondominated solutions in typical multi-objective optimization test problems. The results were compared to those obtained with other methods: SPEA2, NSGA-II, and non-NCGA (NCGA without neighborhood crossover). Through numerical examples, we found that MOGA was a very effective method for RP, because several good solutions were found with small iterations in one trial. NCGA obtained the best solutions with a small area of layout as compared to the other methods.

Just after this section, we introduce the formulation of RP(Section 2),

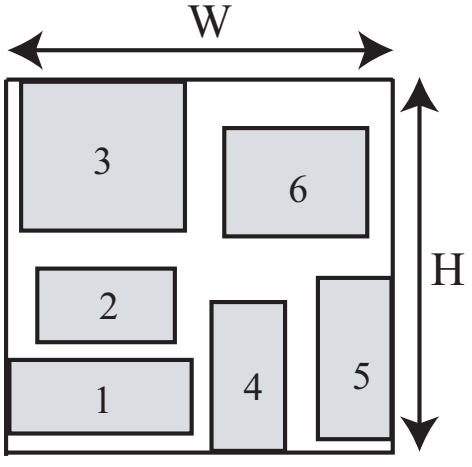


Fig. 1. Example of placement.

followed by a discussion of the application of GA for RP(Section 3). Section 4 introduces our proposed NCGA. Finally, Section 5 presents the results of the experiments for test data.

2. Formulation of Layout Problems

Many layout problems can be treated as rectangular packing problems(RP) in the real world. RP involves the placement of a given set of rectangular blocks of arbitrary size without overlap on a plane within a rectangle of minimum area, and is a well-known discrete combinatorial optimization problem in many applications, such as VLSI layout problems [10,11,12], the truck packing problems [14], etc.

Thus, it is a difficult and time-consuming problem to solve RP by computer simulation. In RP, the number of possible placements of rectangles increases exponentially with an increasing number of rectangles[10].

2.1. *Definition of RP*

Here, we would like to define RP. Let M be a set of m, rectangular blocks (blocks) of fixed orientations, whose heights and widths are given in real numbers. Packing of M requires placement of the blocks with no overlap. This problem is NP-hard, and therefore good heuristics are generally required. A packing example is shown in Fig.1.

As the heights and widths of blocks are real numbers, RP is not simply

a combinatorial problem.

2.2. *Multi-objective RP*

There have been a number of previous studies of multi-objective RP, for example in structural synthesis of cell-based VLSI circuits [1], placement of power electronic devices on liquid-cooled heat sinks[6], and the truck packing problem [14], etc.

In this chapter, we treat the RP as a bi-objective optimization. This multi-objective RP aims to minimize not only the packing area but also both the width and height of the packing area. In this formulation, we can obtain various Pareto solutions that have different aspect ratios by performing a single search. Therefore, a decision maker can select the aspect ratio of the packing area.

Next, we will describe the formulation of the multi-objective RP adopted in this chapter.

$$\begin{aligned} &\min f_1(x) = \text{width of packing area of blocks} \\ &\min f_2(x) = \text{length of packing area of blocks} \end{aligned}$$

These two objectives have tradeoff relations with each other.

3. Genetic Layout Optimization

GAs have been applied to various aspects of digital VLSI design. Examples include cell placement (layout)[4,10,11,12,15,16], channel routing[9], test pattern generation[13], etc.

There are two key issues in the use of a GA for RP.

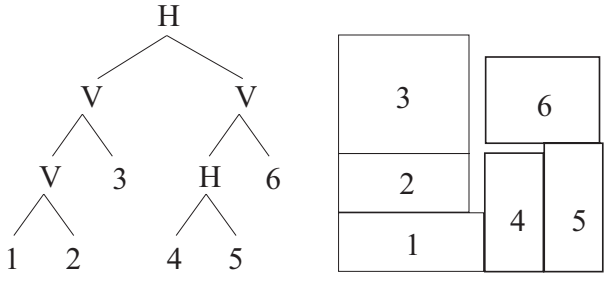
- Representation of individuals
- GA operators

These issues have a strong influence on the search ability of the GA. If these points are not carefully considered, it is not possible to obtain good results in real time.

The following sections describe these considerations in some detail.

3.1. *Representations*

There are two distinct spatial representations of placement configurations[2]. The first is the so-called flat or absolute representation that has been used



Polish Expression:(((12V)3V)((45H)6H)H)

Fig. 2. Slicing structure and Polish expression.

in earlier studies[4]. In this method, block positions are defined in terms of absolute coordinates on a gridless plane.

As this method allows blocks to overlap in possibly illegal ways, this method uses a weighted penalty cost term that is associated with infeasible overlaps, and this penalty must be driven to zero in the optimization process. However, the total overlap in the final placement solution is not necessarily zero. In addition, the weighted penalty cost must be carefully instituted: if it is too small, the blocks may tend to collapse, while if it is too large we may not obtain good search ability. Moreover, the packing variety of this method is infinite.

In contrast to the flat representation, in the topological representations, block positions are specified in a relative manner. The most common representations are based on the slicing model that assumes the blocks are organized in a set of slices that recursively bisect the layout horizontally and vertically. The direction and nesting of the slices is recorded in a slicing tree or equivalently in a normalized Polish expression[15]. In this method, blocks cannot overlap, which may lead to improved efficiency in the placement optimization. However, if the optimal solution is not non-slicing, this representation cannot obtain the optimal solution as this representation is restricted to slicing floor plan topologies. Fig.2 shows an example of a slicing structure.

Recently, the sequence-pair, first suggested by Murata et al. [10], and bounded-sliceline grid (BSG), proposed by Nakatake et al.[11], have been proposed as solutions to this problem. The sequence-pair encodes the "left-right" and "up-down" positioning relations between blocks using two sequences of blocks. BSG can define orthogonal relations between blocks

without physical dimensions.

These methods are particularly suitable for stochastic algorithms, such as GA and simulated annealing (SA). These encoding schemes can represent not only slicing structure but also non-slicing structures.

In this chapter, we used sequence-pair as the representation of a solution, as this method can perform more effective searches than BSG. The number of all sequence-pair combinations is smaller than that of BSG.

3.1.1. *Sequence-Pair*

The sequence-pair is used to represent the solution of rectangular packing. Each block has the sequence-pair (Γ_-, Γ_+). In Fig. 3, an example of sequence-pair is shown. To express the relative position, packages are located on the sequence-pair surface. This surface consist of two axes: Γ_- and Γ_+. These axes are not located perpendicularly and horizontally but leans 45 degrees. The relative positions of two blocks are defined by comparing the sequence-pair of the two blocks. Let blocks A and B have the sequence pairs (x_{a-}, y_{a+}) and (x_{b-}, y_{b+}), respectively. In this case, there is a relationship between the positions of the blocks and the sequence pairs as follows:

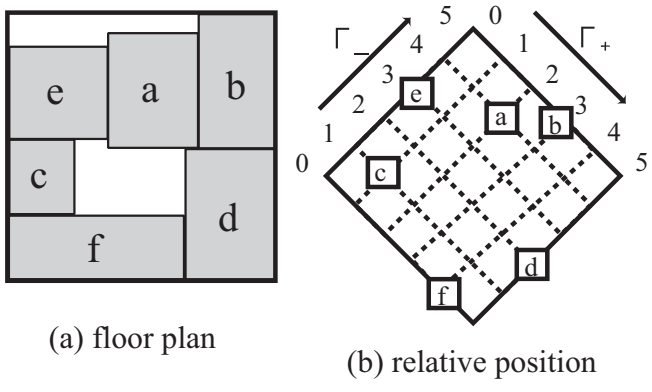
$$\begin{gathered}\text{when } x_{a-} < x_{b-} \text{ and } y_{a+} < y_{b+},\ A \text{ is in the left side of } B\\ \text{when } x_{a-} > x_{b-} \text{ and } y_{a+} > y_{b+},\ A \text{ is in the right side of } B\\ \text{when } x_{a-} < x_{b-} \text{ and } y_{a+} > y_{b+},\ A \text{ is in the upper side of } B\\ \text{when } x_{a-} > x_{b-} \text{ and } y_{a-} < y_{b+},\ A \text{ is in the bottom side of } B.\end{gathered}$$

In addition to the sequence-pair, each block has the orientation information Θ. This information instructs the direction of the block arrangement.

3.1.2. *Encoding System*

A gene of the GA consists of three parts: Γ_-, Γ_+, and Θ. Fig. 3 shows the encoding for 6 blocks.

The relative position (b) is derived from the encoding information (Fig. 3(c)). This position shows the floor plan (a). In this chapter, each block is settled lengthwise or breadthwise. Therefore, Θ takes a value of 0 or 1.



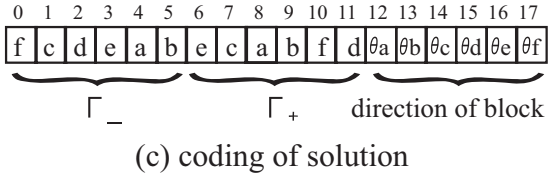


Fig. 3. Encoding example of sequence-pair.

3.2. *GA operators*

For an effective search, it is necessary to choose an appropriate method. Especially, crossover, which is the primary method of optimization, must be chosen carefully.

The traditional genetic crossover operator, one-point crossover, cannot be applied without modification to the combination problem, such as RP or TSP. Some crossovers for combination problems have been previously proposed.

The three crossover methods, which are one of the most commonly used methods for combination problems[16], can be described as follows (Fig.4 shows the concept of these crossover).

Order crossover (OX) : Pass the left segment from parent 1. Construct the right segment by taking the remaining blocks from parent 2 in the same order.

Partially mapped crossover (PMX): The right segments of both parents act as a partial mapping of pairwise exchanges to be performed on parent 1 to generate the offspring.

Cycle crossover (CX): Start with the cell in location 1 of parent 1 (or

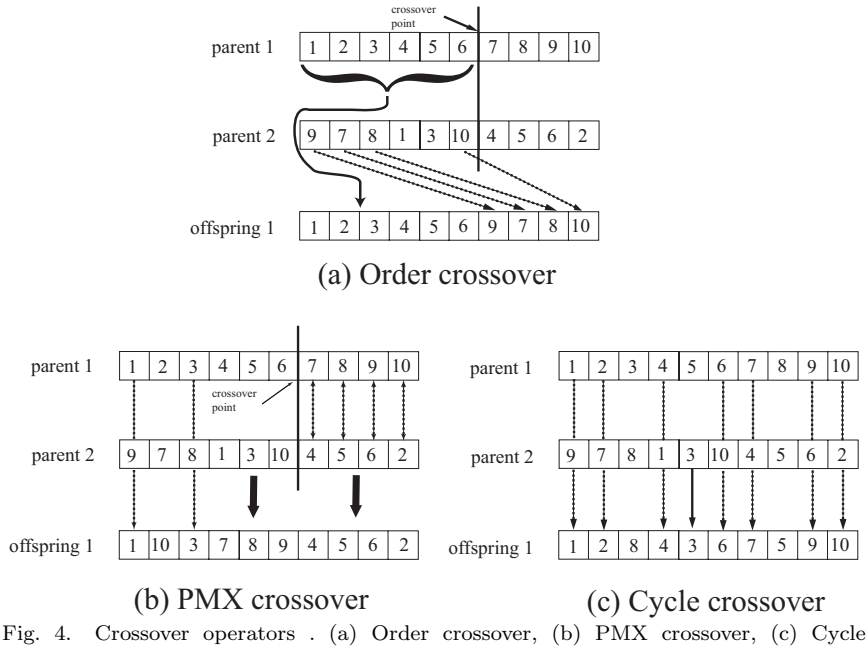


Fig. 4. Crossover operators . (a) Order crossover, (b) PMX crossover, (c) Cycle crossover.

any other reference point) and copy to location 1 of the offspring. The block, which is the same block at location 1 of parent 2, is searched in parent 1 and passed on to the offspring from there. This process continues until we complete a cycle and reach a block that has already been passed.

However, these crossovers cannot provide an efficient search using sequence-pairs, as they do not take into account the features of sequence-pairs[a].

Therefore, an effective crossover must be considered as a position on an oblique grid (Γ_-, Γ_+) that is stetted by two sequences of blocks.

Nakaya et al. proposed a new crossover for sequence-pair, known as Placement-based Partially Exchanging Crossover (PPEX) [12].

[a]In this chapter, we do not describe the performance of these crossover operators. In our previous experience using sequence-pair, however, OX can obtain the best solutions, as compared to other methods. On the other hand, CX does not provide good solutions.

3.2.1. *Placement-based Partially Exchanging Crossover*

Here, we used the Placement-based Partially Exchanging Crossover (PPEX) [12]. PPEX makes a window-territory located in the neighborhood of blocks chosen at random. This window-territory is a continuous part of the oblique-grid that is defined by the sequence-pair. PPEX performs a crossover that exchanges blocks within this window-territory. Therefore, PPEX can exchange blocks within the neighborhood position. The PPEX procedure is illustrated as follows.

Step 1: Two blocks are chosen randomly as parent blocks.
Step 2: The window-territory is created in the neighborhood of the chosen blocks. Let M_c be the set of blocks within window-territory and M_{nc} be the rest of the blocks.
Step 3: Each block of M_c is exchanged according to the sequence of its partner parent and is copied to the child.
Step 4: M_{nc} are directly copied to the child.

Fig.5 displays PPEX when the window-territory size is 4.

In Parent 2, blocks of a and e are chosen for M_c, and blocks of M_c are exchanged.

In this exchange, the relative position of the other parent is referenced. Then, these blocks are copied to the child. With the location information of Parent 1, a, e and f are moven then copied to child 2.

3.2.2. *Mutation Operator*

In this chapter, we describe the use of bit flip of the orientation for block(θ). That is, if θ is 1, let θ be 0. In the opposite case, if θ is 0, let θ be 1.

4. Multi-Objective Optimization Problems by Genetic Algorithms and Neighborhood Cultivation GA

4.1. *Multi-Objective Optimization Problems and Genetic Algorithm*

Several objectives are used in multi-objective optimization problems. These objectives usually cannot be minimized or maximized at the same time due to a tradeoff relationship among them [7]. Therefore, one of the goals of the multi-objective optimization problem is to find a set of Pareto optimal solutions.

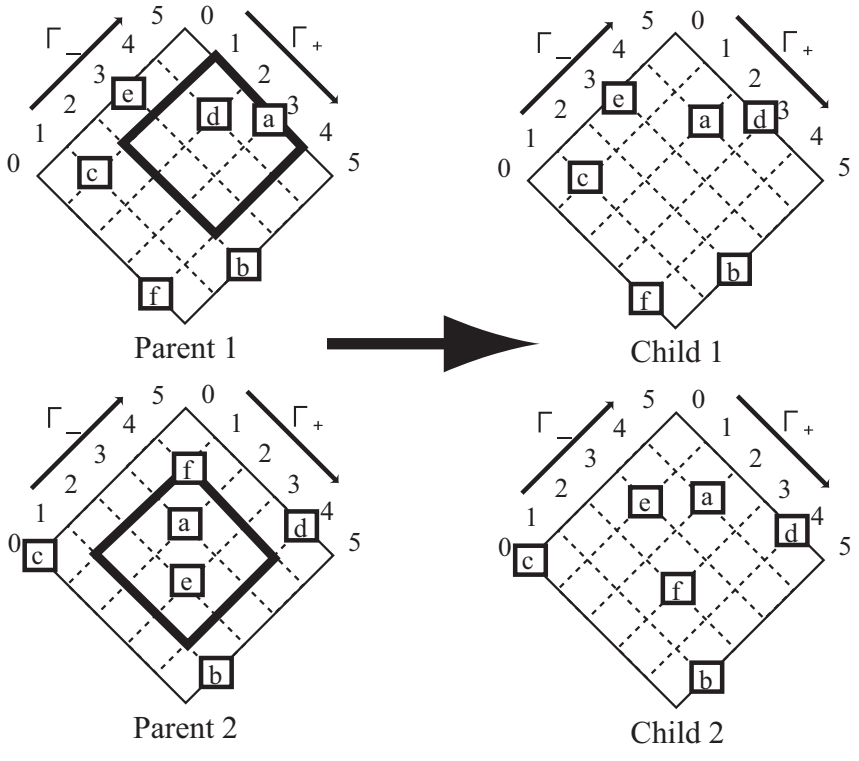


Fig. 5. Placement-based Partially Exchanging Crossover (PPEX).

The Genetic Algorithm is an algorithm that simulates the heredity and evolution of living things [7]. As it is a multi-point search method, an optimum solution can be determined even when the landscape of the objective function is multi-modal. It can also find a Pareto optimum set with one trial in multi-objective optimization. As a result,the GA is a very effective tool for multi-objective optimization problems. There is a great deal of research concerned with multi-objective GA. Also, many new evolutionary algorithms for multi-objective optimization have been recently developed [3,5,7,8,18].

Multi-objective genetic algorithms can be roughly divided into two categories: algorithms that treat Pareto optimal solutions implicitly and those that treat Pareto optimal solutions explicitly [7]. Many of the newest methods treat Pareto optimal solutions explicitly.

Typical algorithms that treat Pareto optimal solutions explicitly include

NSGA-II [5] and SPEA2 [18]. These algorithms have the following similar schemes:

1) Mechanism responsible for retaining nondominated solutions
2) Cut down (sharing) method for maintaining diversity among the nondominated solutions retained
3) Unification mechanism of values of each objective

These mechanisms derive good Pareto optimal solutions. Consequently, a competitive multi-objective genetic algorithm should have these mechanisms.

4.2. *Neighborhood Cultivation Genetic Algorithm*

In this section, we describe the mechanism of a new algorithm called Neighborhood Cultivation Genetic Algorithm (NCGA). NCGA has a neighborhood crossover mechanism in addition to the mechanisms of GAs that were explained in the previous section. In GAs, exploration and exploitation are very important. By exploration, an optimum solution can be found around the elite solution. By exploitation, an optimum solution can be found in a global area. In NCGA, the exploitation factor of the crossover is reinforced. In the crossover operation of NCGA, a pair of individuals for crossover are not chosen randomly, but individuals that are close to each other are chosen. As a result of this operation, child individuals that are generated after the crossover may be close to the parent individuals, and therefore precise exploitation is expected.

Let us denote the search population at generation t by P_t. Also we denote the archive population at generation t by A_t. Using these notations, the overall flow of NCGA can be described as follows.

Step 1: Initialization: Generate an initial population P_0. Population size is N. Set $t = 0$. Calculate fitness values of the initial individuals in P_0. Copy P_0 into A_0. Archive size is also N.
Step 2: Start new generation: set $t = t + 1$.
Step 3: Generate new search population: $P_t = A_{t-1}$.
Step 4: Sorting: Individuals of P_t are sorted according to the values of the focused objective. The focused objective is changed at every generation. For example, when there are three objectives, the first objective is focused in the first generation and the third objective is focused in the third generation. The first objective is focused again in the fourth generation.
Step 5: Grouping: P_t is divided into groups consisting of two individuals.

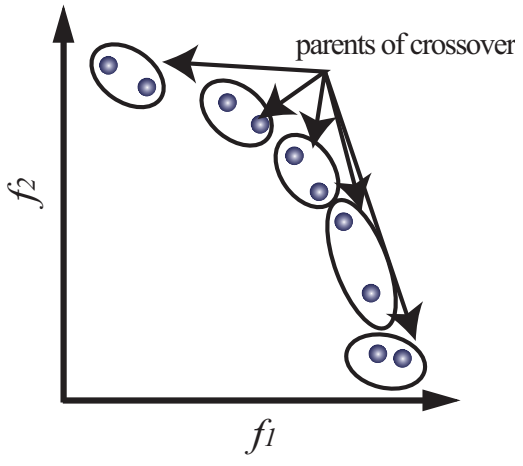


Fig. 6. Neighborhood crossover.

These two individuals are chosen from the top to the bottom of the sorted individuals.

Step 6: Crossover and Mutation: In a group, crossover and mutation operations are performed. From two parent individuals, two child individuals are generated. Here, parent individuals are eliminated.

Step 7: Evaluation: All of the objectives of individuals are derived.

Step 8: Assembling: All the individuals are assembled into one group and this becomes the new P_t.

Step 9: Renewing archives: Assemble P_t and A_{t-1} together. The N individuals are chosen from $2N$ individuals. To reduce the number of individuals, the same operation of SPEA2 (Environment Selection) is performed. In NCGA, this environment selection is applied as a selection operation.

Step 10: Termination: Check the terminal condition. If it is satisfied, the simulation is terminated. If not, the simulation returns to Step 2.

In NCGA, most of the genetic operations are performed in a group consisting of two individuals.

The neighborhood crossover is performed for a crossover operations with population that is sorted according to the values of the focused objective. As two adjacent individuals of the sorted population are relatively close from each other in objective space, a "neighborhood crossover" is realized by using two adjacent individuals. The concept of neighborhood crossover is shown in Fig.6.

However, if the focused objective has completely converged, applying crossover over a pair of individuals may cause no changes at the final stages of the search. Therefore, we use the following techniques within our crossover operator:

1) The focused objective is changed at every generation.
2) The sorted population is slightly disturbed by using "neighborhood shuffle".

The focused objective is changed one by one at every generation. For example, when there are three objectives, the first objective is focused in the first generation and the third objective is focused in the third generation. The first objective is focused again in the fourth generation.

The "neighborhood shuffle" is a technique, which randomly shuffles the population within a definite range. The range of neighborhood shuffle is defined as 10 percent of the population size. For example, when the population size is 100, the population is randomly shuffled within a range of size 10.

To use these techniques, the parents subject to crossover should be changed at every generation, even if the population had stayed unchanged. In addition, an exchange of individuals would be more active.

The following features of NCGA are the differences between SPEA2 and NSGA-II.

1) NCGA has a neighborhood crossover mechanism.
2) NCGA has only environment selection and does not have mating selection[b].

5. Numerical Examples

In this chapter, we describe the application of NCGA to some numerical experiments. We used four instances of this problem: ami33, ami49, rdm100, and rdm500. The instances ami33 and ami49, whose data are in the MCNC benchmark, consist of 33 and 49 blocks (rectangles). The instances rdm100 and rdm500 were randomly generated and have 100 and 500 rectangles, respectively.

The results were compared with those of SPEA2 [18], NSGA-II [5], and non-NCGA. Non-NCGA is the same algorithm as NCGA without neighborhood crossover.

[b]If there are diverse solutions that have the same design variables, neighborhood crossover may not perform effectively. Therefore, the search population (P_t) is produced by making a copy of the archive population (A_t).

Table 1. GA Parameters

population size	200
crossover rate	1.0
mutation rate	1/bit length
terminal generation	400

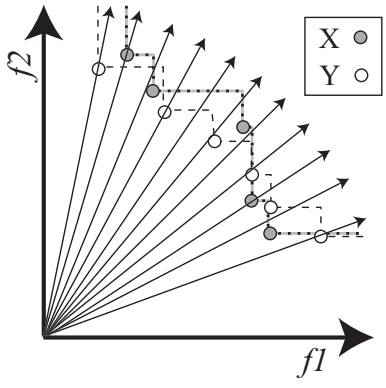


Fig. 7. Sampling of the Pareto frontier lines of interSection

5.1. *Parameters of GAs*

Table 1 displays the GA parameters used. We used the previously described GA operator, PPEX and the bit flip of block orientation. The length of the chromosome is three times as long as the number of blocks.

5.2. *Evaluation methods*

To compare the results obtained by each algorithm, the following evaluation methods were used.

5.2.1. *Sampling of the Pareto frontier lines of intersection(I_{LI})*

This comparison method was reported by Knowles and Corne [8]. The concept of this method is shown in Fig. 7. This figure illustrates two solution sets of X and Y derived by the different methods.

The following three steps are the comparison procedures. Firstly, the attainment surfaces defined by the approximation sets are calculated. Secondly, the uniform sampling lines that cover the Pareto tradeoff area are defined. For each line, the intersections of the line and the attainment surfaces of the derived sets are obtained. These intersections are then compared. Finally, the Indication of Lines of Intersection (I_{LI}) is derived. When the

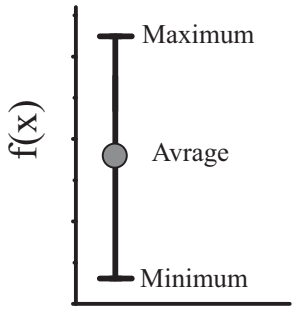


Fig. 8. Example of I_{MMA}.

two approximation sets X and Y are considered, $I_{LI}(X, Y)$ indicates the average number of points X that are ranked higher than Y. Therefore the most significant outcome would be $I_{LI}(X, Y) = 1.0$ and $I_{LI}(Y, X) = 0.0$.

To focus only on the Pareto tradeoff area as defined by the approximation sets and to derive the intuitive evaluation value, the following terms are considered:

- The objective values of approximation sets are normalized.
- The sampling lines are located in the area where the approximation sets exist.
- Many sampling lines are prepared. In the following experiment, 1000 lines were used.

5.2.2. *Maximum, Minimum and Average values of each object of derived solutions (I_{MMA})*

To evaluate the derived solutions, not only the accuracy but also the spread of the solutions is important. To discuss the spread of the solutions, the maximum, minimum, and average values of each object are considered. Figure 8 shows an example of this measurement. In this figure, the maximum and minimum values of the objective function are illustrated and the medium value is shown as a circle.

5.3. *Results*

In this chapter, we examined four types of problem: ami33, ami49, rdm100, and rdm500 blocks. In this section, we discuss only the instances ami33 and rdm500.

Proposed NCGA, SPEA2, NSGA-II, and non-NCGA (NCGA without

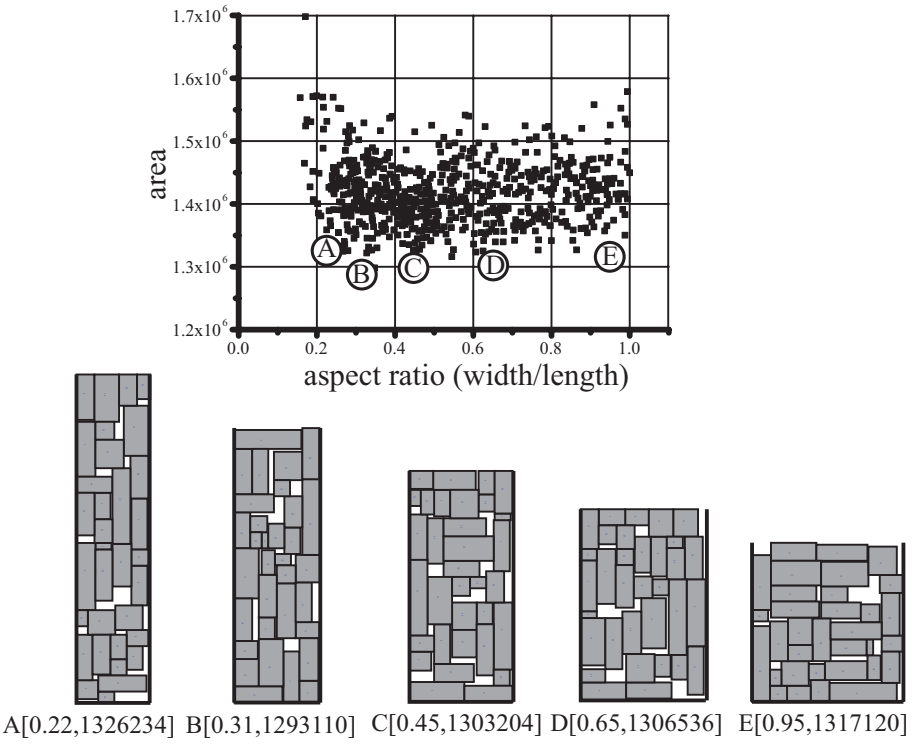


Fig. 9. Placement of the blocks(ami33).

neighborhood crossover) were applied to these problems. Thirty trials were performed and all results shown are the averages of 30 trials.

5.3.1. *Layout of the solution*

It should be verified whether solutions that are derived by the algorithm are opposite placement of blocks. In this section, we focus on ami33, which consisted of 33 blocks. The placement of ami33, which is presented by solutions of NCGA, is shown in Fig. 9.

Some of the typical solutions are illustrated in Fig. 9. As this is a combination of the $N! \times N! \times 2^N$ problem with N blocks, the real optimum solutions were not derived. In this experiment, 80,000 function calls (200 individuals and 400 generations) were performed. These results may be reasonable, as there were very few blank spaces. We also used a sequence-pair and PPEX to derive good solutions as these techniques are very suitable

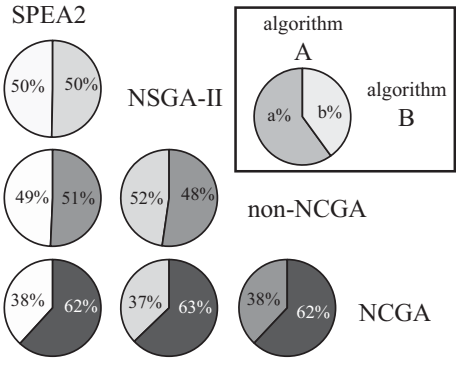


Fig. 10. Results of I_{LI}(ami33).

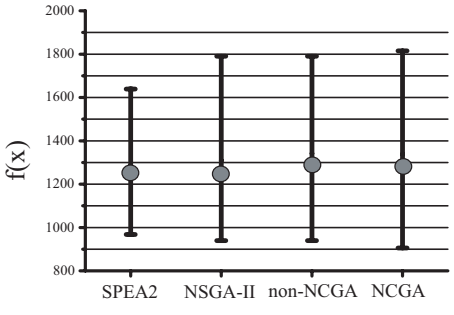


Fig. 11. I_{MMA} of ami33

for GAs and RP.

5.3.2. **ami33**

The results of ami33, I_{LI} are shown in Fig. 10, and those of I_{MMA} are shown in Fig. 11. Fig. 12 shows the nondominated solutions of each algorithm. In this figure, all nondominated solutions derived from the 30 trials are plotted.

I_{LI} of Fig. 10 indicates that solutions of NCGA are closer to the real Pareto solutions than those obtained by the other methods. This is also confirmed by the plots of the nondominated solutions(Fig.12). It is also clear from I_{MMA} of Fig. 11 that NCGA and non-NCGA can find more widely spread nondominated solutions as compared to the other methods.

Non-NCGA can obtain widely spread nondominated solutions. However, as compared to the real Pareto solutions, non-NCGA is not ideal. This result

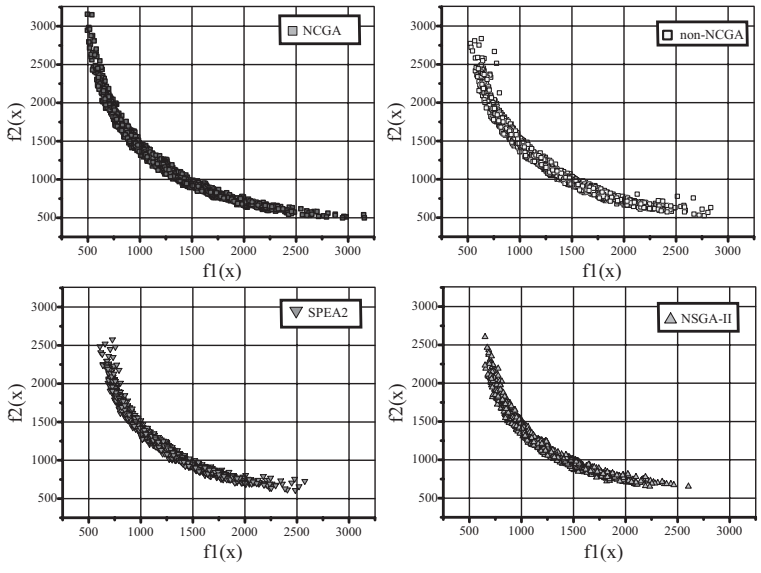


Fig. 12. Nondominated solutions(ami33).

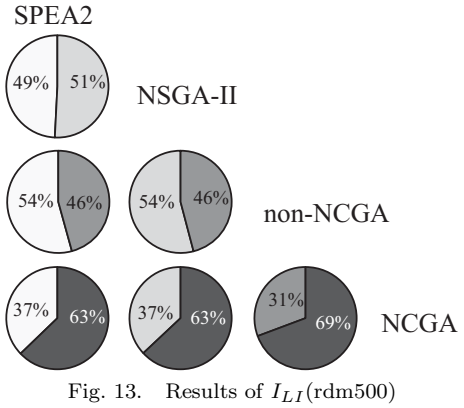


Fig. 13. Results of I_{LI}(rdm500)

shows that neighborhood crossover can derive good solutions in RP.

5.3.3. **rdm500**

The results of rdm500 are shown in Fig. 13 and Fig. 14. Fig. 15 illustrates the nondominated solutions of the different algorithms.

The results from this problem showed a similar trend to those of the previous problem. From Fig.13 and Fig.15, it is clear that NCGA obtained

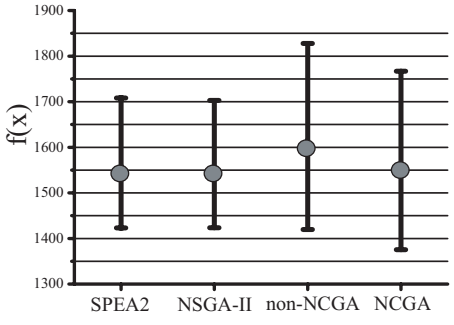


Fig. 14. I_{MMA} of rdm500

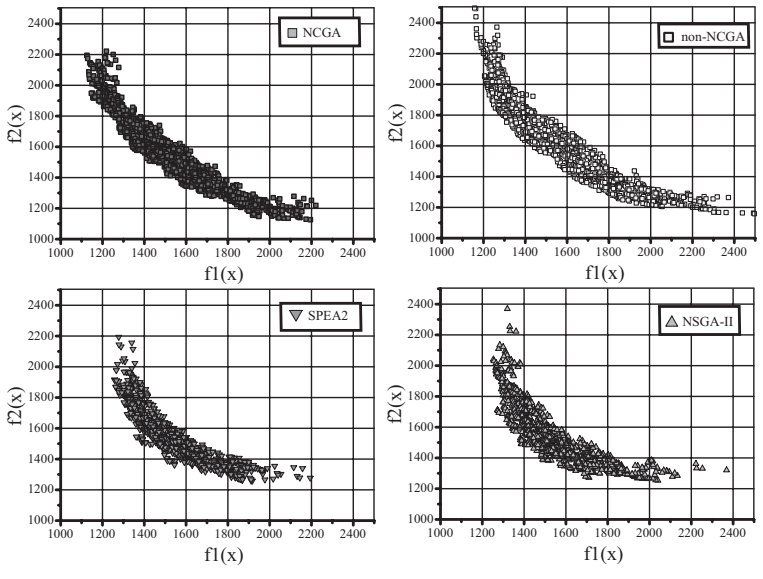


Fig. 15. Nondominated solutions(rdm500).

a better value of I_{LI}; i.e., the solution of NCGA was much better than those of the other methods. Similarly to the previous problem, the solutions of non-NCGA were far from the real Pareto front. Therefore, the neighborhood crossover was very effective to derive good solutions in RP, irrespective of the number of blocks.

On the other hand, in this problem, the solutions of SPEA2 and NSGA-II were gathered around the center of the Pareto front. These observations indicate that SPEA2 and NSGA-II tend to concentrate in one part of the Pareto front when the number of blocks is very large. On the other hand,

Fig. 14 and Fig. 15 indicate that NCGA and non-NCGA maintained high degrees of diversity of their solutions during the search even if the number of blocks was very large.

6. Conclusion

In this chapter, we described the implementation of MOGA for the Multi-Objective Rectangular Packing Problem (RP). We described the formulation of RP, implementation of GA to RP, and our experience with RP using GA.

The main issues associated with the implementation of GA to RP are the representation of a solution and the appropriate GA operator. In this chapter, we explain sequence-pair as an effective representation of placement, and PPEX as an effective crossover in cases using sequence-pair.

In addition, based on our experience using GA for RP, Neighborhood Cultivation GA (NCGA), which has not only the important mechanism of the other methods but also the mechanism of neighborhood crossover selection, was applied to Multi-Objective RP. We confirmed that MOGA is a very effective method for RP. In addition, NCGA can obtain the best solutions as compared to other methods. Through numerical examples, the following points were clarified.

1) The RP described in this chapter is a large scale problem. For this problem, a reasonable solution is derived with a small calculation cost. It is assumed that a sequence-pair and PPEX work well in this problem.
2) In almost all the test functions, the results of NCGA were superior to those of the other methods. From this result, it can be noted that NCGA is a good method for the RP.
3) NCGA was obviously superior to NCGA without neighborhood crossover in all problems. The results emphasized that neighborhood crossover allows the derivation of good solutions in RP.
4) When the number of blocks is very large, the solutions of SPEA2 and NSGA-II tend to concentrate in the center of the Pareto front. However, NCGA and non-NCGA could retain diversity of the solutions.

References

1. T. Arslan, D. H. Horrocks, and E. Ozdemir. Structural synthesis of cell-based vlsi circuits using a multi-objective genetic algorithm. In *IEE Electronic Letters*, volume 32, pages 651–652, 1996.

2. F. Balasa and K. Lampaert. Symmetry within the sequence-pair representation in the context of placement for analog design. In *IEEE Trans. on Comp.-Aided Design of IC's and Systems*, volume 19, pages 721–731, 2000.
3. C. A. Coello Coello, D.A. Van Veldhuizen, and G. B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers, New York, May 2002. ISBN 0-3064-6762-3.
4. J. P. Cohoon and W. D. Paris. Genetic placement. In *Proceedings of The IEEE International Conference on Computer-Aided Design*, pages 422–425, 1986.
5. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.
6. D. Gopinath, Y. K. Joshi, and S. Azarm. Multi-objective placement optimization of power electronic devices on liquid cooled heat sinks. In *the Seventeenth Annual IEEE Symposium on Semiconductor Thermal Measurement and Management*, pages 117–119, 2001.
7. K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. Chichester, UK:Wiley, 2001.
8. J. D. Knowles and D. W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. In *Evolutionary Computation*, volume 8, pages 149–172, 2000.
9. J. Lienig and K. Thulasiraman. A genetic algorithm for channel routing in vlsi circuits. In *Evolutionary Computation*, volume 1, pages 293–311, 1994.
10. H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani. VLSI Module Placement Based on Rectangle-Packing by the Sequence-Pair. In *IEEE Transactions on Computer Aided Design*, volume 15, pages 1518–1524, 1996.
11. S. Nakatake, H. Murata, K. Fujiyoshi, and Y. Kajitani. Module Placement on BSG-Structure and IC Layout Applications. In *Proc. of International Conference on Computer Aided Design '96*, pages 484–491, 1996.
12. S. Nakaya, S. Wakabayashi, and T. Koide. An adaptive genetic algorithm for vlsi floorplanning based on sequence-pair. In *2000 IEEE International Symposium on Circuits and Systems, (ISCAS2000)*, volume 3, pages 65–68, 2000.
13. M. J. O'Dare and T. Arslan. Generating test patterns for vlsi circuits using a genetic algorithm. In *IEE Electronics Letters*, volume 30, pages 778–779, 1994.
14. P. Grignon, J. Wodziack, and G. M. Fadel. Bi-objective optimization of components packing using a genetic algorithm. In *In NASA/AIAA/ISSMO Multidisciplinary Design and Optimization Conference*, pages 352–362, 1996.
15. V. Schnecke and O. Vornberger. An adaptive parallel genetic algorithm for vlsi-layout optimization. In *4th Conf. Parallel Problem Solving from Nature (PPSN IV)*, pages 859–868, 1996.
16. K. Shahookar and P. Mazumber. A genetic approach to standard cell placement using meta-genetic parameter optimization. In *IEEE Transaction on Computer-Aided Design*, volume 9, pages 500–511, 1990.
17. S. Watanabe, T. Hiroyasu, and M. Miki. Neighborhood cultivation genetic

algorithm for multi-objective optimization problems. In *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution And Learning (SEAL-2002)*, pages 198–202, 2002.

18. E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. In K. Giannakoglou, D. Tsahalis, J. Periaux, P. Papailou, and T. Fogarty, editors, *EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, pages 95–100, Athens, Greece, 2002.

INDEX