Automatic Design for Pipe Arrangement using Multi-objective Genetic Algorithms

Satoshi Ikehira
Graduate School of Engineering, Kyushu University, Japan

Hajime Kimura
Faculty of Engineering, Kyushu University, Japan

Hiroyuki Kajiwara
Faculty of Engineering, Kyushu University, Japan

Abstract

This paper presents an automatic design method for pipe arrangement. A pipe arrangement design problem is proposed for a space in which many pipes and objects co-exist. This problem includes large-scale numerical optimization and combinatorial optimization problems, as well as two criteria. For these reasons, it is difficult to optimize the problem using usual optimization techniques such as Random Search. Therefore, a multi-objective genetic algorithm (GAs) suitable for this problem is developed.

A pipe is characterized by both a pattern of generation and numerical parameters. The former describes the way the pipe bends and the latter details the length of the straight parts. For this reason, a combination of the pattern of generation and the numerical parameters is used for the solution representation and a new method of crossover is proposed that takes into account interference with obstacles. As the number of pipes increases, it becomes rapidly more difficult to find feasible solutions where pipes do not interfere with each other. Therefore, two modification operators that transform infeasible solution candidates into feasible ones are introduced. One operator modifies the pipe having a lot of interferences with other pipes so that it will not interfere with them, and the other is related with the operation that modifies the pipe that travels through obstacles. Although there are cases in which pipes cannot completely avoid obstacles in practical designs, this situation is taken into consideration by this design process.

The proposed method for optimizing a pipe arrangement efficiently is demonstrated through several experiments, and remarks are provided for applying this methodology to a practical pipe arrangement design.

1. Introduction

Recently, the shipbuilding industry has been able to integrate design and manufacturing systems due to advances in data processing technology. This integration promotes automation and reduces labor requirements, which have become necessary due to the lack of skilled workers and the change of the younger generation's work ethic. The design, analysis, and production of a given system can all be completed using three-dimensional data by employing CAD (Computer-Aided Design) software. It is quite difficult to visualize a complex pipe arrangement using a two-dimensional blueprint. For this reason, pipe arrangement designs have been generated using trial and error methods. Even now, a large part of a design relies upon the designer's experience, because the automation of pipe arrangements has not yet been achieved. A reason for this may be the underreporting of research on the automatic design of pipe arrangements.

This paper formulates a pipe arrangement design problem where many pipes co-exist in a close space and some obstacles must be avoided. This problem includes large-scale numerical optimization and combinatorial optimization problems, and has two criteria. Therefore, it is difficult to optimize the objective using usual optimization techniques such as Random Search.

A multi-objective optimization problem is an important problem that occurs in a lot of real problems, and a full optimum solution cannot be obtained (Coello [1], Corne [2]). Therefore, solutions are generated using the concept of the Pareto optimum solution, which is a solution that dominates all other solutions for at least one criterion. The genetic algorithm (GA) is an optimization technique that was inspired by the evolution process of natural life (Ono [3]). This method is expected to be an effective method for obtaining Pareto solutions efficiently, because the GA searches on multimodal search space and finds a set of Pareto solutions by explicitly handling plural criteria. Therefore, multi-objective genetic algorithms (GAs) that are suitable for this problem are proposed. It is demonstrated that the proposed method can optimize a pipe arrangement efficiently, and some remarks are provided on applying this method to a practical pipe arrangement design.

2. Formulating Pipe Arrangement Design Problem

A pipe arrangement for a space in which many pipes co-exist closely and several obstacles exist is formulated in this section.

2.1 Generating Pipes

The process of designing a pipe arrangement consists of three parts: 1) designing a distribution diagram that shows which equipment connect with other equipment, 2) designing an arrangement of the equipment, and 3) designing the route for the pipes between equipment.

In this paper, a pipe arrangement is automatically designed when both origin and endpoint coordinates and the directions in which the pipes expand (direction vectors) are provided.

The objective for the pipe arrangement design is to minimize the total length of pipes within the arrangement, in order to improve the economical efficiency, while providing a design in which the pipes can be attached easily. Pipe materials are specified in JIS (Japanese Industrial Standards), and it is economically desirable to use these materials. When a pipe is squarely bent, the two pipes are welded together using a material specified in JIS. However, when a pipe is bent at an arbitrary angle, the pipe is bent with a pressing machine. Because the pressing machine is very time-consuming, pipes are bent squarely unless special conditions exist. When pipes need to be bent at arbitrary angles due to the demands of the design, this situation is considered using direction vectors.

There is an infinite number of piping routes that could be used to connect a given origin and endpoint. Therefore, a limit is imposed on the degrees-of-freedom for the route. A piping route is designed that essentially specifies the bending points (nodes). First, a route is designated as mode-1, mode-2, or mode-3, according to number of direction vectors. Next, a generation pattern is set according to the number of nodes and the relative position between the origin and the end-point, as shown in Figs. 1, 2, and 3. A pipe is characterized by both a pattern of generation and three numerical parameters that describe the length of the straight parts. It is possible to expand the generation pattern number by increasing the number of nodes. However, it is easier to assemble a pipe with a fewer number of nodes, so the number of nodes should be minimized while avoiding interference from other pipes and obstacles.

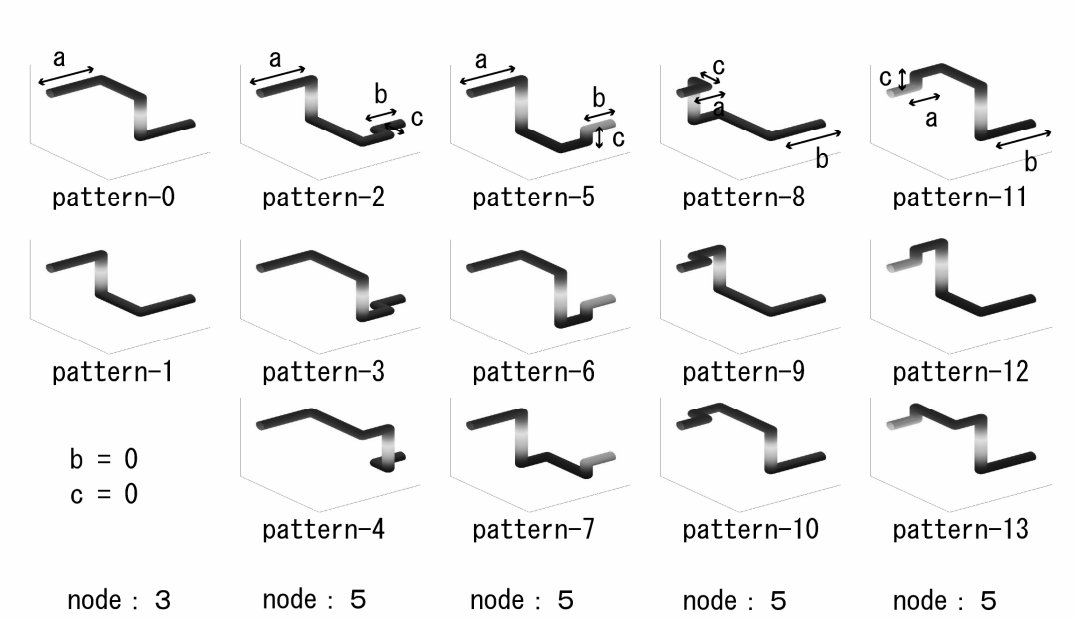


Fig. 1 Mode-1

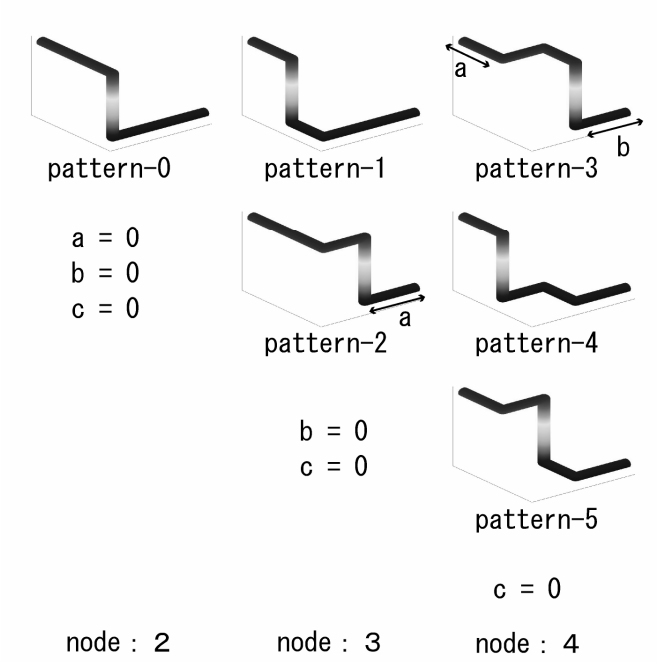


Fig. 2 Mode-2

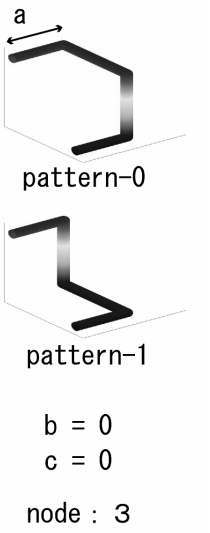


Fig. 3 Mode-3

2.2 Interference with Other Pipes

The conditional expression that describes two pipes interfering with each other is as follows:

$$d \le r_0 + r_1 \tag{1}$$

where d is the shortest distance between two pipes, and r_0 and r_1 are the two pipe radii.

However, in a practical design, the size of a material called the flange must be considered. The size of the flange is set according to the diameter of the pipe, and must be taken into account when calculating pipe interference.

2.3 Interference with Obstacles

Cubes are arranged as the obstacles in this experiment and an obstacle with a complicated shape can be modeled by a combination of several cubes.

Two methods for automatically avoiding obstacles are considered. One method involves labeling the pipe infeasible when the pipe interferes with an obstacle, and the other method makes the pipe feasible by assigning an evaluation value when a pipe interferes with an obstacle. In a practical pipe arrangement design, a space is occasionally set aside to allow for maintenance people to pass. This space is then considered as an obstacle in the design. In such a case, it is necessary for pipes to avoid the obstacle as much as possible, although pipes need not completely avoid the obstacle. For this purpose, the latter method of making the pipe feasible by assigning an evaluation value is adopted.

[The evaluation function for obstacle]

The evaluation function for an obstacle is shown in Equation (2). In this equation, that evaluation values are worse when a pipe passes through the center of an obstacle and the length of the intersection is long.

$$f_{\_obstacle} = \sum_{l=1}^{n_o} \sum_{k=1}^{n_p} (b_{kl} - \overline{a}_{kl} + A_l) \tag{2}$$

Here, k is the index of each pipe, l is the index of each obstacle, n_o is the number of obstacles, n_p is the number of pipes, b_{kl} is the length of the intersection when a pipe with index k intersects an obstacle with index l, $\overline{a}_{kl}$ is the averaged length between the center of the gravity of the obstacle with index l and the part divided by each node of the pipe with index k, and A_l is the length between the center of the gravity and the top of the obstacle with index l.

If all of the pipes avoid all of the obstacles, the evaluation value is 0. Fig. 4 shows an example of calculating the evaluation function.

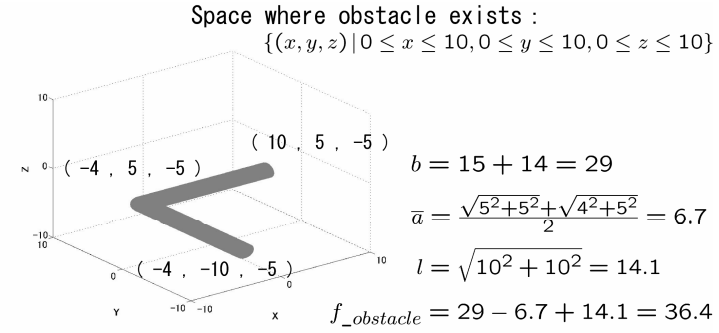


Fig. 4 An example of calculating of the evaluation function for an obstacle

3. Designing Multi-objective GAs

This section first provides a brief explanation of GAs. Then, a multi-objective GA is proposed for which the GA optimization technique will be applied to solve the multi-objective optimization problem. Finally, the effectiveness of the proposed method is verified through some experiments.

3.1 Genetic Algorithms (GAs)

The genetic algorithm (GA) is an optimization technique inspired by the evolution process of natural life. The GA consists of a very flexible framework that recently has been applied to not only global optimization problems, but also to multi-objective optimization methods in various fields. The general algorithm is described as follows:

1. *Generation of Initial Population*

Generate an initial population that consists of plural individuals, called solution candidates, generated at random. Let the initial population be the current population.

2. *Selection for Reproduction*

Choose pairs of individuals from the current population.

3. *Generation of Children*

Apply a crossover and a mutation to the pairs of individuals selected in Step 2 to produce new children, called new solution candidates.

4. *Selection for Survival*

Select individuals from the children generated in Step 3 and the individuals in the current population to produce the next population. Let the next population be the current population.

5. Repeat the previous Steps 2 to 4 until a certain stop condition is satisfied.

The process of designing a GA consists of two parts: 1) designing a representation, a crossover operator and a mutation operator, and 2) designing a generation-alternation model. When designing a representation, a crossover operator, and a mutation operator, it must be determined how a solution will be represented on the computer and how to generate a new solution from two or more solutions. The performance of GAs heavily depends on the representation, crossover operator, and mutation operator. It is important to consider characteristics of the problem domain when designing a representation, crossover operator, and mutation operator. In the generation-alternation model design, the method for choosing pairs of parents for generating children by crossover and mutation and the method for selecting individuals for survive into the next generation must be determined. The generation-alternation model allows the GA to search multimodal search space effectively and to find a set of non-dominated solutions in a single run by explicitly handling plural criteria.

3.2 Designing Representation and Crossover

The representation and crossover that are traditionally used for function optimization are Binary cording or Gray cording that are defined on a bit string and a one-point crossover, a two-point crossover, and a uniform crossover. Recently, various real-coded genetic algorithms have been proposed in which the phase of the phenotype place is corresponds to the phase of the genotype place (Ono [4]). Unfortunately, it is difficult to apply the traditional representation and crossover to the current problem because it contains both numerical optimization and combinatorial optimization problems. Therefore, a representation and crossover suitable for this problem are developed.

Instead, a combination of both a pattern of generation and numerical parameters are used as the representation, and is called the generation gene.

A Crossover with Two Genes (XTG) is proposed as the crossover for this problem. The XTG generates one child from two parents using a gene named the obstacle gene, which contains the obstacle interference information, as shown in Fig. 5. The child generation gene is basically to be the same one as either of the two parents or generated at random using obstacle gene assisting. The outline of XTG is described as follows:

1. In the case that both parents avoid obstacles, the generation gene for the child is the same one as either of the two parents.

2. In the case that only one parent avoids obstacles, the generation gene for the child is same as that parent.

3. In the case that both parents interfere with obstacles, the generation gene for the child is generated at random.

To demonstrate the effectiveness of the XTG, a crossover based on a uniform crossover (UX), which is a traditional crossover, is also designed. The UX generates one child from two parents, as shown Fig. 6. The generation gene for the child of each pipe is the same one as either of its parents with an equal probability of being chosen.

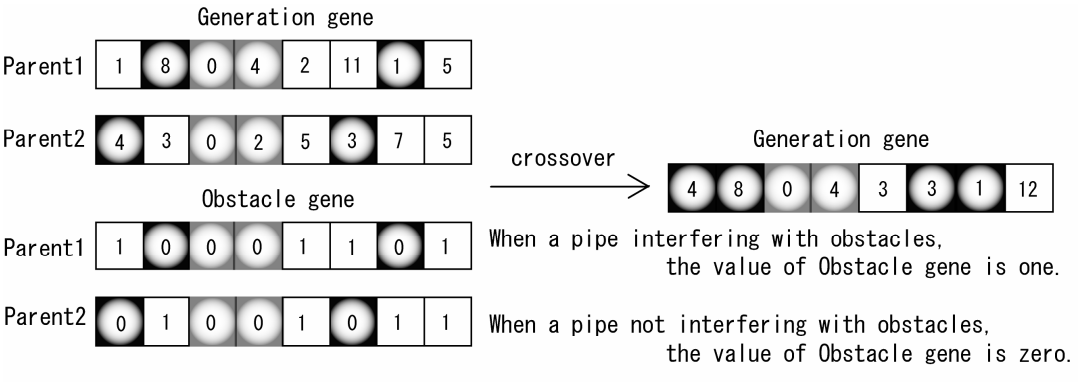


Fig. 5 An example of the Crossover with Two Genes (XTG)

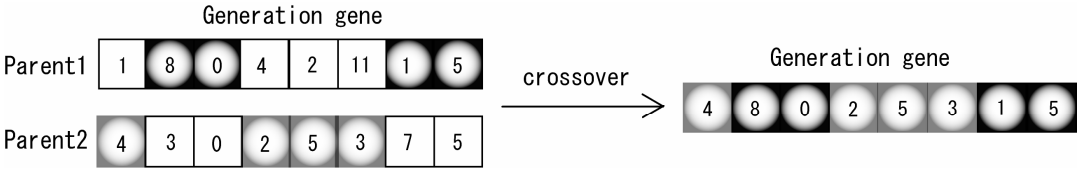


Fig. 6 An example of the Uniform Crossover (UX)

3.3 Designing Mutation

The designed mutation shown in Fig. 7. The mutation is applied to each generation gene at a constant probability (mutation rate). The gene to which the mutation is applied is regenerated at random.

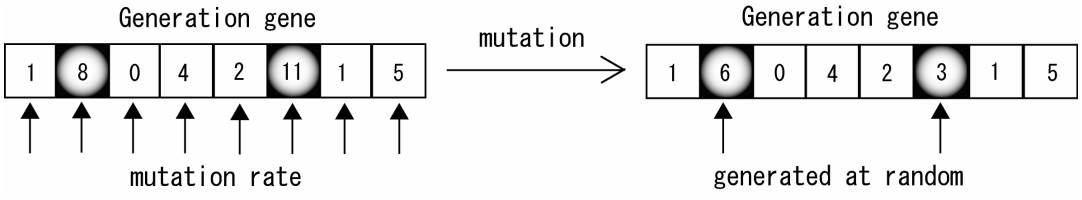


Fig. 7 An example of the mutation

3.4 Designing Generation Alternation Model

A generation alternation model based on the Pareto optimal selection strategy, as shown in Fig. 8 (Kobayashi [5]), is employed.

The outline of the model is as follows. First, a pair of parents is selected randomly from the population of the current generation. Next, children are generated by applying a crossover or a mutation operator to the parents *n* times. Non-dominated individuals from the children and the current population are then selected, and these individuals become the population of the next generation. A non-dominated individual is a solution that is superior to all other solutions in at least one criterion. The generation alternation model using this strategy can explicitly optimize plural criteria without setting a trade-off ratio among the plural criteria.

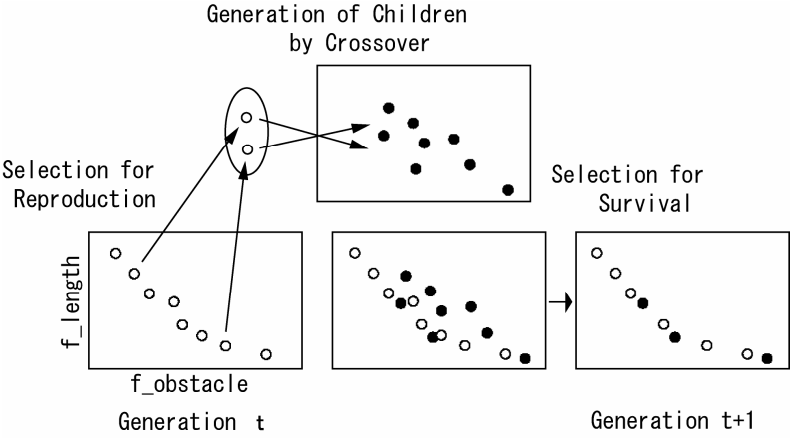


Fig. 8 Generation alternation model

3.5 Designing Modification

In this paper, a solution candidate is regarded as infeasible when pipes interfere with each other. When initial solution candidates are generated, the probability of pipes avoiding interference with each other rapidly decreases as the number of pipes increases because the ratio of occupied space increases. In the search phases, new solution candidates created by a crossover and a mutation operator have a low probability of feasibility because the crossover and mutation operators generally do not take into account pipe interference.

To overcome this problem, a Modification Operator on Contact (MOC) is proposed that makes infeasible solution candidates feasible as a modification operator, which changes the properties of the original solution candidates as little as possible.

3.5.1 Basic Concepts

The number of interferences with other pipes is counted and solutions that have a lot of interferences are modified so that pipes that do not interfere with each other. If at least one pipe interferes with other pipes, this modification is repeated until no pipes interfere with each other.

3.5.2 Algorithm of the MOC

The proposed algorithm for modifying a solution candidate is described below.

1. Rank the pipes in order of the most number of interferences.

2. Restructure the pipe with the most interferences at random using the following process:

(a) In the case that all pipes do not interfere with each other, it ends.

(b) In the case that at least one pipe interferes with other pipes, the process is as follows:

(i) In the case that the same one as the restructured pipe appears α times consecutively, go to Step 3.

(ii) Otherwise, go to Step 1.

3. Restructure both the pipe with the most interferences and the pipe with the second-most interferences at random using the following process:

(a) In the case all pipes do not interfere with each other, it ends.

(b) In the case that at least one pipe interferes with other pipes, the process is as follows:

(i) In the case that the same ones as the two restructured pipes appear α times consecutively, go to Step 4.

(ii) Otherwise, go to Step 1.

4. In the same way, repeat to the last pipe, the pipe with the fewest interferences with other pipes.

In this algorithm, α is the slip number. α is a constant that is set by the designer according to the specific problem.

3.5.3 How to use the MOC

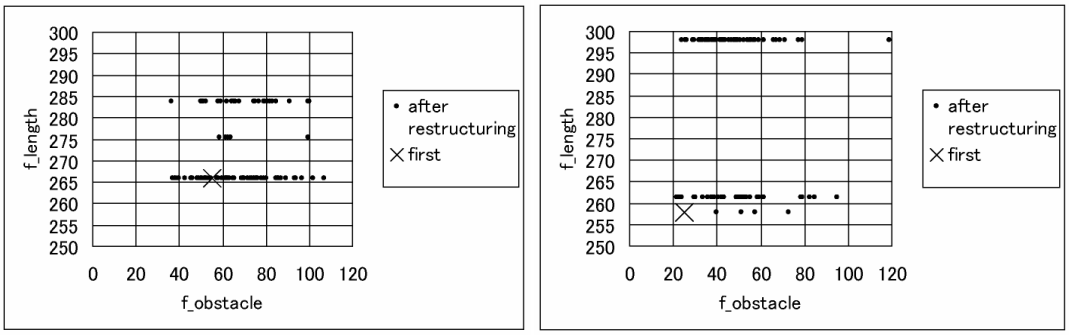
The modification operator is used not only when generating an initial population, but also when applying a crossover and a mutation operator.

3.5.4 Effectiveness of the MOC

The MOC was applied to the pipe arrangement design problem shown in Table 1 in Section 3.6 to confirm its effectiveness. α was set to 10 and the XTG was used as the crossover operator. Figs. 9 (a) and 9(b) display the distribution charts that

show plots of the change of evaluation values after applying the MOC to an infeasible solution candidate 100 times.

The evaluation values are distributed in the upper right quadrant in both figures after the MOC is applied, denoting a bad evaluation score. This occurs because the XTG generates children to improve the evaluation value for an obstacle without considering pipe interferences. Moreover, only a few pipes were restructured because the evaluation value for pipe length did not change discretely.



(a) Example 1 (b) Example 2

Fig. 9 Change of the evaluation value after applying the MOC

3.6 Experiments and Results

To confirm the effectiveness of the proposed method, it was applied to the pipe arrangement design problem shown in Table.1. In the pilot experiment, the initial number of generating solution candidates was set to 100, the number of crossovers was set at 100, the mutation rate was set at 0.1, and the α value of the MOC was set at 10.

Table 1 Problem specification

	Mode-1	mode-2	mode-3
The number of pipes	5	7	3
The number of variables	3	2	1
The number of combinations	14	6	2
The total number of variables	15	14	3
The total number of combinations	537,824	279,936	8
The number of all variables	32		
The number of all combinations	1,204,450,394,000		

Space where pipes arranged: $\{(x, y, z) \mid 0 \le x \le 10, 0 \le y \le 10, 0 \le z \le 10\}$

Space where obstacle exists: $\{(x, y, z) \mid 2 \le x \le 8, 2 \le y \le 8, 2 \le z \le 8\}$

Figs. 10, 11, 12 show the first Pareto solutions, the Pareto solutions for the 100th generation, and the Pareto solutions for 1000th generation, respectively. Using the XTG as a crossover, a solution with no obstacle interference was found in the 100th generation. These evaluation values can be considered settled because the

improvement to these values is little in the 1000th generation. However, when UX is used as a crossover, a solution without interferences was not found, even in the 1000th generation. Fig. 13 shows the solution for an evaluation value of 0 when XTG was used in Fig. 12. The cube at the center of the figure represents the obstacle. This figure indicates that a pipe arrangement without interferences was found even though pipes co-exist in a small space.

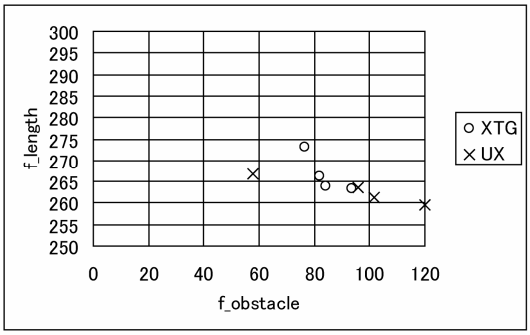


Fig.10 First Pareto solutions

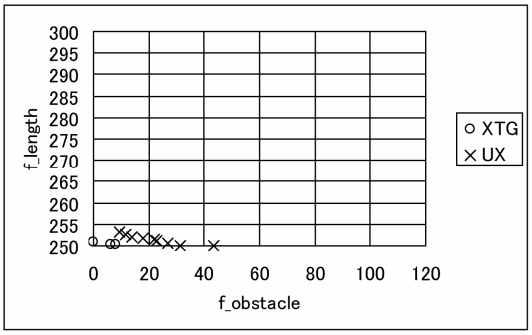


Fig.11 Pareto solutions in the 100th generation

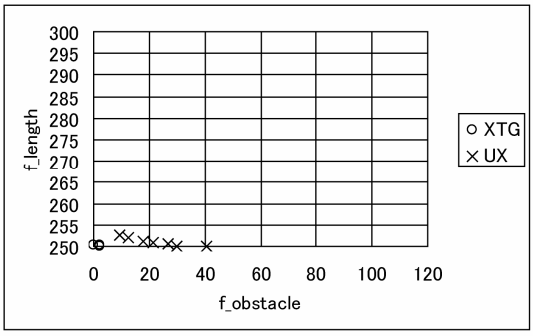


Fig.12 Pareto solutions in the 1000th generation

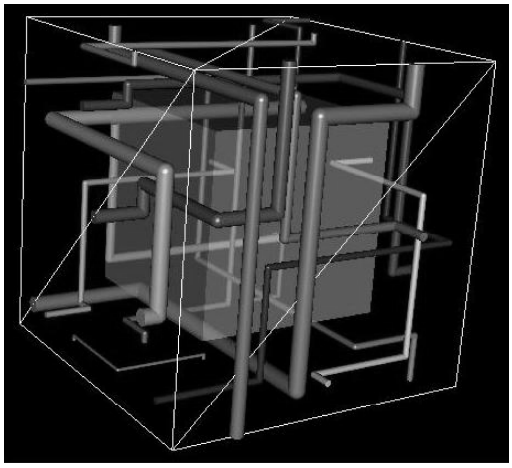


Fig.13 Result of 15 pipes arranged

4. Designing Modification Operator improving the evaluation value for obstacle

In this section, a Modification Operator on Obstacle (MOO) is designed to improve the evaluation value for obstacle. The effectiveness of the MOO is verified by applying it to the problem shown in Table 1 in Section 3.6.

4.1 Basic Concepts

A solution candidate that interferes with obstacles is modified by restructuring all of the pipes that interfere with obstacles. However, the regenerated solution candidate might be infeasible. Therefore, the MOC that makes an infeasible solution candidate feasible is used as well.

4.2 Algorithm of the MOO

The proposed algorithm for modifying a solution candidate is described as follows:

1. Apply the MOC to the infeasible solution candidate.
   (a) In the case all pipes do not interfere with all obstacles, it ends.
   (b) Otherwise, go to Step 2.
2. Restructure all pipes that interfere with obstacles.
   (a) In the case all pipes do not interfere with all obstacles, go to Step 3.
   (b) Otherwise, this step is repeated β times. Once this step is repeated β times, go to Step 3.
3. Repeat Steps 1 and 2 γ times. Once these steps are repeated γ times, apply the MOC to the solution candidate and it ends.

In this algorithm, β and γ are the avoidance number and the modification number, respectively. β and γ are constants that a designer should set according to the specific problem.

4.3 How to use the MOO

As well as the MOC, the MOO is used not only when generating an initial population, but also when applying a crossover and a mutation operator.

4.4 Effectiveness of the MOO

To confirm the effectiveness of the MOO, it was applied to the pipe arrangement design problem shown in Table 1 in Section 3.6. In this pilot experiment, the initial number of generating solution candidates was set to 20, the number of crossovers was set to 20, the mutation rate was set to 0.1, and β and γ of the MOO were set to 100 and 100, respectively.

Figs. 14 and 15 display the first Pareto solutions and Pareto solutions for the 100th generation, respectively. A solution with no interference with the obstacle was found in the first generation in the effect of the MOO, as shown in Fig.14. The evaluation value is almost the same whether XTG or UX was used as the crossover because the MOO restructures more pipes than the MOC and the MOO changes the properties of the original solution candidate more than the MOC. The time required to calculate the 100th generation using the MOO was almost the same as the time required to calculate the 1000th generation using the MOC. When the evaluation values for length are compared, the values using the MOC were better.

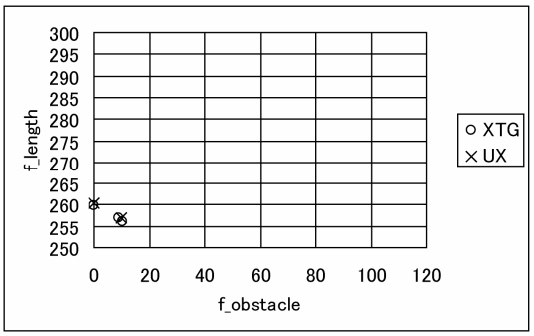


Fig.14 First Pareto solutions with MOO

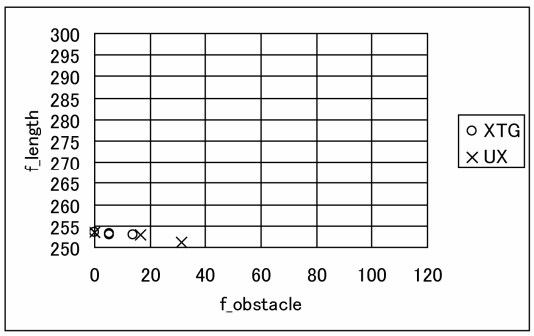


Fig.15 Pareto solutions in the 100th generation with MOO

4.5 Parameters of modification operators

[The slip number: α]

α is related to the number of pipes because it is used in conjunction with the MOC that make infeasible solution candidates feasible. Although it is desirable to have as large a value of α as possible that does not change the properties of the original solution candidate, a feasible solution might not be able to be generated easily.

[The avoidance number: β]

β is not related to the number of pipes, because it is used in conjunction with the MOO and it is related only to each pipe. This value depends on the size and the arrangement of the obstacle. If the value of β is too large, the probability of a useless search might increase because a pipe might not be able to completely avoid obstacles using the given pipe origin and endpoint coordinates.

[The modification number: γ]

γ is related to not only the number of pipes but also the size and the location of an obstacle because it is used to prevent pipes from interfering with each other and from interfering with obstacles. As with β, if the value of γ is too large, the probability of a useless search might increase.

Each parameter is highly dependent upon the given problem because they are related to not only the number of pipes but also to the diameter and coordinates of pipe origin and endpoints. Therefore, each parameter should be experimentally determined.

5. Conclusion

In this paper, a design problem for pipe arrangement in a small space where many pipes and obstacles co-exist was formulated. This problem includes large-scale numerical optimization and combinatorial optimization problems and two criteria. For these reasons, it was difficult to optimize the objective using usual optimization techniques such as Random Search. Therefore, the representation, crossover operator, mutation operator, and modification operator were introduced to transform infeasible solution candidates into feasible ones, and a multi-objective

Genetic Algorithm suitable for this problem was developed. The effectiveness of the proposed method was verified through several experiments. Moreover, another modification operator was proposed to improve the evaluation value for obstacle, and its effectiveness was demonstrated.

In the future, the work efficiency for installing a pipe should be considered, and the design should be integrated with the production process. Moreover, the total space in which the pipes is arranged should be minimized by searching for coordinates of origin and endpoints, although the origin and endpoint coordinates and the directions in which the pipes expand (direction vectors) are currently given. Because the search time increases as the number of pipes increases, improvements to the proposed algorithms should be continued.

Acknowledgement

The authors thank Mr. Eiske Ikezaki at Fukuoka Shipbuilding Co., Ltd for introducing the practical problem on pipe arrangement design kindly.

References

1. Coello, C. A. C., An Updated Survey of Evolutionary Multiobjective Optimization Techniques, State of the Art and Future Trends, Proc. Congress on Evolutionary Computation 1999, pp.3-13.
2. Corne, D.W., Knowles, J.D. and Oates, M.J., The Pareto Envelope-Based Selection Algorithm for Multiobjective Optimization, Parallel Problem Solving from Nature 6 (PPSN2000), pp.839-848.
3. Ono, I., 'Genetic Algorithms for Optimization Taking Account of Characteristics Preservation', Tokyo Institute of Technology 1997 (In Japanese).
4. Ono, I., Satoh, H. and Kobayashi, S., A Real-Coded Genetic Algorithm for Function Optimization Using the Unimodal Normal Distribution Crossover, Journal of the Japanese Society for Artificial Intelligence, Vol.14, No.6, pp.1146-1155 1998 (In Japanese).
5. Kobayashi, S., Yoshida, K., and Yamamura, M., Generating Pareto Optimal Decision Trees by Genetic Algorithms, Journal of the Japanese Society for Artificial Intelligence, Vol.11, No.5, pp.778-785 1996 (In Japanese).