Genetic Multi-Step Search in Interpolation and Extrapolation Domain

Yoshiko Hanada
Department of Knowledge Engineering and Computer Sciences, Doshisha University, Kyoto, Japan
hanada@mikilab.doshisha.ac.jp

Tomoyuki Hiroyasu
Department of Knowledge Engineering and Computer Sciences, Doshisha University, Kyoto, Japan
tomo@is.doshisha.ac.jp

Miki Mitsunori
Department of Knowledge Engineering and Computer Sciences, Doshisha University, Kyoto, Japan
mmiki@mail.doshisha.ac.jp

ABSTRACT

The deterministic Multi-step Crossover Fusion (dMSXF) is an improved crossover method of MSXF which is a promising method of JSP, and it shows high availability in TSP. Both of these crossover methods introduce a neighborhood structure and distance in each permutation problem and perform multi-step searches in the interpolation domain focusing on inheritance of parents' characteristic. They cannot work effectively when parents stand close each other since they search in interpolation domain. Therefore in the case of the MSXF, the Multi-step Mutation Fusion (MSMF), which is the multi-step search in the extrapolation domain, is combined as the supplementary search to improve its search performance. On the other hand, the search mechanism for acquisition of characteristics, such as MSMF, is not applied to dMSXF. In this paper, we introduce a deterministic MSMF (dMSMF) mechanism as complementary multi-step extrapolation search. We apply dMSXF+dMSMF to TSP and JSP, which have structural difference between their landscapes. Through the experiments it was shown that the deterministic multi-step search in interpolation/extrapolation domain performed effectively in combinatorial problems.

Categories and Subject Descriptors

G.2.1 [**Discrete Mathematics**]: Combinatorics - Permutations and combinations

General Terms

Algorithms

Keywords

Genetic Algorithm, Combinatorial Optimization, Local Search

1. INTRODUCTION

Genetic Algorithms (GA) are among the most effective approximation algorithms for optimization problems. GAs are applicable to a wide range of problems and have found many applications in combinatorial problems, such as the Traveling Salesman problem (TSP) and various scheduling problems.

A GA actualizes an effectual search using genetic operators for inheritance and acquisition of characteristics. These two classes of search, focusing on inheritance or acquisition, are called, respectively, the *interpolation search* and the *extrapolation search* by introducing a distance measure between solutions whose definitions are given in the next section [1]. In the general framework of GA, crossover plays a role in the former exploiting parents' characteristics, while the latter corresponds to mutations that explore outside the distribution of the population.

When we apply GA to a particular problem, especially for permutation problems, it is important to design a crossover method with emphasis on the heredity of favorable characteristics of parents. Various types of crossover have been proposed in consideration of problem-specific structures and features [2, 3, 4, 5, 6]. Deterministic Multi-step Crossover Fusion (dMSXF) [7] is a promising interpolation-directed crossover method based on neighborhood search. dMSXF is an improved version of Multi-step Crossover Fusion (MSXF) [8], which introduces a problem-specific neighborhood structure and a distance measure, and performs a multi-step neighborhood search between parents by a deterministic rule. This method can generate a wide variety of offspring between parents; however, a search mechanism for exploring the external domain is required because it does not work effectively when parents' characteristics are extremely similar to each other.

In this paper, we propose deterministic Multi-step Mutation Fusion (dMSMF) as a complementary search of dMSXF for exploring the extrapolation domain. Our method, dMSMF, performs a multi-step neighborhood search and it starts from the neighborhood of the parents and advances its search in the direction separate from them when parents are close to each other. Unlike a random mutation to be applied to perturb the population, it generates offspring in an efficient manner to gradually increase acquisition of characteristics that do not appear in the parents by multi-step search. We examined the effectiveness of our method in two problems, TSP and Job-shop Scheduling Problem (JSP) that is among the most difficult scheduling problem. dMSXF has been reported to show good search ability in TSP [7]. Here, we first show the effectiveness of incorporation of dMSMF into dMSXF on TSP, and then dMSXF and dM-

GECCO '07, July 7-11, 2007, London, England, United Kingdom.

SXF+ dMSMF were applied to JSP. These two problems are different with regard to the landscape; TSP has a big valley structure [9], while JSP is considered to have a complex multimodal landscape [10]. These experiments on two typical problems demonstrated the effectiveness of the deterministic multi-step search qualitatively in interpolation and extrapolation domains on combinatorial problems.

2. DETERMINISTIC MULTI-STEP CROSSOVER FUSION

2.1 Deterministic Multi-step Crossover Fusion

dMSXF and its original method MSXF that are multi-step searches based on a neighborhood search are proposed since incorporation of neighborhood searches into GAs is essential to adjust structural details of solutions in combinatorial problems [11]. Both these crossovers implements multi-step neighborhood searches from a parent p_1 in the direction approaching the other parent p_2. The previous method MSXF shows good search ability on JSP; however, it requires Metropolis criterion consisting of the temperature parameter T in the neighborhood search process. The parameter T has intensified impact on the performance of MSXF, moreover it is difficult to set due to dependence on the scale of fitness. dMSXF [7] is the improved crossover, which can be constructed using a neighborhood structure and a distance measure and searches in a deterministic manner using both quality of solutions and the distance measure.

The procedure of dMSXF is as follows and its search aspect is illustrated in Fig. 1. Here, $d(s_1, s_2)$ denotes the distance between solutions s_1 and s_2. The set of offspring generated by parents p_1, p_2 is indicated by $C(p_1, p_2)$.

Procedure of dMSXF

0. Let p_1, p_2 be parents and set their offspring $C(p_1, p_2) = \phi$.

1. k=1. Set the initial search point $x_1 = p_1$ and add x_1 into $C(p_1, p_2)$.

2. /Step k/ Prepare $N(x_k)$ composed of μ neighbors generated from the current solution x_k. $\forall y_i \in N(x_k)$ must satisfy $d(y_i, p_2) < d(x_k, p_2)$.

3. Select the best solution y from $N(x_k)$. Let the next search point x_{k+1} be y and add x_{k+1} into $C(p_1, p_2)$.

4. Set $k = k + 1$ and go to 2. until $k = k_{max}$ or x_k equals p_2.

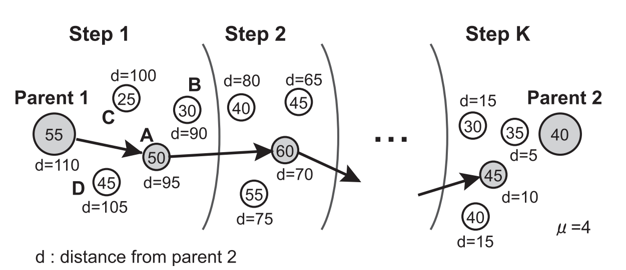

Figure 1: Aspects of dMSXF: dMSXF selects the best candidate A for a transition target

At step 2 of the procedure of dMSXF, every neighborhood candidate y_i $(1 \le i \le \mu)$ generated from x_k must be closer to p_2 than x_k. In addition, dMSXF necessarily moves its transition toward p_2 even if all solutions in $N(x_k)$ are inferior to the current solution x_k. dMSXF requires two parameters, k_{max} and μ. k_{max} is the number of steps of neighborhood search and μ is the number of generated solutions at each step of the neighborhood search. In the procedure of dMSXF, at most $k_{max} * \mu$ solutions would be generated, and $C(p_1, p_2)$ is comprised of the best neighbor solutions, i.e., $\{x_1, x_2, \cdots, x_{k_{max}}\}$. The generation-alternation model of p_1, p_2 and $C(p_1, p_2)$ used is described in section 3.2.

2.2 Aspects of dMSXF from the perspective of interpolation/extrapolation domain

The entire solution space can be divided into two domains - the *interpolation domain* and the *extrapolation domain* - under the definition of a distance measure in discrete space [1]. Once these domains are defined, we can comprehend where dMSXF searches. Given a distance measure d, the interpolation domain D_{in} and the extrapolation domain D_{ex} are defined as follows, where S denotes the entire solution space, and their aspects are illustrated in Fig. 2. In addition, this discussion can be seen not only in GAs but in Path-relinking (PR) [12], which is often adopted as an operator of Scatter Search [13, 14].

$$D_{in} = \{s \in S \mid d(s, p_1) \le d(p_1, p_2) \text{ and } d(s, p_2) \le d(p_1, p_2)\}$$
$$D_{ex} = \{s \in S \mid d(s, p_1) > d(p_1, p_2) \text{ or } d(s, p_2) > d(p_1, p_2)\}$$

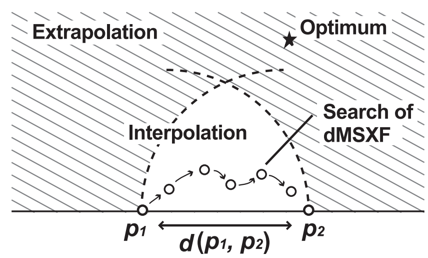

Figure 2: Interpolation and extrapolation domains

dMSXF searches inside D_{in} in Fig. 2, and to search globally, an exploration mechanism for D_{ex} is required to enhance its efficacy, especially for complicated problems. One choice as an extrapolation search is a random mutation method to be adopted to perturb the population by generating offspring randomly. However, it cannot work due to the broad distribution of offspring against the population in the later stages of the search. On the other hand, our method proposed in the next section was designed based on a stochastic local search on a well-defined extrapolation domain, which targets the acquisition of lost or lacking characteristics in the population.

3. PROPOSAL OF DETERMINISTIC MULTI-STEP MUTATION FUSION

3.1 Deterministic Multi-step Mutation Fusion

In this study, we propose a complementary search method to dMSXF, deterministic Multi-step Mutation Fusion (dMSMF). dMSMF is defined in a problem-independent manner based on a neighborhood search. In contrast to the search with approaching direction of dMSXF, dMSMF advances the search in the direction that separates from the parents' neighborhood using the deterministic rule as follows:

Procedure of dMSMF

0. Let p_1, p_2 be parents and set their offspring $C(p_1, p_2) = \phi$.

1. l=1. Set the initial search point $x_1 = p_1$.

2. /Step l/ Prepare $N(x_l)$ composed of λ neighbors generated from the current solution x_l. $\forall y_i \in N(x_l)$ must satisfy both $d(y_i, p_1) > d(x_l, p_1)$ and $d(y_i, p_2) > d(x_l, p_2)$.

3. Select the best solution y from $N(x_l)$. Let the next search point x_{l+1} be y and add x_{l+1} into $C(p_1, p_2)$.

4. Set $l = l + 1$ and go to 2. until $l = l_{max}$.

At the step 2 of dMSMF, every neighborhood candidate y_i $(1 \leq i \leq \lambda)$ generated from x_l is restricted to satisfy both $d(y_i, p_1) > d(x_l, p_1)$ and $d(y_i, p_2) > d(x_l, p_2)$. Even if all solutions in $N(x_l)$ are inferior to the current solution x_l, the transition to a solution in $N(x_l)$ is necessarily accepted.

The search aspect is illustrated in Fig. 3. In this procedure, at most l_{max} * λ solutions would be generated.

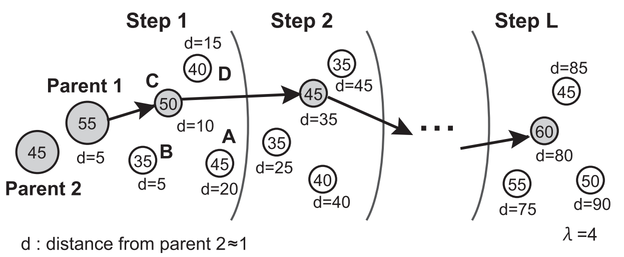

Figure 3: Aspects of dMSMF : dMSMF selects the best candidate C

In applying MSXF to JSP, Multi-step Mutation Fusion (MSMF) has been introduced as a complementary search of MSXF [8]. Our method can be considered an improvement of MSMF, which does not require the Metropolis criterion. In accordance with the generation-alternation model described in section 3.2, the parent p_1 is replaced with the best solution in $C(p_1, p_2)$ after termination of the procedure above. dMSMF does not include x_1 (=p_1) in $C(p_1, p_2)$, while dMSXF does. Therefore, dMSMF is forced to replace p_1 with other obtained solutions. This is because dMSMF must alleviate the bias of the population.

A problem-specific neighborhood structure and distance measure should be defined to apply both dMSMF and dMSXF to each problem. In addition, it is necessary to design a method to generate neighborhood solutions to satisfy the conditions required by step 2 in each method.

Hereafter, we represent the interpolation-directed multi-step search dMSXF as Inter-MSX, while dMSXF that searches the extrapolation domain is denoted by Extra-MSM.

3.2 Procedure of GA with multi-step search in interpolation and extrapolation domain

We outline an application of both Inter-MSX and Extra-MSM to GA as follows. This model bases on the generation-alternation model that showed effectiveness in the original paper of Inter-MSX (dMSXF) [7].

Flow of GA

1. Generate the initial population composed of N_{pop} random solutions, individuals, $\{x_1, x_2, \cdots, x_{N_{pop}}\}$.

2. Reset indexes $\{1, 2, \cdots, N_{pop}\}$ to each individual randomly.

3. Select N_{pop} pairs of parents (x_i, x_{i+1}) $(1 \leq i \leq N_{pop})$ where $x_{N_{pop}+1} = x_1$.

4. For each pair (x_i, x_{i+1}), if $d(x_i, x_{i+1})$ is smaller than predefined value d_m, apply Inter-MSX, otherwise apply Extra-MSM to it.

5. For each pair (x_i, x_{i+1}), select the best individual c from offspring $C(x_i, x_{i+1})$ generated by parents (x_i, x_{i+1}) and replace the parent x_i with c.

6. Go to 2 until some terminal criterion is satisfied, e.g., generations and/or the number of evaluations.

The effectiveness of Extra-MSM is supposed to depend on features of the problem, such as aspects of the landscape. Here, we examined the effectiveness of incorporation of Extra-MSM into Inter-MSX in TSP and JSP; the former is one of the problems that satisfy the big valley hypothesis [9], while the landscape of the latter is globally multimodal and conforms to the UV structure hypothesis [10]. It is thought to be easy for GA to find the global optimum in problems conforming to the big valley hypothesis. On the other hand, problems corresponding to the UV structure hypothesis have a number of influential local optimal solutions by which populations of GA tend to be trapped. The application and experiments of both methods for TSP are described in sections 4 and 5, and those for JSP are shown in sections 6 and 7.

4. APPLICATION OF INTER-MSX AND EXTRA-MSM FOR TSP

In this section, we describe how to apply Inter-MSX and Extra-MSM to TSP.

4.1 Neighborhood and Distance

Inter-MSX has already been applied to TSP, in which the distance measure is defined as the number of different edges between parents, and the neighborhood structure based on the *AB-cycle* generated during the procedure of EAX is adopted [7]. EAX is a state-of-the-art crossover specialized for TSP, and its element, AB-cycle, can be considered as a building block that adequately perceives the characteristics of TSP. The AB-cycle is defined as a closed loop on the set of edges composed of tours of both p_1 and p_2, which can be generated by alternately tracing the edges of p_1 and p_2.

Inter-MSX has been shown to perform very well in TSP using the above definitions. Therefore, we developed Extra-MSM based on the design of Inter-MSX.

4.2 Inter-MSX in TSP

We describe step 2 of previously reported Inter-MSX [7]. Inter-MSX generates neighborhood candidates $N(x_k)$ of the transitional solution x_k as follows, and an example of this procedure is shown in Fig. 4.

Formation of neighborhood candidates of x_k

0. Let p_2 be one of parents and set the neighborhood candidates $N(x_k) = \phi$.

1. Pick one of AB-cycles between x_k and p_2.

2. Generate an intermediate individual x'_k by applying the AB-cycle to x_k in the **XOR** manner, i.e. by removing edges of x_k included in the AB-cycles from x_k, and adding edges of p_2 in the AB-cycles to x_k. Here, x'_k is consisting of subtours and it is not a complete solution yet.

3. Modify x'_k to a valid tour by merging its sub-tours, and add it into $N(x_k)$ as a neighbor of x_k.

4. Go to 1 until $|N(x_k)|=\mu$ is satisfied.

The generation method of x'_k is equivalent to the procedure of EAX between x_k and p_2 with one of the AB-cycles. A neighborhood x'_k that possesses more edges of p_2 than x_k, i.e., $d(x'_k, p_2) < d(x_k, p_2)$, is necessarily generated by applying one AB-cycle.

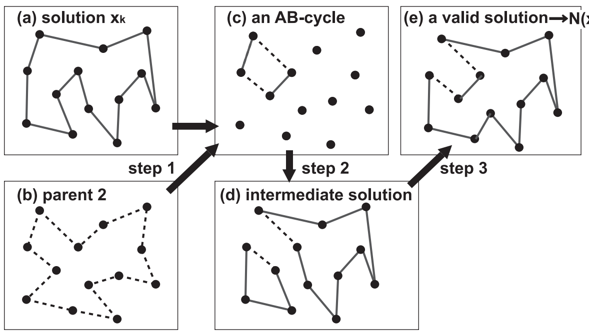

Figure 4: An example of generating a neighborhood: (a) x_k and (b) parent 2 are given, an AB-cycle (c) of (a) and (b) is generated. An intermediate solution (d) is created by mixing (a) and (c) with XOR manner. A valid solution (e) is generated by modifying (d).

The design of the complementary search method, Extra-MSM, is introduced in the next section. For TSP, which is thought to have a big valley structure, the global optimal solution can be obtained easily by continuing combining characteristics observed in the population from the initial generation. Hence, Inter-MSX, which is an interpolation-directed search, has a greater possibility of finding the global optimum in accordance with increases in the population size. In contrast, this is difficult for a small population against instances of benchmarks due to lack of favorable characteristics in the initial population. Thus, Extra-MSM should be designed to cover lacking edges and lost edges in the population.

4.3 Design of Extra-MSM in TSP

When we design a method for generation of neighbor solutions in the extrapolation domain of the parents, "a variational scale" should be considered. Here, we express the distance of solutions s_1 and s_2 as a variational scale, where s_2 denotes a solution generated by applying neighborhood search. For example, the variational scale by one step in Inter-MSX is $d(p_1, p_2)/k_{max}$. Operations, such as *2-change*, of which the variational scale is too small[1] against the neighborhood search of Inter-MSX, are not appropriate as one step of Extra-MSM because it is difficult to generate solutions outside the interpolation domain. Here, we adopted a simple method to maintain the variational scale. In this procedure, the variational scale by one step in Extra-MSM is approximately $d(p_1, p_3)/l_{max}$.

Design of Extra-MSM

0. Let p_1, p_2 be parents.

1. Prepare a random tour, solution, as a new individual and let this solution be p_3. Apply *2-opt method* to p_3.

2. Apply l_{max} steps of the neighborhood search of Inter-MSX from p_1 to p_3. Here, neighbor candidates $N(x_l)$ of the transitional solution x_l consist of solutions satisfying both $d(y_i, p_1) > d(x_l, p_1)$ and $d(y_i, p_2) > d(x_l, p_2)$.

Offspring between p_1 and p_2, $C(p_1, p_2)$, consist of $x_2, x_3, \cdots, x_{l_{max}}$ obtained by the above procedure. For each pair of parents for Extra-MSM, p_3 is newly generated to search in the extrapolation domain.

5. SEARCH PERFORMANCE OF INTER-MSX+EXTRA-MSM IN TSP

The effects of incorporation of Extra-MSM into Inter-MSX in TSP were examined using the medium-scale benchmarks from TSPLIB[2]. To confirm the superior ability in the search in the well-defined extrapolation domain, we compared Extra-MSM with a mutation method denoted by Inter-MSX+Extra-MSM and Inter-MSX+Mutation, respectively. Both Inter-MSX and Extra-MSM require the number of steps in the neighborhood search, k_{max} and l_{max}, and the number of neighbor candidates, μ and λ, as parameters. Here, as Extra-MSM substantially implements Inter-MSX outside the parents, we set k_{max}=l_{max} and $\mu = \lambda$. In addition, we used k_{max}=5 and μ=8, as recommended previously [7]. For each pair, p_1 and p_2, for reproduction of offspring, Extra-MSM was applied instead of Inter-MSX when the distance between the parents was smaller than $N_{city} * a$ where N_{city} denotes the total number of cities of the instance. We set a=0.05 for instances, in which N_{city} was below vm1748, and a=0.02 was used for other instances.

For Inter-MSX+Mutation, we adopted Extra-MSM with l_{max}=1 as a mutation method with the exception that neighborhood candidates were generated regardless of interpolation/extrapolation domain. This is because normal mutation methods, such as 2-change, are anticipated to be unproductive operations due to the diminutive variational scale. The mutation generated $l_{max} * \lambda$ offspring, and replaced the parent p_1 with the best solution of the offspring.

Table 1 shows the number of trials that obtained the optimum (#opt), the average number of evaluations to acquire the optimum (#eval), and the average error (%) from 30 trials. Here, we set the population size to 200 for *pcb*3038 and *fl*3795, and 100 for others. In addition, these three methods were terminated after 200 generations of GA for *pr*2392, 300 generations for *fl*3795, and 100 generations for other instances.

As shown in Table 1, both Extra-MSM and the mutation method showed a high possibility of finding the optimum solution, which indicates that incorporation of extrapolation factors leads to improvement of search performance in TSP. Moreover, Extra-MSM enhanced the performance of Inter-MSX compared with the mutation method, which generates offspring randomly. In comparison of #eval, to obtain the optimum, the method more highly focusing on the extrapolation domain requires more evaluations. These observations indicate that the search performance improves, while the convergence speed is reduced by the extrapolation search.

[1]The variational scale of *2-change* is 2.

[2]TSPLIB: http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/

Table 1: Performance of Inter-MSX + Extra-MSM on benchmarks of TSP

Instance	Inter-MSX		Inter-MSX+Mutation		Inter-MSX+Extra-MSM	
	#opt (#eval)	err(%)	#opt (#eval)	err(%)	#opt (#eval)	err(%)
pr439	26 (3.5x10^4)	0.002	30 (3.7x10^4)	0.0	30 (3.8x10^4)	0.0
att532	7 (0.8x10^5)	0.034	11 (1.5x10^5)	0.027	13 (1.8x10^5)	0.023
rat575	10 (0.8x10^5)	0.015	17 (1.2x10^5)	0.009	23 (1.6x10^5)	0.004
rat783	18 (8.8x10^4)	0.012	25 (9.2x10^4)	0.008	28 (9.7x10^4)	0.005
pr1002	15 (1.2x10^5)	0.019	23 (1.6x10^5)	0.012	25 (1.9x10^5)	0.006
pcb1173	11 (1.4x10^5)	0.007	12 (1.9x10^5)	0.005	19 (2.2x10^5)	0.004
vm1748	2 (1.8x10^5)	0.054	7 (3.1x10^5)	0.047	10 (4.0x10^5)	0.046
pr2392	14 (2.2x10^5)	0.010	16 (2.7x10^5)	0.008	24 (3.0x10^5)	0.002
pcb3038	1 (7.6x10^5)	0.007	3 (8.3x10^5)	0.006	4 (9.8x10^5)	0.006
fl3795	14 (1.8x10^6)	0.022	16 (1.9x10^6)	0.017	18 (1.9x10^6)	0.017

The number of trials out of 30 that reached the optimum, average number of evaluations needed, and average error

Table 2: Influence of population size in Inter-MSX + Extra-MSM

Instance	$N_{pop} = 50$		$N_{pop} = 100$		$N_{pop} = 200$		$N_{pop} = 300$	
	Int-MSX	+Ext-MSM	Int-MSX	+Ext-MSM	Int-MSX	+Ext-MSM	Int-MSX	+Ext-MSM
rat575	1	9	10	23	19	30	27	30
rat783	0	22	18	28	29	30	30	30
pr1002	2	14	15	25	29	30	30	30

The number of trials out of 30 that reached the optimum

Next, we discuss the influence of population size, N_{pop}, to highlight the impact of Extra-MSM. Table 2 shows a comparison between Inter-MSX and Inter-MSX+Extra-MSM under N_{pop}=50, 100, 200, and 300.

As shown in Table 2, when N_{pop} is set sufficient for the scale of the instances, Inter-MSX+Extra-MSM performs the search equally well as Inter-MSX. Moreover, Inter-MSX+Extra-MSM shows superior performance to Inter-MSX when N_{pop} is small.

From these results, we conclude that combination of interpolation/extrapolation multi-step search is effective in TSP, which has a big valley structure.

6. APPLICATION OF INTER-MSX AND EXTRA-MSM FOR JSP

In this section, we discuss the effectiveness of both Inter-MSX and Inter-MSX+Extra-MSM in JSP, which is a globally multimodal problem for which there would be a strong requirement for extrapolation searches.

6.1 Neighborhood and Distance

We cover the *active schedule* as the search space and adopt the *active CB neighborhood* [8] that has been used in MSXF and EDX [1]. The active CB neighborhood is composed of the solutions generated by shifting an operation inside a *critical block*, which are parts of the *critical path*, to either the head or the end of the block on a solution. In addition, these solutions of the active CB neighborhood are corrected to be active schedules using the *GT algorithm* proposed elsewhere [15]. We then adopt the I_2 distance [1] based on the absolute positions of operations belonging in each machine due to its high affinity with the active CB neighborhood.

With M machines and N jobs, I_2 distance on job i of the schedule s_a and s_b, $I_{2i}(s_a, s_b)$, and I_2 distance of these schedules, $I_2(s_a, s_b)$ are defined as equations (1) and (2), respectively. In these equations, $o(p, q)$ denotes the operation to be processed by the machine q and belonging to the job p. The set of operations belonging to job i is represented by $J_i(= \{o(i, k)|k = 1, \cdots, M\})$. $L(o)$ denotes the absolute position of operation o. An example for this distance metric can be found in [1].

$$I_{2i}(s_a, s_b) = \Sigma_{k=1}^{M} |L(o_a(i, k)) - L(o_b(i, k))| \quad (1)$$

$$I_2(s_a, s_b) = \Sigma_{k=1}^{N} I_{2k}(s_a, s_b) \quad (2)$$

6.2 Design of Inter-MSX

Here, a generation method of neighborhood solutions at step 2 in the procedure described in the section 2.1 is designed. At step 2 of Inter-MSX, every neighborhood candidate y_i $(1 \le i \le \mu)$ generated from x_k is restricted to satisfy $d(y_i, p_2) < d(x_k, p_2)$. To satisfy the condition, Inter-MSX generates the intermediate solution x'_k and active CB neighbors of x'_k as follows, and an example of this procedure is shown in Fig. 5.

Formation of neighborhood candidates of x_k

1. Select a job J_i randomly, but with a bias in favor of J_i with a large I_{2i} distance.

2. Copy the operations belonging to J_i chosen at step 1 from parent p_2 into the intermediate solution x'_k, preserving their loci.

3. Copy the operations of all jobs except for J_i, from x_k into x'_k, preserving their orders.

4. Generate (μ-1) active CB neighborhoods of x'_k.

These offspring, x'_k and neighborhoods of x'_k, construct $N(x_k)$. This procedure generates the intermediate solution

Table 3: Performance of Inter-MSX on benchmarks of JSP

Instance	JOX	Inter-MSX			
		$k_{max} = 4$	$k_{max} = 6$	$k_{max} = 8$	$k_{max} = 10$
ft10	30 (1.4x10^5)	28 (1.2x10^5)	30 (1.1x10^5)	30 (1.0x10^5)	30 (0.9x10^5)
ft20	12 (1.6x10^5)	10 (3.1x10^5)	16 (4.5x10^5)	19 (4.9x10^5)	24 (5.8x10^5)
abz5	1 (1.8x10^5)	19 (1.5x10^5)	21 (1.2x10^5)	23 (0.9x10^5)	27 (1.0x10^5)

The number of trials out of 30 that reached the optimum, average number of evaluations needed

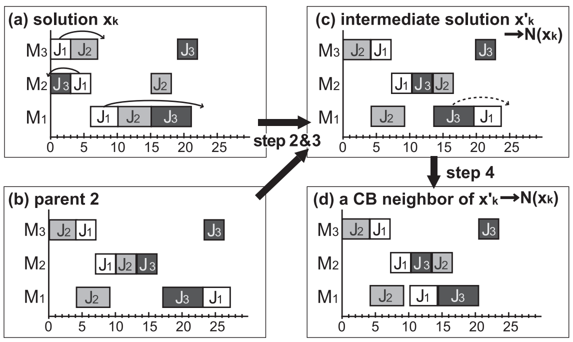

Figure 5: An example of generating neighborhoods: (a) x_k and (b) parent 2 are given, intermediate solution x'_k (c) is created from x_k by inheriting J_1 of parent 2, and a CB neighborhood (d) is generated from x'_k.

x'_k in consideration of technological sequence of machines to be processed on each job, since the swap between two operations on a certain machine, which is adopted in PR [12], is difficult to generate feasible solutions due to strong dependency among machines.

At each step k ($1 \le k \le k_{max}$), Inter-MSX selects the best solution y among x'_k and (μ-1) neighborhoods of x'_k and moves its transition from x_k to y.

6.3 Design of Extra-MSM

We designed Extra-MSM for multimodal problems, the function of which should search across valleys of local optima.

Here, formation of neighborhood candidates of x_l at step 2 of Extra-MSM mentioned in the section 2.1 is described. Every neighborhood candidate y_i ($1 \le i \le \lambda$) generated from x_l must satisfy both $d(y_i, p_1) > d(x_l, p_1)$ and $d(y_i, p_2) > d(x_l, p_2)$. Extra-MSM first generates the mutated solution x'_l to advance its search in the direction to obtain candidates that have larger I_2 distance. It then generates (λ-1) active CB neighbors of x'_l. $N(x_l)$ composed of x'_l and neighborhoods of x'_l is generated as follows:

Formation of neighborhood candidates of x_l

1. Set the mutated solution $x'_l = x_l$ and select a job J_i randomly.

2. On the mutated solution x'_l, shift all operations belonging to J_i leftward or rightward randomly.

3. Generate (λ-1) active CB neighborhoods of x'_l.

At each step l ($1 \le l \le l_{max}$) in Extra-MSM, the best solution y among x'_l and (λ-1) neighborhoods of x'_l is selected as the next solution x_{l+1}.

7. SEARCH PERFORMANCE OF INTER-MSX+EXTRA-MSM IN JSP

7.1 Performance of Inter-MSX in JSP

We examined the performance of Inter-MSX on JSP using the benchmarks $ft10$, $ft20$, and $abz5$. $ft10$ and $abz5$ have 10 jobs and 10 machines and $ft20$ has 20 jobs and 5 machines. The population size was 100 and the termination was set to 200 generations. Here, we set μ=5 and k_{max}=4, 6, 8, 10. In the experiments described in this paper, the *LR method* [8] was used for evaluation of individuals. In addition, we generated μ active CB neighbor solutions of the individuals on evaluation and replaced them with the best solution of the neighborhoods to improve the performance.

Table 3 shows the number of trials that obtained the optimum and the average number of evaluations to acquire the optimum. These results are from 30 trials. To compare Inter-MSX with another interpolation-directed crossover, we show the results of inter machine JOX [5], which is one of promising crossovers on JSP. In the comparative method, the job-based shift change [5] was applied as the mutation, and CCM [5] was adopted for the generation alternation model focusing on inheritance of parents' characteristics. The population size was the same as that of Inter-MSX and each pair at the crossover generated 20 offspring.

Table 3 shows the superiority of Inter-MSX to JOX in terms of successful trials and its performance becomes good in accordance with increases in k_{max}. Inter-MSX performed well without relying on setting of the parameter μ in preparative experiments. Inter-MSX was shown to be effective on TSP, which has a big valley structure. In addition, it showed good ability on JSP, which is one of the problems with a global multimodal landscape. The above results confirm the assertion in [7] that the multi-step search with definition of both a distance measure and a neighborhood structure enables efficient searches in combinatorial problems.

7.2 Performance of Inter-MSX+Extra-MSM

7.2.1 *Efficacy of extrapolation multi-step search*

We examined the search performance of incorporating Extra-MSM with Inter-MSX. To highlight the effectiveness of the multi-step search in the extrapolation domain, we compared Inter-MSX+Extra-MSM with Inter-MSX+Mutation.

The benchmarks $ft10$, $ft20$, and $abz5$ are applied for this examination. We set k_{max}=l_{max}=5 for $ft10$ and $abz5$, and k_{max}=l_{max}=10 for $ft20$. The values of the parameters of GA were the same as in section 7.1. To each pair, p_1 and p_2, for reproduction, Extra-MSM was applied instead of Inter-MSX in two cases as follows: 1) I_2 distance between parents is smaller than N_{op} * 0.1 where N_{op} denotes the total number

Table 4: Performance of Inter-MSX+Extra-MSM on benchmarks of JSP

	Inter-MSX		Inter-MSX+Mutation		Inter-MSX+Extra-MSM	
Instance	#opt	err(%)	#opt	err(%)	#opt	err(%)
ft10	29 (1.4x10^5)	0.022	29 (1.2x10^5)	0.025	30 (1.2x10^5)	0.0
ft20	24 (5.8x10^5)	0.19	25 (4.8x10^5)	0.11	30 (5.3x10^5)	0.0
abz5	19 (1.4x10^5)	0.16	22 (2.1x10^5)	0.12	30 (1.8x10^5)	0.0

The number of trials out of 30 that reached the optimum, average number of evaluations needed, and average error

of operations of the intended instance, 2) The fitnesses of p_1 and p_2 are the same.

Table 4 shows the number of trials that obtained the optimum, the average number of evaluations to acquire the optimum, and the average error (%) from 30 trials. In this examination, for Inter-MSX+Mutation, the job-based shift change [5] was adopted as the mutation method. It generated the same number of offspring as Extra-MSM, i.e., l_{max} * λ, and replaced the parent p_1 with the best solution of the offspring.

From Table 4, we can see that both Extra-MSM and the mutation, i.e., extrapolation factors, improve search performance of Inter-MSX, as in the case of TSP. In addition, incorporation of Extra-MSM is more effective than applying the mutation that generates solutions with no consideration of the interpolation domain and the extrapolation domain. These results indicate the importance of precise search mechanism in the well-defined extrapolation domain.

7.2.2 Analysis of Extra-MSM

Inter-MSX obtains the optimum with higher probability at small instances than other interpolation-directed crossover algorithms. However, JSP has a complex landscape consisting of a number of influential local optimal solutions and GA has the possibility of lapsing into a local optimum on large instances. It is difficult to find the global optimum standing in another valley once GA with an interpolation-directed crossover progresses its search into the valley of a local optimum.

We use the instance *abz*5 supposed to possess serious UV structure and the behavior of GA with initial populations biased toward an influential local optimum illustrated in Fig. 6 is elucidated. Here, we use the local optimum of which fitness is 1236 and the distance from the global optimum is d = 114. It denotes $l_{opt}(1236)$ and the global optimum denotes g_{opt}.

Table 5 shows the convergence tendency of GA with Inter-MSX using initial populations composed of individuals satisfying, respectively, the distance $d < 100$, $d < 120$, and $d < 130$ from g_{opt}. It also shows the results of Inter-MSX+Extra-MSM with the initial population restricted to $d < 100$. These results are from 50 runs. The values of the parameters of GA are the same as in sections 7.1.

Table 5: Convergence tendency of GA

	l_{opt}(1236)	g_{opt}	another l_{opt}
$d < 100$ (Inter-MSM)	**48**	**1**	**1**
$d < 120$ (Inter-MSM)	38	6	6
$d < 130$ (Inter-MSM)	29	12	9
$d < 100$ (Inter-MSX +Extra-MSM)	**21**	**28**	**1**

The number of trials out of 50 that reached each solution

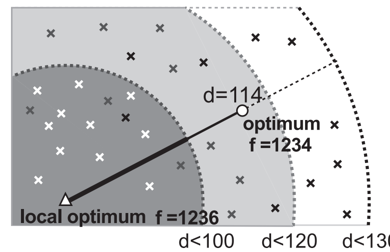

Figure 6: Generation of biased population: Each initial solution generated with a few applications of mutation from l_{opt}(1236)

As shown in Table 5, Inter-MSX with the population restricted to the domain $d < 100$ that does not include the global optimal solution is difficult to find. The populations initialized in the domain $d < 120$ and $d < 130$ covering the global optimal solution but with convergence to $l_{opt}(1236)$ can find the optimal solution or another local optimum several times. It is quite difficult for interpolation-directed Inter-MSX to obtain g_{opt} although $l_{opt}(1236)$ is not some distance from g_{opt} once the population begins to converge toward a local optimum. In contrast, it is highly possible to find g_{opt} by incorporating Extra-MSM even if the initial population does not cover g_{opt}. These results indicate that the extrapolation-directed search is also essential for problems with complex landscapes.

7.2.3 Performance in 10 tough problems

We examined the search performance of Inter-MSX+Extra-MSM on 10 tough problems as relatively large instances. The results confirmed the superiority of our method in comparison with other multi-step search methods.

Here, we set the population size to 400, μ=λ=20, k_{max}=20, and l_{max}=10. GA is terminated when 1) no progress of best fitness is found within 200 generations or 2) the total number of evaluations is 5.0x10^7. The conditions of applying Extra-MSM instead of Inter-MSX were same as in the section 7.1.

Table 6 compares Inter-MSX+Extra-MSM and Inter-MSX. These results are the best fitness in 10 trials or the number of trials finding the optimum, and average fitness and worst fitness. To compare our method with other promising methods, we draw the results of MSXF+MSMF [8] and JOX+EDX [1] that are crossovers consisting of interpolation and/or extrapolation multi-step searches. The number of evaluations used, 5.0x10^7, is the termination criterion of JOX+EDX. We confined the comparison to the best fitness because other indicators of performance, such as the number of evaluations to acquire the optimum, were not described previously [1] and [8].

Table 6: Performance of Inter-MSX+Extra-MSM on 10 tough problems

	*opt/UB	Inter-MSX+Extra-MSM best(#opt)	avg	wst	Inter-MSX best(#opt)	avg	wst	MSXF +MSMF best(#opt)	JOX +EDX best(#opt)
abz7	*656	**658**	665.3	668	664	666.6	669	678	670
abz8	665	**668**	670.4	675	670	672.1	676	686	683
abz9	679	**679**	685.9	689	686	687	689	697	686
la21	*1046	1047	1051.6	1053	1052	1052.4	1055	**1046(9/30)**	1046(1/10)
la24	*935	**935(5/10)**	936.5	938	935(1/10)	939.2	941	935(4/30)	935(4/10)
la25	*977	**977(7/10)**	978.1	984	977(1/10)	980.8	984	977(9/30)	977(4/10)
la27	*1235	**1235(5/10)**	1237.8	1242	1235(1/10)	1242.6	1250	1235(1/30)	1236
la29	*1152	**1154**	1164	1167	1163	1166.6	1168	1166	1167
la38	*1196	**1196(10/10)**	1196	1196	1196(2/10)	1200.7	1206	1196(21/30)	1196(1/10)
la40	*1222	1224	1227	1234	1225	1230	1240	1224	1224

The best fitness, number of trials out of 10 that reached the optimum, average, and worst fitness

From Table 6, we can see that Inter-MSX performs well in terms of accuracy of solutions. Moreover, application of Extra-MSM considerably improves search performance and shows superiority to both MSXF and EDX, both of which perform more effectively than other approximation algorithms, such as PR [12], SA [16] and TS [17].

8. CONCLUSIONS

The deterministic Multi-step Crossover Fusion (dMSXF), denoted here as Inter-MSX, is a promising crossover method that can be constructed by introducing a problem-specific neighborhood structure and a distance measure. This method performs a neighborhood search using the deterministic rule composed of only a distance measure in a problem-independent manner. However, Inter-MSX does not work effectively when parents are close to each other as it searches in the interpolation domain focusing on inheritance of parent characteristics. In this paper, we proposed the deterministic Multi-step Mutation Fusion (dMSMF), also denoted as Extra-MSM, as a complementary search of Inter-MSX. Extra-MSM performs a multi-step search in the extrapolation domain to acquire characteristics that do not appear in the parents. We designed Inter-MSX and Extra-MSM for both TSP and JSP. We first demonstrated the effectiveness of incorporation of Extra-MSM in TSP, which is a big valley structure problem. Next, we investigated the efficacy of Inter-MSX and Extra-MSM in JSP, which is another problem class with a complicated multimodal landscape. The results demonstrated the superiority of our method to other methods. From these results, we qualitatively confirmed that the deterministic multi-step search in interpolation and extrapolation domains is effective in combinatorial problems. In future studies, statistical analyses of Inter-MSX and Extra-MSM are required, and we should determine the efficiencies of these methods in a quantitative manner.

9. REFERENCES

[1] Sakuma, J. and Kobayashi, S.: Extrapolation-Directed Crossover for Job-shop Scheduling Problems: Complementary Combination with JOX, *Proc. of GECCO 2000*, pp. 973–980 (2000).

[2] Kobayashi, S., Ono, I. and Yamamura, M.: An Efficient Genetic Algorithm for Job Shop Scheduling Problems, *Proc. of 6th Int. Conf. on Genetic Algorithms*, pp. 506–511 (1995).

[3] Yamamura, M., Ono, I. and Kobayashi, S.: Emergent Search on Double Circle TSPs using Subtour Exchange Crossvoer, *Proc. of 1996 IEEE International Conference on Evolutionary Computation*, pp. 535–540 (1996).

[4] Shi, G. and Sannomiya, N.: A New Encoding Scheme for Solving Job Shop Problem by Genetic Algorithm, *Proc. of 35th IEEE Conf. on Decision and Control*, pp. 4395–4400 (1996).

[5] Ono, I. and Kobayashi, S.: A Genetic Algorithm Taking Account of Characteristics Preservation for Job Shop Scheduling Problems, *Proc. of the International Conference on Intelligent Autonomous Systems 5*, pp. 711–718 (1998).

[6] Nagata, Y.: New EAX crossover for large TSP instances, *Proc. of Parallel Problem Solving fron Nature, PPSN IX*, pp. 372–381 (2006).

[7] Ikeda, K. and Kobayashi, S.: Deterministic Multi-step Crossover Fusion: A Handy Crossover for GAs, *Proc. of Parallel Problem Solving fron Nature, PPSN VII*, pp. 162–171 (2002).

[8] Yamada, T. and Nakano, R.: Scheduling by Genetic Local Search with Multi-Step Crossover, *Proc. of Parallel Problem Solving fron Nature, PPSN IV*, pp. 960–969 (1996).

[9] Boese, K. D.: Cost Versus Distance in the Traveling Salesman Problem, Technical report, TR-950018, UCLA CS Department (1995).

[10] Ikeda, K. and Kobayashi, S.: GA Based on the UV-Structure Hypothesis and Its Application to JSP, *Proc. of Parallel Problem Solving fron Nature, PPSN VI*, pp. 273–282 (2000).

[11] Yagiura, M. and Ibaraki, T.: *Genetic and Local Search Algorithms as Robust and Simple Optimization Tools*, Kluwer Academic Publishers, MA, USA (1996).

[12] Aiex, R. M., Binato, S. and Resende, M. G. C.: Parallel GRASP with path-relinking for job shop scheduling, *Parallel Computing*, Vol. 29, pp. 393–430 (2003).

[13] Glover, F.: Genetic algorithms and scatter search: unsuspected potentials, *Statistics and Computing 4*, pp. 131–140 (1994).

[14] Marti, R., Laguna, M. and Glover, F.: Principles of Scatter Search, *European Journal of Operational Research*, Vol. 169 (2), pp. 359–372 (2006).

[15] Giffler, B. and Thompson, G.: Algorithms for Solving Production Scheduling Problems, *Operations Research*, Vol. 8, pp. 487–503 (1960).

[16] Aarts, E., van Laarhoven, P., Lenstra, J. and Ulder, N.: A Computational Study of Local Search Algorithms for Job-shop Scheduling, *ORSA J. on Comput*, Vol. 6, No. 2, pp. 118–125 (1994).

[17] Nowicki, E. and Smutnicki, C.: A Fast Taboo Search Algorithm for the Job-shop Scheduling Problem, *Management Science*, Vol. 42, No. 6, pp. 797–813 (1996).