2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Find Robust Solutions over Time by Two-layer Multi-Objective Optimization Method

Yi-nan Guo, Meirong Chen, Haobo Fu, and Yun Liu

***Abstract*—Robust optimization over time is a practical dynamic optimization method, which provides two detailed computable metrics to get the possible robust solutions for dynamic scalar optimization problems. However, the robust solutions fit for more time-varying moments or approximate the optimum more because only one metric is considered as the optimization objective. To find the true robust solution set satisfying maximum both survival time and average fitness simultaneously during all dynamic environments, a novel two-layer multi-objective optimization method is proposed. In the first layer, considering both metrics, the acceptable optimal solutions for each changing environment is found. Subsequently, they are composed of the practical robust solution set in the second layer. Taking the average fitness and the length of the robust solution set as two objectives, the optimal combinations for the whole time-varying environments are explored. The experimental results for the modified moving peaks benchmark shows that the robust solution sets considering both metrics are superior to the robust solutions gotten by ROOT. As the key parameters, the fitness threshold has the more obvious impact on the performances of MROOT than the time window, whereas ROOT is more sensitive to both of them.**

I. Introduction

In most of the practical optimization problems, many uncertain factors, such as the noise and disturbance, cause the time-varying objectives. Even the number of the objectives may be changed during the optimization. This kind of optimization problems is termed as dynamic optimization problems(DOPs)[1-4]. To solve the DOPs, many population-based optimization methods were proposed to detect the changed environments and improve the diversity. Tracking moving optimum (TMO)[5-7] is a conventional idea to detect where and when the environment changed. In case a change occurred and automatically checks out by the self-check operator[8], the new evolutionary algorithm is targeted to find the global optimum. However, the time-varying moment is difficult to be sensed in time and the global optimum may not to be found during a limited time, especially in the quickly changing environment, due to large computation complexity and time-consuming.

In order to get the more efficient global optimum during the optimization, the concept of robust optimization over time(ROOT) was proposed by Jin[9-10]. The purpose was to find a sequence of acceptable optimal or sub-optimal solutions, whose qualities meet the need of more than one changing environment in terms of certain problem-specific criterion. Though the framework of ROOT given by Jin is rational, it is not easy to computation because the complex integral operation. Consequently, Fu[11] proposed a more practical and computable method to find robust solutions over time. Because the fitness function only changes along certain time instants, DOP is converted into discrete optimization problem, noted as $F(\vec{X},\alpha(t))$. $\vec{X} \in \mathbb{R}^p$ stands for the design variables. $\alpha(t)$ is the time-dependent problem parameters changed at a series of time-varying moments $t \in (0, t_{\max}]$. $t_{\max}$ is the maximum time-varying moment. Furthermore, $F(\vec{X},\alpha(t))$ is simplified into $F_t(\vec{X})$ for short. To measure the robustness of all solutions, two metrics[10], including the survival time F^s and the average fitness F^a, are proposed.

The survival time is defined as maximum time interval starting from t, during which the fitness of $\vec{X_t}$ stays above the user-defined fitness threshold δ.

$$F^s(\vec{X},t,\delta) = \max\{0 \cup \{F_i(\vec{X}) \geq \delta, \forall i, t \leq i \leq t+l\}\} \quad (1)$$

The average fitness is defined as the average fitness value over the user-defined time window starting from time t. Suppose T is the time window:

$$F^a(\vec{X},t,T) = \frac{1}{T}\sum_{i=t}^{t+T-1} F_i(\vec{X}) \quad (2)$$

Taking each above metric as an objective, a sequence of robust solutions is achieved. Each solution can approximate the global optimum for at least one changing environment and may be the sub-optimal satisfying the requirement of the user-defined fitness threshold. Compared with TMO, population-based optimization methods are not done under each dynamic environment, which makes the cost of optimization less. ROOT, consequently, fits for DOPs with severely and quickly changed parameters.

Though Fu[11] proposed two detailed computable metrics, , only one metric is considered as the optimization objective to get possible robust solutions over time. That means the robust solutions found at each time-varying moment have larger fitness values or longer survive time. So it is actually a single-objective optimization method. How to find the true robust solutions set satisfying both maximum survival time

Yi-nan Guo is with the China University of Mining and Technology, Xuzhou, Jiangsu, CO 221116 CHINA (Corresponding Tel.: 86-13852000540; Fax: 86-516-83590817; e-mail: guoyinan@cumt.edu.cn).
Meirong Chen is with the China University of Mining and Technology, Xuzhou, Jiangsu, CO 221116 CHINA (e-mail: cmrzl@126.com).
Haobo Fu is with the University of Birmingham, Birmingham, CO B27 7EQ UK (e-mail: hxf990@cs.bham.ac.uk).
Yun Liu is with the China University of Mining and Technology, Xuzhou, Jiangsu, CO 221116 CHINA (e-mail: 635523709@qq.com).
This work was supported in part by Jiangsu Natural Science Foundation under Grant BK2010183 and Jiangsu Overseas Research & Training Program for University Prominent Young & Middle-aged Teachers and Presidents.

F^s and average fitness F^a during all dynamic environments is an open issue.

In this paper, multi-objective robust optimization over time (MROOT) is defined and a novel two-layer multi-objective optimization method is discussed. Its goal is to find a sequence of robust solutions considering both the survival time F^s and the average fitness F^a simultaneously. In the first layer, the acceptable non-domination solutions satisfied both metrics are found for each changing environment. Subsequently, taking the average fitness and the length of the robust solution set as the objectives, the sequences of robust solutions having the optimal performances during the whole time-varying moments are found in the second layer. Obviously, the dynamic scalar optimization problem $F_t(\vec{X})$ is transformed to a two-objective optimization problem.

This paper is organized as follows. Section 2 describes the detailed algorithm steps of two-layer multi-objective ROOT. The simulation results are compared with the ROOT[11] and analyzed in Section 3. At last, the strength of the proposed algorithm is summarized in Section 4.

II. Two-layer Multi-Objective Optimization Method to Find Robust Solutions over Time

The robust solutions over time, which have longest survival time or largest average fitness at each time-varying moment, can be found by Fu[11]. In order to obtain the sequence of robust solutions having both maximum survival time and average fitness simultaneously during all dynamic environments, a two-layer multi-objective optimization method based on NSGA-II is proposed. We call this multi-objective robust optimization over time.

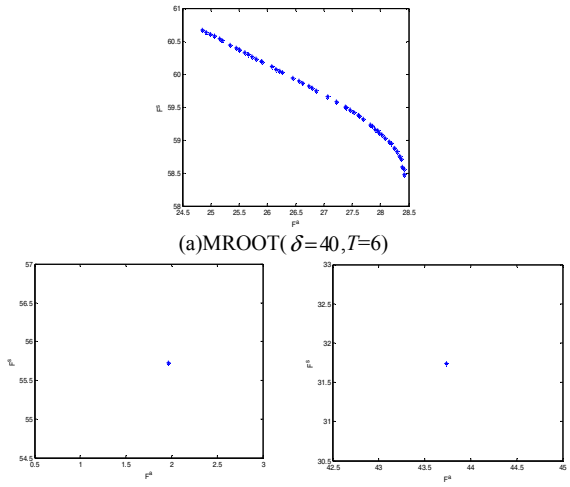
In the first layer, all acceptable non-domination solutions with the longer survival time and better average fitness are explored for each time-varying environment. They are actually composed of a Pareto front, which provides the alternative acceptable solutions for the second layer. Taking the survival time F^s and the average fitness F^a as two objectives, a multi-objective optimization problem is formed.

$$\begin{aligned} \max\ & F(\vec{X}_t) = \{F^s(\vec{X},t,\delta), F^a(\vec{X},t,T)\} \\ \text{s.t.}\ & \vec{X}_{\min} \le \vec{X} \le \vec{X}_{\max} \\ & 0 < t \le t_{\max} \end{aligned} \tag{3}$$

δ stands for the fitness threshold assigned by human in advance. It reflects the tolerance of robust non-domination solutions apart from the optimum. T is the time window, which restricts maximum time interval about how many consecutive dynamic environments $\vec{X}_t$ has the acceptable average fitness over δ. Considering above multi-objective optimization problem, the acceptable non-domination robust solutions at each time-varying moment, denoted by P_t, are gotten by NSGA-II[12]. NSGA-II proposed by Deb is a conventional multi-objective optimization algorithm and has been widely used. The non-dominated sort and the crowd operator are the effective approaches to ensure the approximation and distribution of the Pareto front. The pseudo code is listed as follows:

For t
BEGIN
$g = 0$;
Initialize $P_t(0)$;
while ($g < G$)
{
get $Q_t(g)$ *from* $P_t(g)$ *by crossover and mutation*;
calculate $F^s(\vec{X}_t^i(g))$, $F^a(\vec{X}_t^i(g))$;
$R_t(g) \leftarrow Q_t(g) \cup P_t(g)$;
$r(\vec{X}_t^i(g))$ =*non-dominated_sort*($R_t(g)$);
$d(\vec{X}_t^i(g))$ =*crowding_distance_assignment*($R_t(g)$);
$\{\vec{X}_t^i(g) \in P_t(g+1), r(\vec{X}_t^i(g)) = 0\}$;
$\{\vec{X}_t^i(g) \in P_t(g+1), \min d(\vec{X}_t^i(g))$ and $r(\vec{X}_t^i(g)) > 0\}$;
$g = g + 1$;
}
Output P_t^*
END

Compared with the optimal Pareto front on any time instant shown in Fig.1(a), only one robust solution is gotten at the same time by Fu[11] because only one metric is considered, as shown in Fig.1(b)-(c). They present the robust solution under $\delta = 40$ or $T = 6$ respectively. No matter the robust solutions found by ROOT or the robust Pareto front gotten by MROOT, both of them just reflect the robustness of solutions at any time-varying moment. They are not the true sequence of robust solutions, which directly apply to solve the practical DOPs.



(a)MROOT($\delta = 40, T = 6$)

(b) ROOT($\delta = 40$) (c) ROOT(T=6)

Fig.1. The robust solutions in any time-varying moment

In the second layer, the sequence of robust solutions for the whole dynamic environments are constituted based on P_t and the optimal robust solution set is found by multi-objective optimization method. The key issues in the optimization process are: (i) How to construct a sequence of robust solutions during the optimization? (ii) How to measure the performance of each robust solution set?

Suppose $\overrightarrow{X_t^i} \in P_t$ is the ith non-domination robust solution at t. Its survival time denoted by $l(\overrightarrow{X_t^i}) \geq 1$ means how many continuously changed environments $\overrightarrow{X_t^i}$ fits for. A sequence of robust solutions consists of more than one $\overrightarrow{X_t^i}$ in proper order. At first, $\overrightarrow{X_1^i} \in P_1$ is chosen randomly. $l(\overrightarrow{X_1^i}) \geq 1$ means the performances of $\overrightarrow{X_1^i}$ are acceptable under consecutive dynamic environments from t=1 to $t = l(\overrightarrow{X_1^i})$. Thus $\overrightarrow{X_1^i}$ is assigned as the robust solution when $t \in [1, l(\overrightarrow{X_1^i})]$. Subsequently, from $t = 1 + l(\overrightarrow{X_1^i})$, any robust solution is chosen from $P_{1+l(\overrightarrow{X_1^i})}$ so as to construct a robust solution set with $\overrightarrow{X_1^i}$. In case $l(\overrightarrow{X^j_{1+l(\overrightarrow{X_1^i})}}) \geq 1$, $\overrightarrow{X^j_{1+l(\overrightarrow{X_1^i})}}$ is assigned as the robust solution from $t = 1 + l(\overrightarrow{X_1^i})$ to $t = 1 + l(\overrightarrow{X_1^i}) + l(\overrightarrow{X^j_{1+l(\overrightarrow{X_1^i})}})$. This process is not ended until $t = t_{\max}$, as shown in Fig.2. The sequence of robust solutions covering all time-varying moments is called the robust solution set, denoted by $\vec{S} = \{\overrightarrow{X_1^i}, \overrightarrow{X^j_{1+l(\overrightarrow{X_1^i})}}, \cdots\}$, $1 \leq |\vec{S}| \leq t_{\max}$.

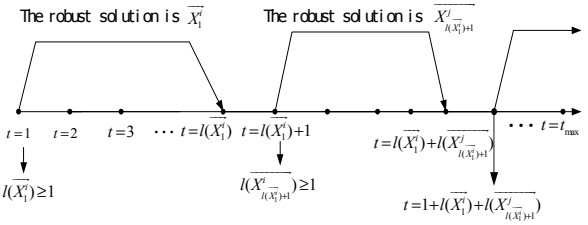


Fig. 2. Construct a robust solution set

The robust solution set containing less $\overrightarrow{X_t^i}$ or having larger average fitness is better. Hence, two metrics are defined to measure the performance of each $\vec{S}$. One is the length of $\vec{S}$, denoted by $F^V(\vec{S}) = |\vec{S}|$. Obviously, $1 \leq F^V(\vec{S}) \leq t_{\max}$. As $F^V(\vec{S})$=1, $\vec{S}$=$\{\overrightarrow{X_1^i}\}$ and $l(\overrightarrow{X_1^i}) = t_{\max}$. That means $\overrightarrow{X_1^i} \in P_1$ stays above the fitness threshold δ during all dynamic environments and the robustness of $\vec{S}$ is best. On the contrary, $F^V(\vec{S}) = t_{\max}$ means $\vec{S} = \{\overrightarrow{X_1^i}, \overrightarrow{X_2^j}, \cdots \overrightarrow{X_{t_{\max}}^k}\}$ and $l(\overrightarrow{X_t^i})$=1. Each $\overrightarrow{X_t^i}$ does not satisfy the fitness constrain under dynamic environment at t+1. The other metric is the average fitness of $\vec{S}$, denoted by $F^a(\vec{S})$.

$$F^a(\vec{S}) = \frac{1}{F^V(\vec{S})} \sum_{i=1}^{F^V(\vec{S})} f(\overrightarrow{S^i}) \tag{4}$$

$$f(\overrightarrow{S^i}) = f(\overrightarrow{X_t^k}) \tag{5}$$

Taking $F^V(\vec{S})$ and $F^a(\vec{S})$ as two objectives, to find the optimal robust solution set is in fact a multi-objective optimization problem.

$$\max \ F(\vec{S}) = \{F^V(\vec{S}), F^a(\vec{S})\} \tag{6}$$

NSGA-II is adopted to find the optimal robust solution sets. The pseudo code of the second layer of MROOT is shown as follows:

BEGIN
$g = 0$;
construct $\overrightarrow{S^i}(0)$ *from* P_t^*;
Initialize $PS(0)$;
while($g < G$)
{
get $QS(g)$ *from* $PS(g)$ *by crossover and mutation*;
calculate $F^V(\overrightarrow{S^i}(g))$, $F^a(\overrightarrow{S^i}(g))$;
$RS(g) \leftarrow QS(g) \cup PS(g)$;
$r(\overrightarrow{S^i}(g))$=*non-dominated_sort*($RS(g)$);
$d(\overrightarrow{S^i}(g))$=*crowding_distance_assignment*($RS(g)$);
$\{\overrightarrow{S^i}(g) \in PS(g+1), r(\overrightarrow{S^i}(g)) = 0\}$;
$\{\overrightarrow{S^i}(g) \in PS(g+1), \min d(\overrightarrow{S^i}(g))$ and $r(\overrightarrow{S^i}(g)) > 0\}$;
$g = g + 1$;
}
Output SP^*
END

III. Simulation Results and Their Analysis

Taking the modified moving peaks functions as the benchmark, the strength and weakness of MROOT and ROOT are compared by the following experiments.

A. The Benchmark Functions

The moving peaks function is the most commonly used benchmark for DOPs. In this paper, all experiments are done on the modified moving peaks benchmark (mMPB)[13].

$$F_t(\overrightarrow{X}) = \max_{i=1,\cdots m} \{H_t^i - W_t^i * \| \overrightarrow{X} - \overrightarrow{C_t^i} \|_2\} \tag{7}$$

where

$$H_t^i = H_t^i + height\_severity^i * N(0,1)$$
$$W_t^i = W_t^i + width\_severity^i * N(0,1)$$
$$\overrightarrow{C_{t+1}^i} = \overrightarrow{C_t^i} + \overrightarrow{v_{t+1}^i}$$
$$\overrightarrow{v_{t+1}^i} = \frac{s * ((1-\lambda) * \vec{r} + \lambda * \overrightarrow{v_t^i}}{\| (1-\lambda) * \vec{r} + \lambda * \overrightarrow{v_t^i} \|}$$

H_t^i, W_t^i and $\overrightarrow{C_t^i}$ respectively stand for the height, width and center of the ith peak at t. $\overrightarrow{X}$ is the variable. $N(0,1)$ represents a random number drawn from Gaussian distribution with zero mean and one variance. The parameters of mMPB and the algorithms are summarized in Table I.

TABLE I
THE PARAMETERS OF MMPB

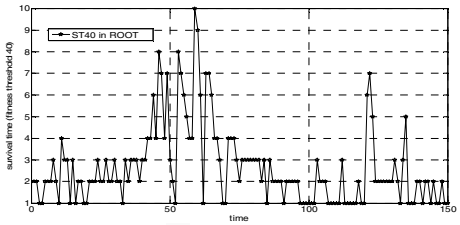
Peaks m	Change frequency	Dimension D	Trend parameter	Scale parameter
5	2500	2	1	1
Initial height	Initial width	Height range	Width range	Search range
50	6	[30,70]	[1,12]	[0,50]
Height_severity range	Width_severity range	Population size	θ	Generation n
[1,10]	[0.1,1]	50	1	1000

B. Comparison and analysis of the experimental results

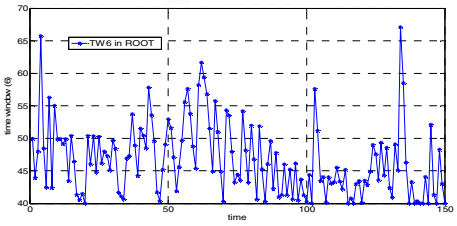
Nine groups of experiments are designed to further explain the concept of MROOT and compare the difference and the strength between ROOT and MROOT.

(1)*Comparison on the robust solutions and their performances gotten by ROOT and MROOT*

All of the experiments are done for the mMPB and the main parameters are same: δ=40, T=6. The robust solutions gotten by ROOT are the optimal one satisfying maximum survival time or maximum average fitness, shown as Fig.3.



(a) $F^s(\delta = 40)$



(b) $F^a(T = 6)$

Fig. 3. The robust solutions by ROOT

Taking $t = 67$ as example, we see from Fig.4 that each robust solution gotten by ROOT is a point, which has best performance in one side at each time-varying moment. To be different with ROOT, the acceptable robust solutions found in the first layer of MROOT contain more than one solution. They compose of a robust Pareto front, as shown in Fig.4. Furthermore, the optimal robust solution sets gotten by NSGA-II in the second layer of MROOT are shown in Fig.5.

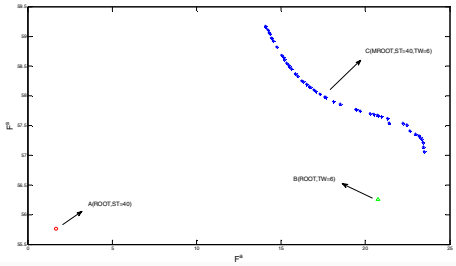


Fig.4.The robust solutions gotten by ROOT and the first layer of MROOT(t=67)

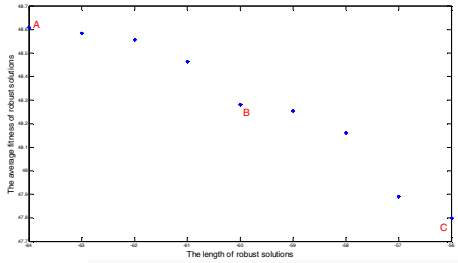
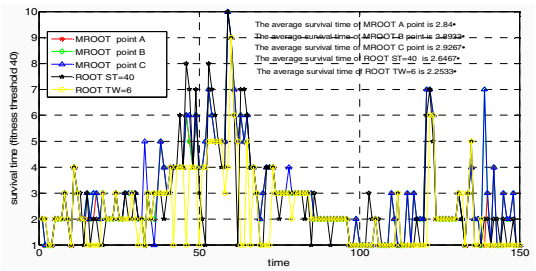
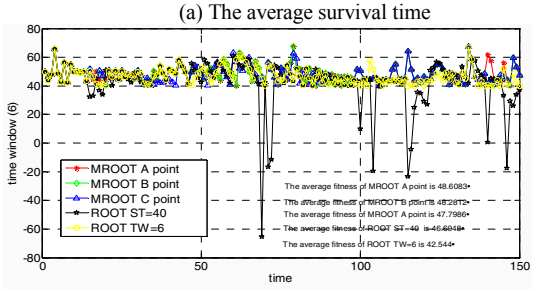


Fig.5. The optimal non-domination robust solution set found by the second layer of MROOT

Three typical robust solution sets are chosen from the optimal robust Pareto front shown in Fig.5 to compare the average survival time and average fitness with the robust solutions gotten by ROOT. Point A and C are the terminal point of the optimal robust Pareto front. B is its midpoint. From Fig.6, we see that average survival time and fitness of any robust solution set are superior to the robust solutions gotten by ROOT.



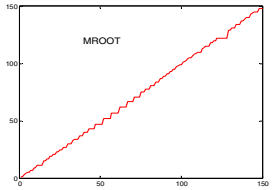
(a) The average survival time



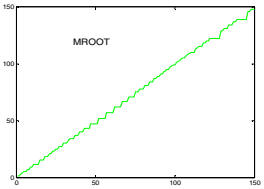
(b) The average fitness

Fig.6.Comparison of the robust solution's performance between ROOT and MROOT

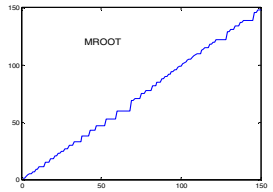
The number of dynamic environments that each robust solution fits for are shown in Fig.7. Compared with the robust solutions found by ROOT shown in Fig.7(d)-(e), $\overrightarrow{S_A}$, $\overrightarrow{S_B}$, $\overrightarrow{S_C}$ contain less robust solutions. That means their robustness are better.



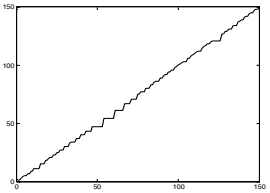
(a)MROOT($F^V(\overrightarrow{S_A})=64$)



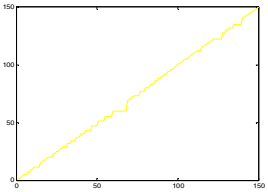
(b)MROOT($F^V(\overrightarrow{S_B})=60$)



(c)MROOT($F^V(\overrightarrow{S_C})=56$)



(d)ROOT($F^V(\overrightarrow{X})=68$)($\delta = 40$)



(e)ROOT($F^V(\overrightarrow{X})=77$)($T = 40$)

Fig.7. The robust solutions for 150 dynamic moments

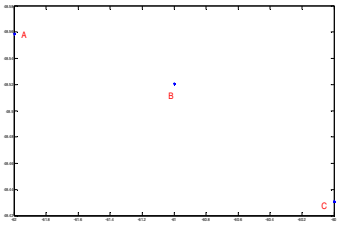
By compared the performances between the optimal robust Pareto front found by MROOT and the robust solutions gotten by ROOT listed in Table Ⅱ, it has shown that the former averagely fits for more time-varying moments with better average fitness. In other words, less robust solutions are contained in the non-domination robust solution set.

TABLE II
COMPARISON BETWEEN THE PERFORMANCES OF THE ROBUST SOLUTIONS BY ROOT AND MROOT($\delta = 40, T = 6$)

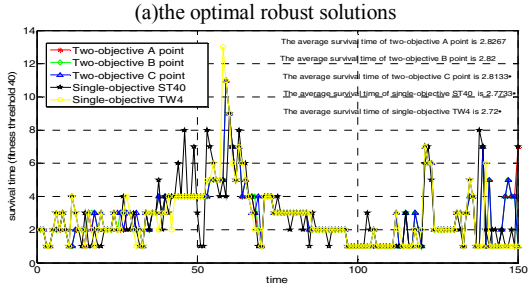
Algorithm	Average survival time	Average fitness	The length of robust solutions (set)
ROOT(ST=40)	2.6467	46.6948	68
ROOT(TW=6)	2.6533	42.5440	77
MROOT(A)	2.8400	48.6083	64
MROOT(B)	2.8933	48.2812	60
MROOT(C)	2.9267	47.7986	56

(2)*Analysis of the performance under different parameters*

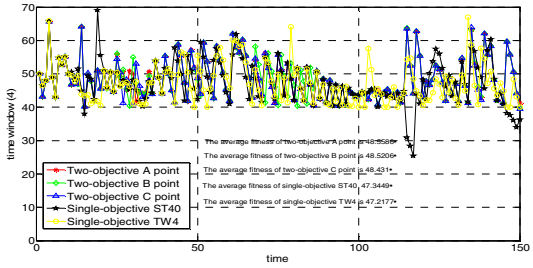
The fitness threshold and time window directly influence the evaluation criterion for the robust solutions. We further compare and analyze the performances of the optimal robust solutions under different δ or T. The simulation results under $\delta = 40, 45, 50$ and $T = 2, 4, 6$ are listed in Table Ⅲ. Corresponding average survival time and average fitness are shown in Fig.8-Fig.15.



(a)the optimal robust solutions



(b)average survival time



(c)average fitness

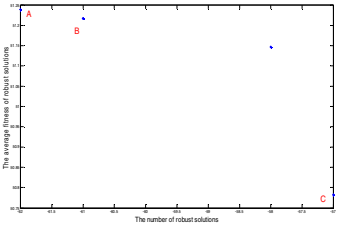
Fig.8.Comparison of the performances between the optimal robust solutions by ROOT and MROOT under $\delta = 40, T = 4$

On the following parts, we compare above experimental results deeply and further analyze them from four sides.

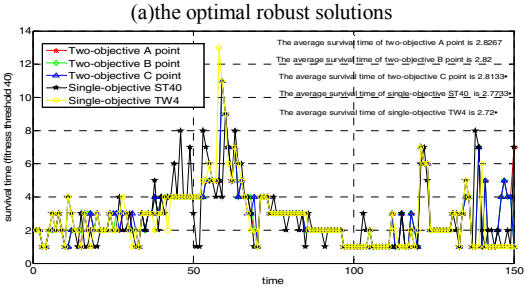
(i) By comparing the performances of three robust solution sets(A,B,C) located at the robust Pareto front gotten by MROOT under the same parameters, we see that the robust solution set A has shorter average survival time and better average fitness than C. That means there are more robust solutions composed of the robust solution set A. By analyzing, we known that during the optimization process, if we pay more attention to the average fitness, the robust solution set A will be gotten. On the contrary, the robust solution set C means the survival time is more important than average fitness. By compared with A and C, the robust solution set B has the compromise performance between the survival time and average fitness. In practical application, any solution set located in the robust Pareto front can be chosen in terms of the preference of the decision maker.

TABLE III
COMPARISON OF THE PERFORMANCES BETWEEN THE OPTIMAL ROBUST SOLUTIONS BY ROOT AND MROOT UNDER DIFFERENT δ AND T

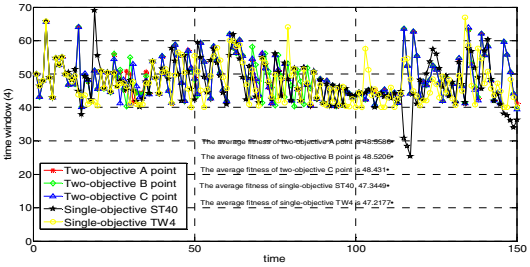
δ	T	Algorithm	Average survival time	Average fitness	The length of robust solution(set)
40	2	ROOT(ST=40)	2.7538	46.3659	63
		ROOT(TW=2)	3.0600	49.7000	56
		MROOT(A)	2.9800	51.2391	62
		MROOT(B)	2.9867	51.2167	61
		MROOT(C)	3.000	50.7822	57
	4	ROOT(ST=40)	2.7733	47.3449	64
		ROOT(TW=4)	2.7200	47.2177	67
		MROOT(A)	2.8267	48.5586	62
		MROOT(B)	2.8200	48.5206	61
		MROOT(C)	2.8133	48.4310	60
	6	ROOT(ST=40)	2.6467	46.6948	68
		ROOT(TW=6)	2.6533	42.5440	77
		MROOT(A)	2.8400	48.6083	64
		MROOT(B)	2.8933	48.2812	60
		MROOT(C)	2.9267	47.7986	56
45	2	ROOT(ST=45)	2.2667	50.4401	78
		ROOT(TW=2)	2.2667	52.8932	79
		MROOT(A)	2.4067	51.9935	76
		MROOT(B)	2.4067	51.9461	75
		MROOT(C)	2.4267	51.9256	74
	4	ROOT(ST=45)	2.3400	49.8702	78
		ROOT(TW=4)	2.0467	48.5826	95
		MROOT(A)	2.3000	52.0505	75
		MROOT(B)	2.3100	52.0040	74
		MROOT(C)	2.3300	51.8036	73
	6	ROOT(ST=45)	2.2800	50.3388	79
		ROOT(TW=6)	1.7533	42.5500	106
		MROOT(A)	2.3067	51.7803	74
		MROOT(B)	2.3067	51.7543	73
		MROOT(C)	2.3333	51.6035	72
50	2	ROOT(ST=50)	1.7933	53.3959	98
		ROOT(TW=2)	1.7533	55.2953	100
		MROOT(A)	1.8133	55.7652	96
		MROOT(B)	1.8200	55.7314	95
		MROOT(C)	1.8267	55.6954	94
	4	ROOT(ST=50)	1.7667	53.5350	99
		ROOT(TW=4)	1.4067	49.5130	119
		MROOT(A)	1.8333	55.5228	95
		MROOT(B)	1.8467	55.5056	94
		MROOT(C)	1.8533	55.3511	93
	6	ROOT(ST=50)	1.7533	53.6832	97
		ROOT(TW=6)	1.3600	43.5366	123
		MROOT(A)	1.8000	56.1983	99
		MROOT(B)	1.8067	56.1746	98
		MROOT(C)	1.8133	56.1416	97



(a)the optimal robust solutions

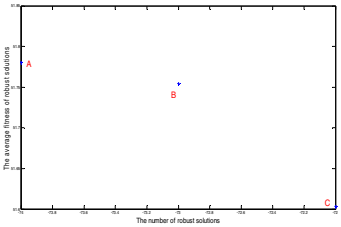


(b)average survival time

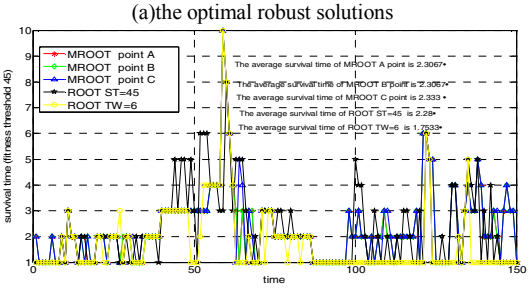


(c)average fitness

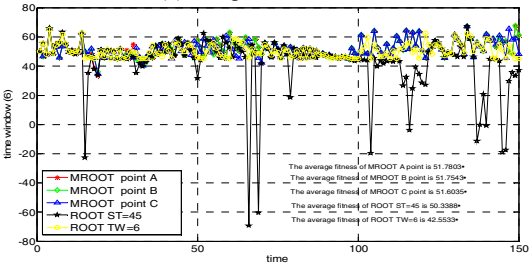
Fig.9.Comparison of the performances between the optimal robust solutions by ROOT and MROOT under $\delta = 40, T = 2$



(a)the optimal robust solutions

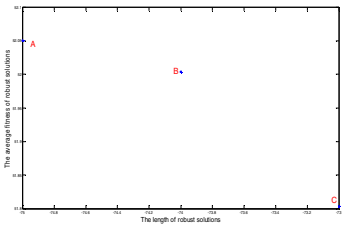


(b)average survival time

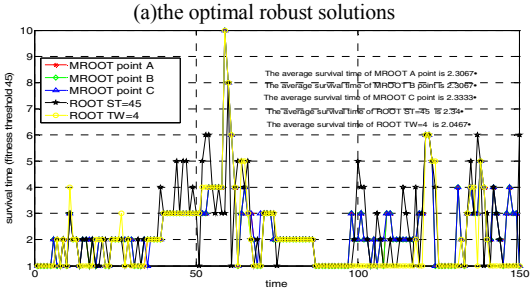


(c)average fitness

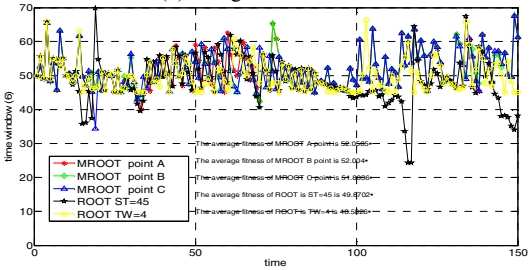
Fig.10.Comparison of the performances between the optimal robust solutions by ROOT and MROOT under $\delta = 45, T = 6$



(a)the optimal robust solutions

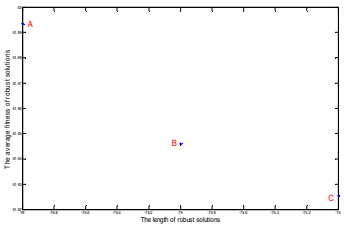


(b)average survival time

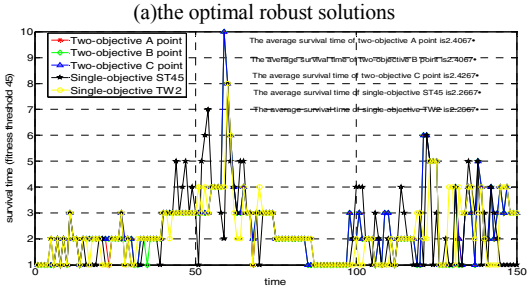


(c)average fitness

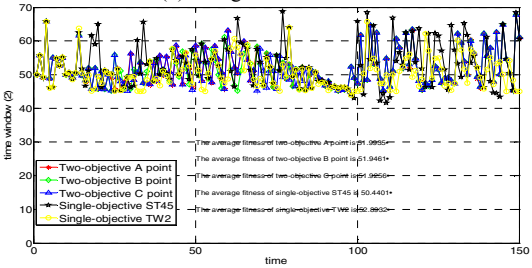
Fig.11.Comparison of the performances between the optimal robust solutions by ROOT and MROOT under $\delta = 45, T = 4$



(a)the optimal robust solutions

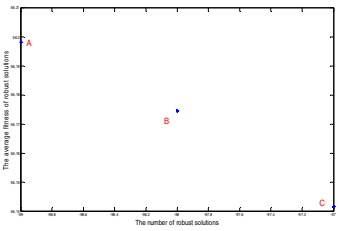


(b)average survival time

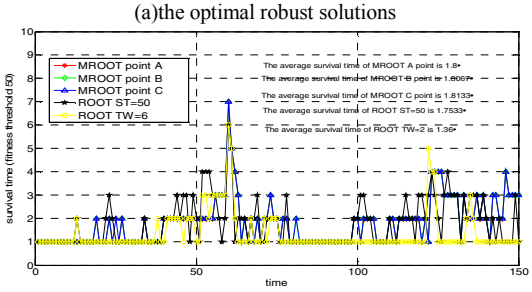


(c)average fitness

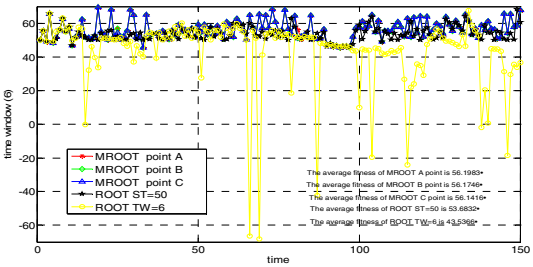
Fig.12.Comparison of the performances between the optimal robust solutions by ROOT and MROOT under $\delta = 45, T = 2$



(a)the optimal robust solutions

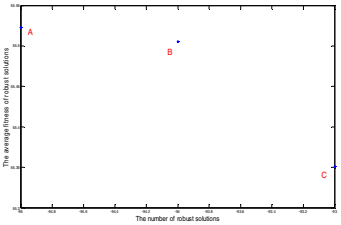


(b)average survival time

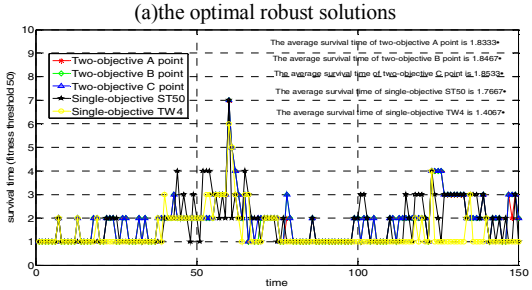


(c)average fitness

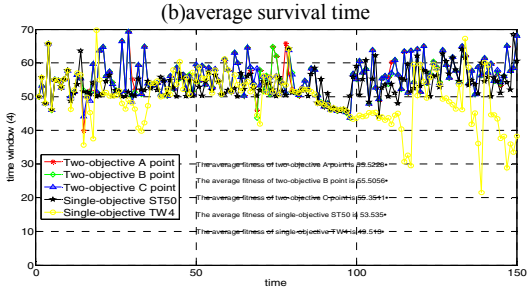
Fig.13.Comparison of the performances between the optimal robust solutions by ROOT and MROOT under $\delta = 50, T = 6$



(a)the optimal robust solutions

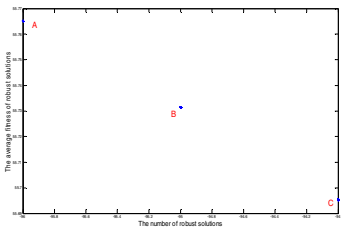


(b)average survival time

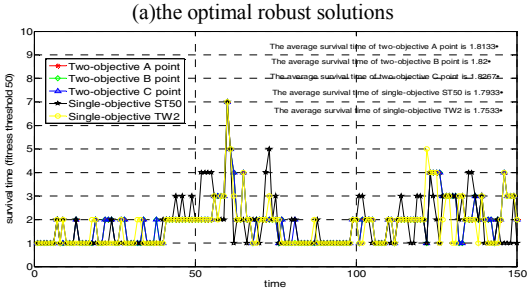


(c)average fitness

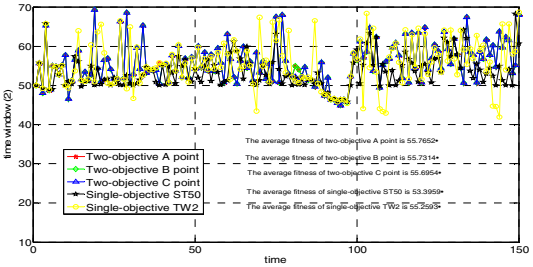
Fig.14.Comparison of the performances between the optimal robust solutions by ROOT and MROOT under $\delta = 50, T = 4$



(a)the optimal robust solutions



(b)average survival time



(c)average fitness

Fig.15.Comparison of the performances between the optimal robust solutions by ROOT and MROOT under $\delta = 50, T = 2$

(ii) The robust solutions gotten by ROOT and MROOT under the same parameters show that no matter which aspect is considered, the survival time or average fitness, the robust solution sets gotten by MROOT have superior performances than them by ROOT generally. The reason for that is through the two-layer optimization process in MROOT, the optimal combinations of the robust solutions are further explored based on the non-domination solutions at each time-varying moment. Especially, the combination optimization strategy in the second layer of MROOT provides the possibility to find the realizable robust solution set used in practice.

(iii) The experimental results under different T indicate that the time window has less impact on the performance of the robust solution set by MROOT. Because MROOT is multi-objective optimization actually, it is difficult to get a robust solution set with longer survival time than the time window by considering the constraint of average fitness at the same time. However, the time window plays a more obvious role in the optimization process of ROOT. It directly restricts the quality and quantity of the possible robust solutions at each time-varying moment. Less T makes the average survival time shorter and average fitness worse.

(iv) By compared the robust solution sets gotten by MROOT under different δ, it is obvious that δ has the large influence on MROOT. The robust solution sets' average survival time are decreasing and their average fitness becoming larger with the increasing of δ. Meanwhile, the robust solution sets contain more elements. That means the robustness of the robust solution sets deteriorate so as to

satisfy the higher requirements of average fitness. In practice, δ reflects the tolerance of the robust solutions apart from the true optimum at each time-varying moment. So it needs to be cautiously assigned in terms of the detail optimization problems.

IV. CONCLUSIONS

Though Fu[11] provides the detailed computable metrics for ROOT to get the possible robust solutions over time of DOPs, only one metric is considered as the optimization objective. Moreover, the obtained robust solutions only reflect the single performance at each time-varying moment. They are not the sequence of robust solutions during the optimization in nature. In order to find the true robust solution set satisfying maximum both survival time and average fitness simultaneously during all dynamic environments, a scalar dynamic optimization problem is converted to a two-layer two-objective optimization problems. In the first layer, the acceptable robust Pareto front considering both metrics is found at each time-varying moment. In the second layer, taking the average fitness and the length of the robust solution set as two objectives, the sequences of robust solutions chosen from the first layer during all time-varying moments are explored. NSGA-II is adopted as the multi-objective optimization method to solve two-objective optimization problems in both layers. The experimental results for the modified moving peaks benchmark show that the robust solution sets found by MROOT have larger average fitness and longer average survive time than the robust solutions gotten by ROOT. As the key parameters, the fitness threshold has the obvious impact on MROOT whereas the time window influences the performances of the robust solution sets less. However, ROOT is more sensitive to both of them.

In the future work, suitable test functions need to be constructed. Moreover, since the dynamic system is performed online, the solutions' fitness in the past or future dynamic moment shall be considered in MROOT. So the estimate and prediction task are inevitable.

REFERENCES

[1] Trung Thanh Nguyen and Xin Yao, "Dynamic Time-Linkage Evolutionary Optimization: Definitions and Potential Solutions," *Metaheuristics for Dynamic Optimization*, pp.371-395.2013

[2] Rotar, C.,Ileana, I.,Kadar, M.,and Muntean, M., "Evolutionary optimization in dynamic environment," in 2009 IEEE 5th International Con*ference on Intelligent Computer Communication Processing*, pp. 11-18

[3] A.Rauf Baig,and M. Rashid,"Honey bee foraging algorithm for multimodal and dynamic optimization problems,"in *2007 ACM GECCO*,pp.1326-1334

[4] T.M.Blackwell and J.Brank,"Multi-Swarm Optimization in Dynamic Environments", *Applications of Evolutionary Computation*, vol.3005,pp. 489-500. 2004.

[5] R.Eberhart, and Y. Shi, "Tracking and Optimizing Dynamic Systems with Particle Swarms,"in *2001 Proc of Computational Evolution*.pp.94-100

[6] Daniel Parrot, and Xiaodong Li,"Locating and Tracking Multiple Dynamic Optima by a Particle Swarm Model Using Speciation," *IEEE Transactions on Evolutionary Computation*, vol.10.pp.440-458,August 2006.

[7] Li, C., and Yang, S. "A general framework of multi-population methods with clustering in undetectable dynamic environments," *IEEE Transactions on Evolutionary Computation*.vol.16,pp.556-577,2012

[8] Chun-an Liu, and Yuping Wang, "Dynamic Multi-objective Optimization Evolutionary Algorithm,"in *2007 IEEE Third International Conference on Natural Computation*, pp.456-459

[9] Yu, X., Jin, Y., Tang, K., and Yao, X. "Robust optimization over time-A new perspective on dynamic optimization problems,"in *2010 IEEE Congress on Evolutionary Computation*. pp.1-6

[10] Jin, Y., Tang, K., Yu, X., Sendhoff, B., and Yao, X. "A framework for finding robust optimal solutions over time," *Memetic Computing*, vol. 5, pp. 3-18, 2013.

[11] Haobo Fu, Bernhard Sendhoff, Ke Tang, and Xin Yao. "Finding Robust Solutions to Dynamic Optimization Problems," *LNCS*,vol.7835: pp.616-625,2013

[12] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. "A fast and elitist multi-objective genetic algorithm:NSGA-II," *IEEE Transactions on Evolutionary Computation*. vol. 6, pp. 182-197, 2002.

[13] Haobo Fu, Bernhard Sendhoff, Ke Tang, and Xin Yao. "Finding Robust Solutions to Dynamic Optimization problems," *EvoApplications*. pp. 616-625,2013