Water Resour Manage (2010) 24:2611–2632
DOI 10.1007/s11269-009-9570-7

Multi-objective Cultured Differential Evolution for Generating Optimal Trade-offs in Reservoir Flood Control Operation

Hui Qin · Jianzhong Zhou · Youlin Lu · Yinghai Li · Yongchuan Zhang

Received: 25 March 2009 / Accepted: 27 December 2009 /
Published online: 12 January 2010

Abstract Reservoir flood control operation (RFCO) is a complex problem because it needs to consider multiple objectives and a large number of constraints. Traditional methods usually convert multiple objectives into a single objective to solve, using weighted methods or constrained methods. In this paper, a new approach named multi-objective cultured differential evolution (MOCDE) is proposed to deal with RFCO. MOCDE takes cultural algorithm as its framework and adopts differential evolution (DE) in its population space. Considering the features of DE and multi-objective optimization, three knowledge structures are defined in belief space to improve the searching efficiency of MOCDE. MOCDE is first tested on several benchmark problems and compared with some well known multi-objective optimization algorithms. On achieving satisfactory performance for test problems, MOCDE is applied to a case study of RFCO. It is found that MOCDE provides decision makers many alternative non-dominated schemes with uniform coverage and convergence to true Pareto optimal solutions in a short time. The results obtained show that MOCDE can be a viable alternative for generating optimal trade-offs in reservoir multi-objective flood control operation.

Keywords Reservoir flood control operation · Multi-objective optimization · Cultural algorithm · Differential evolution · Knowledge structure · Adaptive Cauchy mutation

H. Qin · J. Zhou (✉) · Y. Lu · Y. Zhang
College of Hydropower and Information Engineering, Huazhong University of Science and Technology, Wuhan, 430074, People's Republic of China
e-mail: jz.zhou@hust.edu.cn

Y. Li
College of Civil & Hydroelectric Engineering, China Three Gorges University, Yichang, 443002, People's Republic of China

1 Introduction

Flood disaster is one of the most damaging natural disasters, due to its high frequency and enormous destruction strength. Especially in China, the threat of flood disaster is much greater because large numbers of people live around rivers. To defend flood disaster, many hydro projects have been built in China, such as Three Gorges Project (TGP) in the Yangtze River. Yangtze River is the longest river in China and third longest in the world. It nurtures 400 million people of China, however, its floods are often a great threat to the life and property safety of persons living along the downstream. Especially in the middle-lower Yangtze area, the loss is tremendous once a flood disaster occurs, due to the flat terrain, developed economy and high population density. TGP is an extremely important project for Yangtze River flood control and it highly improves the flood control capacity of Yangtze River Basin, owing to its huge flood storage capacity. The goal of reservoir flood control operation (RFCO) is to minify the flood peaks utilizing flood storage capacity of reservoirs. It is generally complex as it involves multiple objectives (such as safety of the dam, upstream and downstream flood control requirement, hydropower generation, navigation, and so on) and a large number of constraints. Because of the multiple conflictive objectives, it is not possible to find a single optimal solution, which will satisfy all the objectives. Instead, the solution exists in the form of alternative trade-offs, also known as the Pareto optimal solutions.

In the past, researchers have used many optimization techniques to solve this problem. such as linear programming (Windsor 1973), dynamic programming (DP) (Schultz and Plate 1976; Mei 1999), nonlinear programming (Unver and Mays 1990), and folded dynamic programming (FDP) (Nagesh Kumar and Baliarsingh 2003; Nagesh Kumar et al. 2009). More recently, intelligent methods inspired by biological intelligence, such as genetic algorithm (GA) (Li-Chiu 2008; Karamouz et al. 2009) and artificial neural network (Wei and Hsu 2008; Mehta and Jain 2009) were presented to solve reservoir flood control and management problem. However, most of these methods adopted a weighted approach or a constrained approach to convert multiple objectives into single objective, without considering all the objectives simultaneously. Because these methods use point-by-point searching approaches, so the outcome is a single solution from each run. It needs to iterate several times to get a set of solutions. Furthermore, these methods usually fail in yielding true Pareto optimal solutions, when the objective functions are nonconvex and consist of disconnected Pareto fronts (Deb 2001). Cheng and Wang (1995) presented an interactive method for generating flood control operation alternatives, depending on the operation rules and operators' intuition and experience. This method can generate alternatives fast and consider the operators' experience, so it is usable in practical flood control operation. However, it just generates feasible alternatives and can't ensure that these alternatives are non-dominated, and the strong subjectivity of this method may weaken its validity and practicability.

Compared to classical approaches, multi-objective evolutionary algorithms (MOEAs) have two advantages in dealing with multi-objective optimization problems: (1) MOEAs can get several non-dominated solutions in a single run, due to their population-based characteristic; (2) MOEAs are less sensitive to the shape or continuity of the Pareto surface. Since Schaffer (1985) first used evolutionary algorithms to solve multi-objective optimization problems, many MOEAs have been proposed in recent decades, NSGA-II (Deb et al. 2002) and SPEA2 (Zitzler et al. 2001) are the most popular ones. In recent years, researchers have begun to apply MOEAs to the field of multipurpose reservoir operation problems and achieved various degrees of success. Kim et al. (2006) applied NSGA-II to four interconnected reservoirs operation in the Han River Basin. The objectives of reservoirs operation include maximizing reservoirs' releases and storage levels, subject to continuity constraints and end-of-period storage constraints. Janga Reddy and Nagesh Kumar (2006, 2007a, b) proposed three approaches, Multi-objective Genetic Algorithm, Multi-objective particle swarm optimization and Multi-objective Differential Evolution to solve a reservoir operation problem with multiple purposes of irrigation, hydropower generation and river water quality. Chen et al. (2007) presented a Macro-evolutionary Multi-objective Genetic Algorithm (MMGA) for optimizing the multipurpose reservoir rule curves with two objectives involving water supply and hydropower generation. Alexandre and Darrell (2008) presented an implementation of Multi-objective Particle Swarm Optimization (MOPSO) on multipurpose reservoir operation problem and indicated that this method, however, requires some fine tuning of its control parameters. In this paper, we attempt to present a new MOEA to solve RFCO problem.

Differential evolution (DE) is a new type of evolutionary algorithm (Storn and Price 1995). It is simple yet powerful, and has been successfully used in solving single objective optimization problems (Yuan et al. 2008; Mandal and Chakraborty 2009). In recent years, some researchers have extended it to deal with multi-objective optimization problems, such as Pareto differential evolution (PDE) algorithm (Abbass et al. 2001), Pareto differential evolution approach (PDEA) (Madavan 2002), Pareto-based multi-objective differential evolution (abbr. PMODE) (Xue et al. 2003), differential evolution for multi-objective optimization (DEMO) (Rolic and Filipic 2005) and adaptive differential evolution algorithm (ADEA) (Qian and Li 2008). However, most of these DE based multi-objective optimization algorithms suffer from premature convergence at different degrees (Madavan 2002; Xue et al. 2003; Rolic and Filipic 2005). In this paper, we present a new multi-objective optimization algorithm by incorporating cultural algorithm (Reynolds 1994) and DE, and name it multi-objective cultured differential evolution (MOCDE). MOCDE uses cultural algorithm as its framework and DE in its population space. To improve the algorithm's searching efficiency, three knowledge structures are pertinently designed in the belief space: the situational knowledge structure and normative knowledge structure are used to improve convergence rate and diversity of solutions; history knowledge monitors the convergence process and adopts an adaptive Cauchy mutation to avoid premature convergence. MOCDE is first tested by several benchmark test problems and compared with some well known multi-objective optimization algorithms, and the results show that MOCDE is efficient and robust in dealing with multi-objective optimization problems. After that, we apply MOCDE to a practical problem—multi-objective flood control operation of Three Georges Reservoir. It is found that MOCDE can provide decision-makers many alternative Pareto optimal solutions with uniform coverage and convergence to true Pareto optimal solutions in a short time.

The rest of the paper is organized as follows. Flood control operation problem is formulated in Section 2. Thereafter, in Section 3, we describe MOCDE in details.

Section 4 presents the application study of MOCDE to a practical RFCO problem. The paper concludes with Section 5.

2 Model Formulation of RFCO

RFCO involves multiple objectives, the main of which are generally classified into three types (Mei 1999): (1) the safety requirement for the dam; (2) flood control requirement for reservoir area (upstream); (3) flood control requirement for downstream protected area. The first two goals are related to the flood water volume stored in the reservoir, which are represented by the maximum upstream water level of the dam and the duration of high water level. Flood control requirement for downstream protected areas is mainly related to the maximum discharge volume of the dam and the duration of large discharge volume. The first two objectives expect the reservoir doesn't store a large flood water volume to ensure the safety of the dam and decrease loss of upstream area. Contrarily, the third objective expects the reservoir store as much flood water volume as possible to protect the downstream areas. In this paper, we establish the model of RFCO by selecting the maximum upstream water level as the optimization objectives for the first two goals and the maximum discharge volume of the dam as the optimization objective for downstream protected areas. The objectives are expressed as follows.

$$\min F_1 = \min Z_{\max} = \min\{\max(Z_t)\} \quad t = 1, 2, ..., T \tag{1}$$

$$\min F_2 = \min Q_{\max} = \min\{\max(Q_t)\} \quad t = 1, 2, ..., T \tag{2}$$

Where Z_t is the upstream water level of the t-th period, T is the number of periods, Q_t is the discharge volume of the t-th period, $Z_{\max}$ and $Q_{\max}$ are the maximal water level and discharge volume of all periods, respectively.

Constraints of this model are described as follows:

(1) Upstream water level limit.

$$Z_t^{\min} \le Z_t \le Z_t^{\max}, \quad t \in [1, T] \tag{3}$$

Where $Z_t^{\min}$ and $Z_t^{\max}$ are the minimum and maximum limit of upstream water level of the t-th period. This limit is the intersection of the dam's own physical restriction and level limit prescribed in reservoir operating regulations.

(2) Water release ability limit of the dam.

$$Q_t \le Q_{\max}\left(Z_t^{avg}\right) \tag{4}$$

Where Q_t is the discharge volume of the t-th period, Z_t^{avg} (equal to $(Z_t + Z_{t-1})/2$) is the average upstream water level of the t-th period, $Q_{\max}\left(Z_t^{avg}\right)$ is the maximum water release ability of the dam at level Z_t^{avg}.

(3) Discharge volume limit.

$$Q_t^{\min} \le Q_t \le Q_t^{\max}, \ t \in [1, T] \tag{5}$$

Where $[Q_t^{\min}, Q_t^{\max}]$ is the intersection of discharge volume limit prescribed in reservoir operating regulations and the physical restriction of the dam.

(4) Water balance equation.

$$V_t = V_{t-1} + I_t - Q_t \tag{6}$$

Where V_t and V_{t-1} are the reservoir storages, I_t and Q_t are the reservoir inflow and discharge volume, respectively.

(5) Final upstream water level limit.

$$Z_T \rightarrow Z_{FL} \tag{7}$$

Where Z_T is the final upstream water level, Z_{FL} is the flood control limit level. This constraint expects the final upstream water level to fall back to flood control limit level, to cope with following possible floods. It is a soft constraint that needn't to be satisfied accurately.

3 Multi-objective Cultured Differential Evolution

DE is easy to implement and its convergence rate is fast, but it is prone to converge prematurely when solving problems that have many local optima. The MOEAs based on DE, mentioned in Section 1, don't take direct measures to avoid this problem and suffer from premature convergence at different degrees, which can also be seen from the convergence metric values in Table 5 presented in Section 3.5. The main goal of this section is to present MOCDE which can avoid this problem effectively. Firstly, we give a brief introduction of cultural algorithm and DE, then we present a detailed description of the knowledge structures defined in belief space, at last we outline MOCDE and test it using some benchmark test problems.

3.1 Background

3.1.1 Framework of Cultural Algorithm

The Cultural Algorithm is a computational model of social evolution based upon a general model of the cultural evolution process. The key idea behind Cultural Algorithm is to acquire problem-solving knowledge (beliefs) from the evolving population and in return make use of that knowledge to guide the searching process (Reynolds 1994; Jin and Reynolds 1999). The basic framework of cultural algorithm is shown in Fig. 1.

Cultural algorithm consists of two spaces: population space and belief space. Population space consists of a set of individuals, and it can be modeled using any population-based technique, such as genetic algorithms (Reynolds 1994), evolutionary programming (Jin and Reynolds 1999), particle swarm optimization (Iacoban et al. 2003), differential evolution (Becerra and Coello 2006), and so on; the knowledge acquired by the individuals along the evolutionary process is stored in belief space. Evolution processes of these two spaces are relatively independent. To unify both spaces, a communication protocol is established. For example, to update the belief space, the individual experiences (a select set of excellent individuals

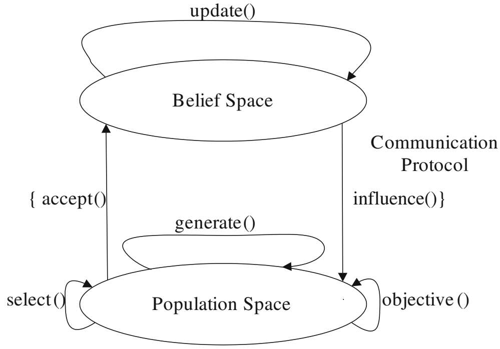


Fig. 1 Framework of cultural algorithm

obtained during evolution process) in population space are added to belief space with the function *accept*(), then belief space uses function *update*() to update its knowledge, according to some special rules. On the other hand, the operators in the population space (e.g. recombination, mutation and selection) are modified by the function *influence*() to improve the algorithm's searching efficiency. Function *objective*() is used to evaluate the fitness of every individual, function *generate*() generates offspring individuals and function *select*() selects the individuals for next generation.

From the description above we can see that cultural algorithm provides a general framework for population-based evolutionary algorithms. Information interaction between population space and belief space increases the computational complexity to some extent, but the searching purposefulness of the evolution process can be enhanced under the guidance of knowledge in belief space, thus the whole searching efficiency of cultural algorithm exceeds the searching efficiency of evolutionary algorithms just based upon biological evolution.

3.1.2 Overview of Differential Evolution

Differential Evolution is an approach developed for single-objective optimization in continuous search spaces (Storn and Price 1995). It is conceptually simple and easy to implement. Details of the algorithm can be found elsewhere (Storn and Price 1995), only main features are summarized here. DE includes three operators: mutation, crossover and selection. It uses a population $\boldsymbol{P}^g$ that contains N_P n-dimensional real-valued parameter vectors (named $\boldsymbol{x}_i^g$, $i = 1, 2...N_P$) in generation g. Different from genetic algorithm, mutation is the key operator of DE. DE mutation generates new trial parameter vectors (named $\boldsymbol{v}_i^{g+1}$) by adding a weighted difference between two (or more) parameter vectors selected randomly from current population to another parameter vectors selected from the same population. $\boldsymbol{v}_i^{g+1}$ is generated according to the following mutation scheme:

$$\boldsymbol{v}_i^{g+1} = \boldsymbol{x}_{r_1}^g + F\left(\boldsymbol{x}_{r_2}^g - \boldsymbol{x}_{r_3}^g\right) \quad i = 1, 2, ..., N_P \tag{8}$$

Where integers r_1, r_2 and r_3 are chosen randomly in the range [1, N_P], and are different from each other. The mutation parameter F ([0, 2]) is a real, constant, user-supplied parameter that controls the amplification of the differential vector. Considering that the mutation operation may lead to new vectors that fall outside the boundaries of the variables, a simple strategy is used here to deal with this case: if a variable of new vectors gets outside its boundary, just let it equal the boundary. Some variants of DE have been developed, and more details can be found elsewhere (Storn 1996).

DE uses discrete recombination to modify the trial parameter vector and get a candidate (named $\boldsymbol{u}_i^{g+1}$), as follows:

$$\boldsymbol{u}_{i,j}^{g+1} = \begin{cases} \boldsymbol{v}_{i,j}^{g+1}, & if\ (\text{random}() \leq CR)\ \ or\ \ j = \text{randomRange}\,(1, n)\,; \\ \boldsymbol{x}_{i,j}^{g}, & \text{otherwise.} \end{cases} \tag{9}$$

In the above, function random() generates a random number in [0, 1], CR ($\in$ [0,1]) is the crossover parameter, and the integer j is a randomly chosen index in {1, 2,..., n} that ensures candidate $\boldsymbol{u}_i^{g+1}$ get at least one parameter from trial parameter vector $\boldsymbol{v}_i^{g+1}$ but not all from $\boldsymbol{x}_{i,j}^{g}$.

The selection scheme used in DE is deterministic and based on local optimization, the better one of candidate $\boldsymbol{u}_i^{g+1}$ and parent $\boldsymbol{x}_{i,j}^{g}$ is chosen to enter the next generation:

$$\boldsymbol{x}_i^{g+1} = \begin{cases} \boldsymbol{u}_i^{g+1} & if\ \boldsymbol{u}_i^{g+1} \text{ better than } \boldsymbol{x}_i^{g} \\ \boldsymbol{x}_i^{g} & \text{otherwise.} \end{cases} \tag{10}$$

3.2 Knowledge Structures Defined in Belief Space and Their Influences to Population Space

Belief space includes a series of knowledge structures. Saleem (2001) gave several knowledge structures for reference: situational knowledge, normative knowledge, history knowledge, domain knowledge and topographic knowledge. According to the features of an actual problem, users can select part of these structures or define their own knowledge structures. Here, we redefine three knowledge structures and give the ways they work, based on the characteristics of DE and multi-objective optimization. These knowledge structures are described as follows.

3.2.1 Situational Knowledge

Situational knowledge consists of the best exemplars obtained along the evolution process. This knowledge structure is similar to an external population or archive set (Zitzler et al. 2001) used in many other MOEAs. These exemplars represent leaders for the other individuals to follow. Table 1 shows the structure of situational knowledge.

Where $\boldsymbol{X}_i$ (i = 1, 2...N_Q) is an elite individual, N_Q is the size of situational knowledge structure. Situational knowledge influences the mutation operator of DE in the following way: the individuals $\boldsymbol{x}_{r_1}^{g}, \boldsymbol{x}_{r_2}^{g}, \boldsymbol{x}_{r_3}^{g}$ in formula (8) for mutation operation are chosen from these exemplars, but not from population space. Because these exemplars are the non-dominated individuals found along the evolutionary process, generating new individuals by these individuals may accelerate convergence of the

Table 1 Structure of situational knowledge

$\boldsymbol{X}_1$	$\boldsymbol{X}_2$	......	$\boldsymbol{X}_i$	......	$\boldsymbol{X}_{N_Q}$

Table 2 Structure of normative knowledge

l_1	l_2	…	l_j	…	l_{n-1}	l_n
u_1	u_2	…	u_j	…	u_{n-1}	u_n

algorithm. This scheme is like the DE/best/1 scheme (Storn 1996) in single-objective optimization.

Situational knowledge is the basis for other knowledge. To update belief space, function *accept*() adds all non-dominated individuals (named $\boldsymbol{NDSet(g)}$) in current population space $\boldsymbol{P^g}$ to situational knowledge structure in the following way. For each individual $\boldsymbol{w}_i^g$ in $\boldsymbol{NDSet(g)}$: if $\boldsymbol{w}_i^g$ is not dominated by any individual in situational knowledge structure, then add $\boldsymbol{w}_i^g$ into situational knowledge structure and delete the individuals dominated by $\boldsymbol{w}_i^g$, otherwise, discard it; If the size of situational knowledge structure is larger than N_Q, a truncation operation is needed to eliminate a redundant individual. Here, we use the crowding distance metric (Deb et al. 2002) to pick out the individual having the minimal crowding distance.

3.2.2 Normative Knowledge

The normative knowledge contains the intervals for the decision variables where non-dominated solutions have been found, in order to move new solutions towards these intervals. The normative knowledge has the structure shown in Table 2.

Where l_j and u_j are the lower and upper bounds on the *j*-th dimension of the individuals in situational knowledge structure. This knowledge structure influences the population space as follows. DE mutation operation may lead to new vectors that fall outside the boundaries of the variables, and the original way we deal with this case is making them equal to the boundaries. Here, we use normative knowledge structure to substitute the original variables boundaries. Since the intervals in normative knowledge structure are more constringent than the original variables boundaries and these intervals are the spaces where excellent individuals have been found, so searching in these spaces may improve the convergence rate of the algorithm. Update of normative knowledge is simple: after updating the situational knowledge, it just needs to recover the lower and upper bounds of the exemplars in the updated situational knowledge.

3.2.3 History Knowledge

This knowledge was originally proposed for dynamic objective functions (Saleem 2001), and it was used to find patterns in the environmental changes. Here we design this knowledge to monitor the searching process of the algorithm and guide the searching direction. Almost all evolutionary algorithms have premature convergence problem, DE is without exception. We use history knowledge to monitor convergence state of the algorithm and take an adaptive Cauchy mutation (ACM) to overcome premature convergence problem. The structure of history knowledge is shown in Table 3.

Where $C(P^g)$ is a running convergence performance metric (Deb and Jain 2002), which evaluates convergence of current non-dominated solutions to a reference set. The reference set of points $\boldsymbol{P}^*$ can be either a set of Pareto-optimal points (if known, such as benchmark testing functions) or the non-dominated set points in a combined

Table 3 Structure of history knowledge

$C(P^{g-h+1})$	$C(P^{g-h+2})$	...	$C(P^{g})$	ξ_t
Diversity(1)	Diversity(2)	...	Diversity(n)	ACM operator

pool of all generation-wise populations obtained from an MOEA run. ξ_t is defined as $|C(P^{(g)}) - C(P^{(g-h+1)})|/C(P^{(g)})$, and it represents the variation degree of $C(P^g)$ along searching process. Diversity (j) describes the diversity of the j-th dimension variables of the individuals in situational knowledge. History knowledge works in the following manner. $C(P^g)$ and ξ_t are calculated at every h (here we set it to 20) generations. Then we check whether ξ_t is smaller than a threshold ξ_{CP} (given beforehand). If so, then we calculate the diversity(j) ($j = 1, 2, ..., n$) and check whether diversity(j) is less than a given threshold ε. If so, a Cauchy mutation operation on this dimension is carried out, as follows:

$$\boldsymbol{x}_{i,j}^{g} = \boldsymbol{x}_{i,j}^{g} {}^{*} (1 + \eta * \mathrm{C}(0, 1)) \ if \ \text{diversity}\,(j) < \varepsilon, \quad i = 1, 2 \ldots N_Q \tag{11}$$

Where $\boldsymbol{x}_{i,j}^{g}$ is the value of the j-th dimension of the i-th individual in situational knowledge structure, C(0, 1) is a standard Cauchy variable, η is the coefficient of Cauchy mutation. Here, we calculate diversity(j) as follows:

$$\text{diversity}(j) = \sqrt{\frac{1}{N_Q} \sum_{i=1}^{N_Q} \left(\frac{\boldsymbol{x}_{i,j}^{g} - \overline{\boldsymbol{x}_{j}^{g}}}{u_j - l_j} \right)^2} \tag{12}$$

Where, $\overline{\boldsymbol{x}_{j}^{g}}$ is the average value of the variables of the j-th dimension.

There are three parameters in adaptive Cauchy mutation: the diversity threshold ε, convergence accuracy threshold ξ_{CP} and the coefficient of Cauchy mutation η. ε and ξ_{CP} are often set to 0.01~0.1, here we set ε to 0.02, ξ_{CP} to 0.05. η controls the intensity of Cauchy mutation and usually set to 0.1~0.5. A larger η will result in a more intense mutation that increases the probability of getting out from local optima; however, too intense mutation may affect the convergence performance of the algorithm. In this paper, we use a linear variable η as follows: $\eta = 0.5$–$0.4*g/GenNum$, $g \in [1, GenNum]$, where *GenNum* is the total generations.

The reason why we adopt this dimensional based mutation is based on the analysis of premature convergence of DE. The mutation operator and crossover operator of DE are based on the scale of dimension. When the decision variables of one dimension are nearly equal and tend to be a constant (named C_j, and it is a local optimal value of this dimension), the j-th dimension value of differential vector is close to zero, thus the algorithm can't generate new values differing from C_j by its own operators, so it loses searching ability on the j-th dimension. To overcome this problem, we monitor the convergence process and take an ACM operation on the individuals in situation knowledge structure to keep their diversity, when these individuals may assemble at local optima.

ACM is a fine-grained mutation based on dimension. It doesn't impact on the variables of other dimensions, so it can help the algorithm overcome premature convergence without influencing the algorithm's convergence rate markedly. Pseudo-codes of ACM are described in Fig. 2.

Fig. 2 Pseudo-codes of ACM

```
If g mod h = 0 and ∇_CP < ξ_CP then
Begin
   For j=1 to n
   Begin
       Calculate diversity (j);
       If diversity (j) < ε then
           Cauchy mutation operation on Q_j^g ;
   End for
End if.
```

3.3 Modification of DE's Selection Operator for Multi-objective Optimization

It is needed to point out that the selection operation in population space is modified to suit multi-objective optimization. In single-objective optimization, it is easy to decide which is better between two individuals. But for multi-objective optimization problems, the decision is not so straightforward. Here we use the truncation method also used in PDEA (Madavan 2002) and DEMO (Rolic and Filipic 2005), as follows. The candidate replaces the parent if it dominates the parent; if the parent dominates the candidate, the candidate is discarded; otherwise, when the candidate and parent are non-dominated with regard to each other, the candidate is added to the population. This step is repeated until N_P candidates are created. After that, we get a population with the size between N_P and $2{\cdot}N_P$. If the population has expanded (larger than N_P), we have to truncate it to prepare it for the next generation. The truncation operation includes sorting the individuals with non-dominated sorting (Deb et al. 2002) and evaluating the individuals of the same front with crowding distance metric (Deb et al. 2002). The best N_P individuals (with regard to these two metrics) enter the next population.

3.4 Outline of MOCDE

The outline of MOCDE is described as follows and the flow chart of MOCDE is shown as Fig. 3.

Step1. Initialization. Initialize population space, belief space and parameters of MOCDE. Set generation counter $g = 0$.

Step2. Population space evolution.

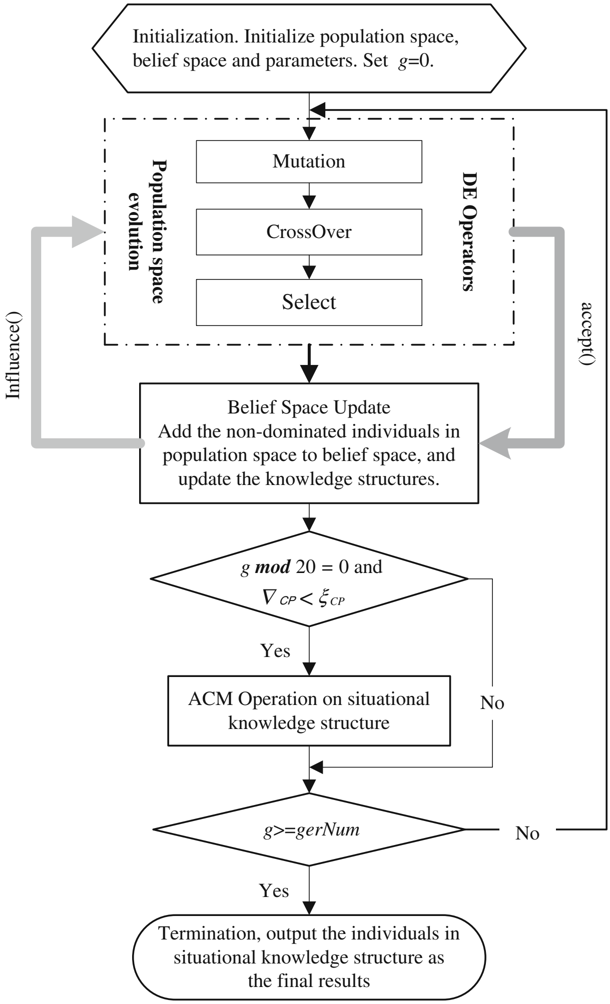
(1) Mutation: Apply DE mutation operator (modified according to the knowledge structures in belief space) to $\boldsymbol{P}^{\mathbf{g}}$, then get a trial population $\boldsymbol{V}^{\mathrm{g}}$.

(2) Crossover: Apply DE crossover operator to the corresponding individuals in $\boldsymbol{P}^{\mathbf{g}}$ and $\boldsymbol{V}^{\mathrm{g}}$, get a set of candidates $\boldsymbol{U}^{\mathrm{g}+1}$.

(3) Selection: compare the corresponding individuals in $\boldsymbol{P}^{\mathbf{g}}$ and $\boldsymbol{U}^{\mathrm{g}+1}$, the superior individuals are added to population $\boldsymbol{P}^{\mathbf{g+1}}$. If $\boldsymbol{P}^{\mathbf{g+1}}$ is too large ($|\boldsymbol{P}^{\mathbf{g+1}}| > N_p$), truncate it to$N_p$.

Step3. Update of belief space. Add the non-dominated individuals in current population space to belief space, and update the knowledge structures in belief space.

Fig. 3 Flow chart of MOCDE



Step4. If generation g **mod** $h = 0$ and the parameter in history knowledge structure ∇_{CP} is lower than ξ_{CP}, then apply adaptive Cauchy mutation to the individuals in situational knowledge structure.

Step5. Termination. If $g = GenNum$, output the individuals in situational knowledge structure as the final results; otherwise, increase the generation counter $g = g + 1$, and go to **Step2**.

To handle constrained problems, we use the following strategies (Deb et al. 2002) to determine the dominance relationship between two individuals: (1) if one individual is feasible and the other one is infeasible, the feasible one dominates the

Table 4 Benchmark test problems

Problems	n	Variable bounds	Objective functions (&Constraints)	Optimal solutions	Comments
ZDT1	30	[0,1]	$f_1(x) = x_1,\ \ f_2(x) = g\left[1 - \sqrt{f_1/g}\right]$ $g(x) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$	$x_1 \in [0,1]$ $x_1 = 0$ $i = 2, 3, \cdots, n$	Convex
ZDT3	30	[0,1]	$f_1(x) = x_1,\ \ f_2(x) = g\left[1 - \sqrt{f_1/g}\right.$ $\left. - (f_1/g)\sin(10\pi f)_1\right]$ $g(x) = 1 + 9(\sum_{i=2}^{n} x_i)/(n-1)$	$x_1 \in [0,1]$ $x_i = 0$ $i = 2, 3, \cdots, n$	Convex, discontinuous
ZDT4	10	$x_1 \in [0,1]$ $x_1 \in [-5,5]$ $i = 2, 3, \cdots, n$	$f_1(x) = x_1,\ \ f_2(x) = g\left[1 - \sqrt{f_1/g}\right]$ $g(x) = 1 + 10(n-1) + \sum_{i=2}^{n}\left[x_i^2 - 10\cos(4\pi x_i)\right]$	$x_1 \in [0,1]$ $x_i = 0$ $i = 2, 3, \cdots, n$	Non-convex
ZDT6	10	[0,1]	$f_1(x) = 1 - \exp(-4x_1)\sin^6(4\pi x_i)$ $f_2(x) = g\left[1 - (f_1/g)^2\right]$ $g(x) = 1 + 9\left[(\sum_{i=2}^{n} x_i)/(n-1)\right]^{0.25}$	$x_1 \in [0,1]$ $x_i = 0$ $i = 2, 3, \cdots, n$	Non-convex, non-uniform
SRN	2	$x_1 \in [-20,20]$ $i = 1,2$	$f_1(x) = (x_1 - 2)^2 + (x_2 - 1)^2 + 2$ $f_2(x) = 9x_1 + (x_2 - 1)^2$ $g_1(x) = x_1^2 + x_2^2 \leq 225,\ \ g_2(x) = x_1 - 3x_2 \leq -10$		Constrained
TNK	2	$x_1 \in [0,\pi]$ $i = 1, 2$	$f_1(x) = x_1,\ \ f_2(x) = x_2$ $g_1(x) = -x_1^2 - x_2^2 + 1 + 0.1\cos\left(16\arctan(x_1/x_2)\right) \leq 0$ $g_2(x) = (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.5$		Discontinuous constrained

infeasible one; (2) if two individuals are both feasible, determine the dominance relationship based on the objectives values; (3) if two individuals are both infeasible, the individual having smaller overall constraint violation dominates the other one.

Fig. 4 Pareto front for ZDT1

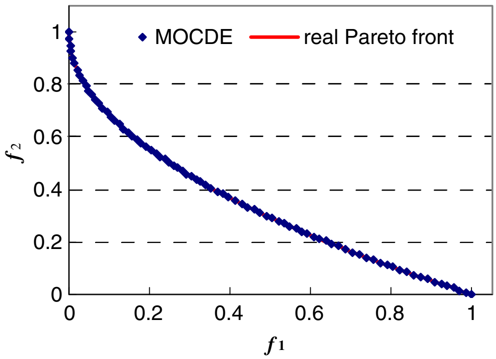
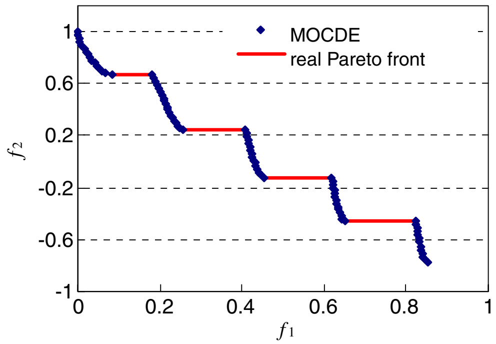


Fig. 5 Pareto front for ZDT3



3.5 Numerical Simulation

3.5.1 Test Problems and Performance Measures

To demonstrate the efficiency of the proposed algorithm for multi-objective optimization problems, several typical and widely used benchmark test problems are chosen to test MOCDE. The main features of these problems are described in Table 4.

Generally, there are two goals in multi-objective optimization: (1) to discover solutions as close to the true Pareto front as possible; (2) to find solutions as diverse as possible in the obtained non-dominated front. To evaluate these two goals, different performance measures have been suggested in literatures. In this paper, we use two widely used performance metrics: Convergence metric γ and Diversity metric Δ (Deb et al. 2002).

3.5.2 Results and Discussion

Main parameters of MOCDE were set as follows. Crossover parameter CR was set to 0.2; scaling factor F was set to 0.5. To match the settings of the algorithms used for comparison, the population size N_P was set to 100 and the algorithm was run for 250 generations. The size of situational knowledge structure N_Q was set to 100.

Fig. 6 Pareto front for ZDT4

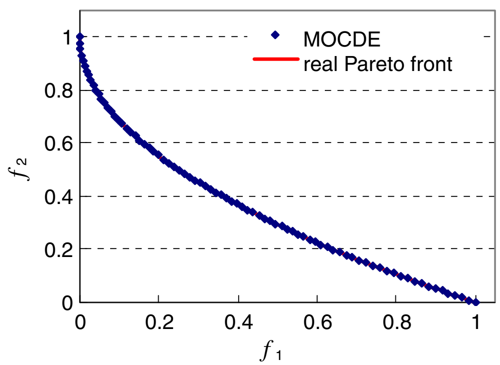
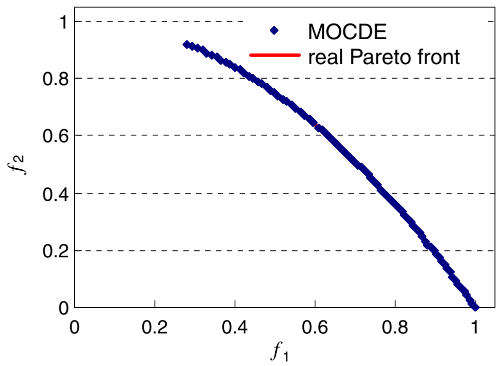


Fig. 7 Pareto front for ZDT6



Figures 4, 5, 6 and 7 show the Pareto fronts obtained by MOCDE and the real Pareto fronts of four ZDT test problems. Table 5 presents the mean (boldfaced font above) and variance (underside) values of the convergence metric γ and diversity metric Δ (zero means that this value is smaller than 10^{-6}) obtained by MOCDE and other contradistinctive algorithms, averaged over ten runs. Results of other algorithms are taken from correlative literatures: Deb et al. (2002) for NSGA-II, Lei and Wu (2005) for SPEA2, Madavan (2002) for PDEA, Xue et al. (2003) for PMODE, Rolic and Filipic (2005) for DEMO/parent, and Qian and Li (2008) for ADEA. Problem SRN and TNK are two constrained test problems. Figures 8 and 9 show the feasible region (marked by blue points) and Pareto fronts obtained by MOCDE (marked by red points).

Table 5 Statistics of results on convergence metric γ and diversity metric Δ (Because the Pareto front of ZDT3 is discontinuous, we delete the distances between break points when calculating Δ.)

Table 5 Statistics of results on convergence metric γ and diversity metric Δ (because the Pareto front of ZDT3 is discontinuous, we delete the distances between break points when calculating Δ)

Metrics	Problems	NSGA-II (real-coded)	SPEA2	PDEA	PMODE	DEMO/ (parent)	ADEA	MOCDE
γ	ZDT1	**0.033482**	**0.023285**	**/**	**0.005800**	**0.001083**	**0.002741**	**0.000150**
		0.004750	0	/	0	0.000113	0.000385	0
	ZDT3	**0.114500**	**0.018409**	**/**	**0.021560**	**0.001178**	**0.002741**	**0.000027**
		0.007940	0	/	0	0.000059	0.000120	0
	ZDT4	**0.513053**	**4.9271**	**/**	**0.638950**	**0.001037**	**0.100100**	**0.000190**
		0.118460	2.703	/	0.500200	0.000134	0.446200	0
	ZDT6	**0.296564**	**0.232551**	**/**	**0.026230**	**0.000629**	**0.000624**	**0**
		0.013135	0.004945	/	0.000861	0.000044	0.000060	0
Δ	ZDT1	**0.390307**	**0.154723**	**0.298567**	**/**	**0.325237**	**0.382890**	**0.099545**
		0.001876	0.000874	0.000742	/	0.030249	0.001435	0.000378
	ZDT3	**0.738540**	**0.469100**	**0.623812**	**/**	**0.309436**	**0.525770**	**0.100462**
		0.019706	0.005265	0.000225	/	0.018603	0.043030	0.000198
	ZDT4	**0.702612**	**0.823900**	**0.840852**	/	**0.359905**	**0.436300**	**0.093290**
		0.064648	0.002883	0.035741	/	0.037672	0.110000	0.000288
	ZDT6	**0.668025**	**1.04422**	**0.473074**	/	**0.442308**	**0.361100**	**0.062343**
		0.009923	0.158106	0.021721	/	0.021721	0.036100	0.000127

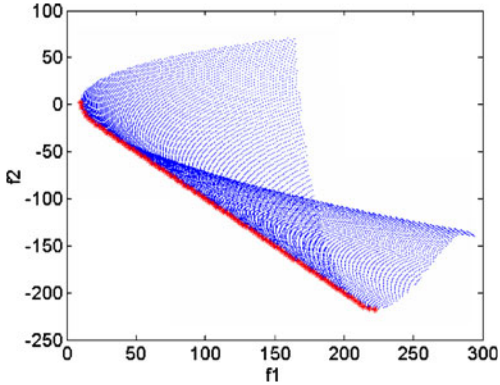


Fig. 8 Pareto front obtained by MOCDE for SRN

From Figs. 4, 5, 6 and 7 we can see intuitively that solutions obtained by MOCDE converge well to the true Pareto front and distribute uniformly. From Table 5 it can be found that MOCDE gets the best convergence performance among the listed algorithms on four ZDT test problems. Especially on problem ZDT4, which has a large number of local Pareto fronts, NSGA-II, SPEA2, PDEA, PMODE and ADEA all have difficulties in converging to the true Pareto front. MOCDE gets good convergence performance on this problem. From the values diversity metric Δ in Table 5 we can see that MOCDE also obtains satisfactory diversity performance, which is much better than other contradistinctive algorithms. From Figs. 8 and 9 we can see that MOCDE handled constraints well and converged to the true Pareto front accurately. After achieving good performance on test problems, next we will apply MOCDE to a practical RFCO problem.

4 Case Study: Multi-objective Flood Control Operation of Three Gorges Reservoir

Three Gorges Project (TGP) locates at the middle of the Yangtze River (Yichang city, Hubei province, China). The index map of Yangtze River basin is presented in Fig. 10 The catchment area of the upper Yangtze River Basin (from headstream

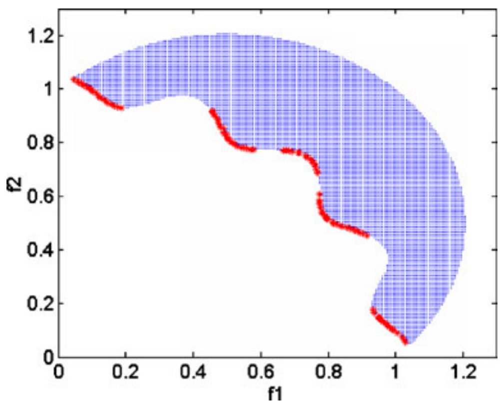


Fig. 9 Pareto front obtained by MOCDE for TNK

Fig. 10 Location map of TGP in the Yangtze River basin

to TGP) is about 1,000,000 km^2 and most of these areas are covered by mountains and gorges. The average annual rainfall in the catchment is 1,203.7 mm, with 75% of the rainfall occurring during flood season (June to September). The main purpose of constructing TGP is providing flood protection to its downstream areas (especially for Wuhan city) of Yangtze River basin in flood season. Furthermore, it also servers for power generation, navigation, water supply, and so on. Main parameters of TGP are presented in Table 6.

The planning and design department of TGP had studied out some basic operating regulations (Zhong 2003) to guide the flood control operation of TGP. The main points of these rules are described as follows. When encountering a not so big flood (smaller than 1% frequency flood), TGP controls its maximum discharge volume no more than 55,000 m^3/s to ensure the safety of downstream areas. If encountering a big flood (between 1% frequency flood and 0.1% frequency flood), TGP controls its discharge volume less than 78,000 m^3/s to decrease flood loss of the downstream areas. The maximum upstream water level of the dam is limited to 175 m, considering the safety of the dam. If the dam front water level reaches 175 m, then increase discharge volume to ensure that the dam front water level doesn't continue to increase. At the end of a flood process, the upstream water level of the dam is expected to fall back to flood control limit level—145 m, to cope with next possible floods.

Table 6 Main characteristics of Three Gorges Reservoir

Characteristics	Quantity	Characteristics	Quantity
Gross storage capacity	393×10^8 m^3	Firm power output	4,990 MW
Regulation storage capacity	269×10^8 m^3	Crest elevation of the dam	185 m
Dead storage capacity	124×10^8 m^3	Normal water level	175 m
Average annual inflow	$4{,}529 \times 10^8$ m^3	Flood control limit level	145 m
Designed discharge capacity	98,800 m^3/s	Dead water level	135 m
Installation capacity	18,200 MW		

4.1 Solution Methodology Based on MOCDE

4.1.1 Individuals Encoding Strategy

The population space consists of N_P individuals and the belief space contains N_Q individuals. In this paper, we use discharge volume as the decision variable to encode the individuals. Every individual vector is expressed as a series of discharge volumes, that is $X_j = \{q_j^1, q_j^2, \ldots, q_j^t, \ldots, q_j^T\}$, where q_j^t is the discharge volume of the j-th individual in the t-th period.

4.1.2 Dimensionless Conversion of the Objective Values

It needs to sum different objective function values when calculating crowding distance (Deb et al. 2002). However, these objective functions have different dimension, so we need to convert them into dimensionless quantities. The conversion formulae are as follows.

$$F_1 = \left(Z_{\max} - \underline{Z}\right) \Big/ \left(\overline{Z} - \underline{Z}\right) \tag{13}$$

$$F_2 = \left(Q_{\max} - \underline{Q}\right) \Big/ \left(\overline{Q} - \underline{Q}\right) \tag{14}$$

Where $\overline{Z}$ and $\underline{Z}$ are the lower and upper bounds of upstream water level, $\overline{Q}$ and $\underline{Q}$ are the lower and upper bounds of discharge volume.

4.1.3 Constraints Handling

We still use the constraint handling strategies described in Section 3.4. However, we have difficulties in calculating the total violation of an individual, because different constraints are often incommensurable with each other for real world optimization problems. It is hard to determine proper weight coefficients to sum the total

Table 7 Non-dominated operation schemes of 5% frequency typical flood in 1954

Scheme index	Max Z_t (m)	Max Q_t (m^3/s)	Z_T (m)	$\lvert Z_T - Z_{FL} \rvert$ (m)	Scheme index	Max Z_t (m)	Max Q_t (m^3/s)	Z_T (m)	$\lvert Z_T - Z_{FL} \rvert$ (m)
1	151.8	55,000	147.2	2.20	16	157.0	49,535	156.0	11.02
2	152.2	54,731	147.9	2.87	17	157.3	49,236	156.5	11.52
3	152.5	54,265	148.6	3.57	18	157.7	48,998	157.0	11.98
4	152.9	53,888	149.3	4.27	19	158.1	48,700	157.5	12.52
5	153.3	53,489	150.0	4.98	20	158.4	48,489	158.0	12.93
6	153.6	53,107	150.6	5.64	21	158.9	48,211	158.5	13.45
7	154.0	52,739	151.2	6.24	22	159.3	48,014	159.0	13.96
8	154.3	52,369	151.8	6.82	23	159.9	47,729	159.7	14.65
9	154.7	51,975	152.5	7.45	24	160.5	47,576	160.2	15.20
10	155.0	51,598	153.0	8.02	25	161.1	47,396	160.9	15.90
11	155.4	51,189	153.6	8.58	26	161.6	47,109	161.5	16.50
12	155.7	50,861	154.1	9.05	27	162.1	46,899	162.1	17.05
13	156.0	50,503	154.6	9.57	28	162.9	46,591	162.9	17.89
14	156.3	50,158	155.1	10.07	29	163.5	46,368	163.4	18.44
15	156.6	49,854	155.6	10.57	30	164.0	46,158	164.0	18.97

Table 8 Non-dominated operation schemes of 0.2% frequency typical flood in 1981

Scheme index	Max Z_t (m)	Max Q_t (m^3/s)	Z_T (m)	$\lvert Z_T - Z_{FL}\rvert$ (m)	Scheme index	Max Z_t (m)	Max Q_t (m^3/s)	Z_T (m)	$\lvert Z_T - Z_{FL}\rvert$ (m)
1	152.1	77,911	145.0	0	16	163.7	58,152	162.7	17.72
2	152.9	76,135	145.0	0	17	164.4	57,026	163.8	18.75
3	153.7	74,657	145.0	0	18	165.2	55,815	164.8	19.82
4	154.5	73,211	146.3	1.33	19	166.0	54,755	165.8	20.76
5	155.2	71,937	148.0	3.03	20	166.7	53,660	166.7	21.71
6	155.9	70,668	149.8	4.75	21	167.6	52,651	167.6	22.56
7	156.6	69,404	151.2	6.18	22	168.4	51,626	168.4	23.41
8	157.4	68,205	152.6	7.58	23	169.2	50,611	169.2	24.24
9	158.2	66,878	153.9	8.93	24	170.1	49,662	170.1	25.12
10	159.0	65,565	155.4	10.40	25	171.0	48,429	171.0	26.00
11	159.7	64,388	156.5	11.53	26	171.7	47,508	171.7	26.72
12	160.6	62,921	158.0	13.03	27	172.5	46,569	172.5	27.47
13	161.4	61,692	159.3	14.29	28	173.3	45,520	173.3	28.30
14	162.1	60,540	160.4	15.43	29	174.1	44,493	174.1	29.12
15	162.9	59,386	161.6	16.61	30	175.0	43,394	175.0	30.00

violation. Considering that the constraints mainly contain water level (or reservoir storage volumes) limit and discharge volume limit, and the water level limit can be converted into discharge volume limit through the water balance equation (formula 6), so we just need to handle discharge volume constraint and don't need these weight coefficients, which makes the constraint handling of this problem easier.

4.2 Results and Discussion

MOCDE is applied to deal with two typical floods: 5% frequency flood (20-year flood) in the year 1954 and 0.2% frequency flood (500-year flood) in 1981. The starting regulation levels for these two floods are the flood control limit level—145 m, and floods end levels are also expected (but not forced) to be back to this level. The period numbers of the floods in 1954 and 1981 are 95 and 40, respectively. Duration of each period is 6 h. The main parameters of MOCDE are set as follows. Crossover parameter CR is set to 0.2, scaling factor F is set to 0.1, size of population space

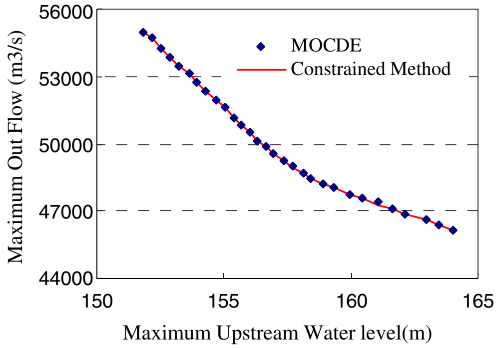


Fig. 11 Pareto front obtained by MOCDE and constrained method for 5% frequency flood in 1954

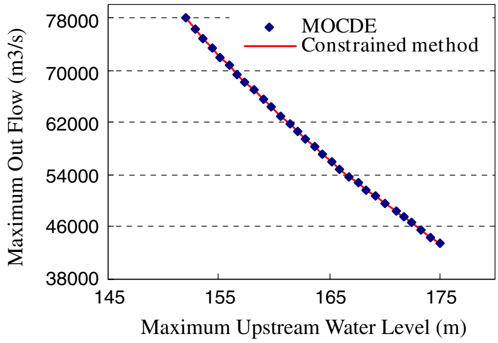


Fig. 12 Pareto front obtained by MOCDE and constrained method for 0.2% frequency flood in 1981

N_P is set to 50, the individual number in belief space N_Q is set to 30, and the total generation number *GenNum* is set to 1,000.

Tables 7 and 8 list the obtained non-dominated schemes for these two floods, besides max Z_t and max Q_t, the values of final water level and its violation $|Z_T - Z_{FL}|$ are also listed. From Table 7 we can see that, for the operation schemes set of 5% frequency typical flood in 1954, peak flood discharges are in the range that from 46,158 to 55,000 m^3/s and do not exceed 55,000 m^3/s. Correspondingly, maximum water level are decreased from 164.0 to 151.8 m. They all satisfy the relevant constraints. Final water levels are in the range of 147.2 to 164.0 m that can't turn back to 145 m because of the peak flood discharge restriction and they decrease as the maximum water levels decrease. Similarly, from Table 8, it can be seen that the operation schemes set of 0.2% frequency typical flood in 1981 satisfy the relevant constraints. The range of maximum water levels and final water levels are almost 30 m, because this flood has larger flood volume. The maximum water levels also decrease as the peak flood discharges increase. Where, the final water level of scheme 1 to 3 can turn back to 145 m due to their larger flood discharges. The final water levels of other schemes can't turn back to 145 m and they decrease as the maximum water levels decrease.

Figures 11 and 12 show the Pareto front obtained by MOCDE and constrained method. The constrained method works as follows: for every scheme obtained by MOCDE, we fix the value of objective F_2 and just optimize objective F_1, using DP

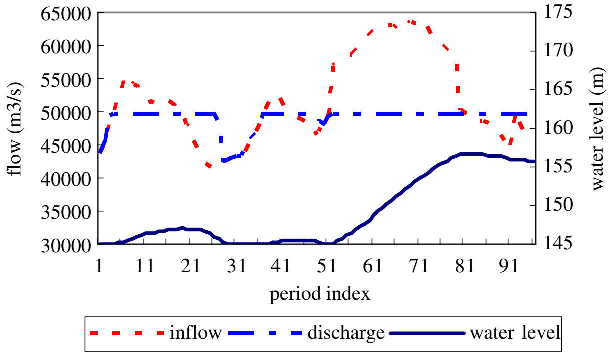


Fig. 13 Discharge process and water lever process of scheme 15 for the flood in 1954

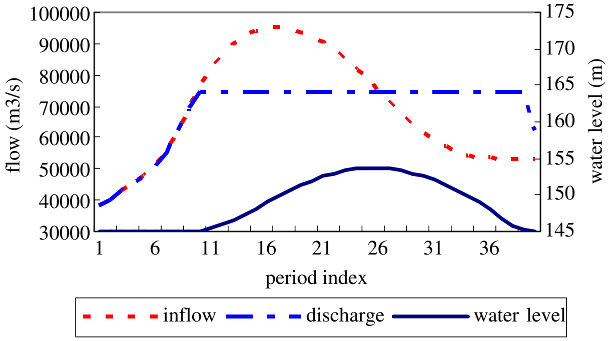


Fig. 14 Discharge process and water lever process of scheme 3 for the flood in 1981

method (Mei 1999). In DP, period water level is set as state variable and the discrete precision is 0.01 m. Generally, DP can obtain the theoretic true Pareto optimal with a certain discrete precision. From Figs. 11 and 12 we can see that the solutions obtained by MOCDE converge well to the Pareto front obtained by DP and distributed widely with good diversity performance. By using constrained method with DP, the RFCO problem is handled as single-objective problem and just one solution is obtained in a run. Thus, a number of repeats are needed to obtain a set of schemes. For DP, it needs about 41 s to get a scheme for the flood in 1954 under the discrete precision of 0.01 m, and for the flood in 1981, the time is about 25 s. The total computational times to get 30 schemes for these two floods are about 20.5 and 12.5 min. In contrast, the average computational times of MOCDE to get 30 schemes for these two floods are just 57 and 38 s, respectively, which are much shorter than the constrained method with DP. Therefore, MOCDE is more efficient and practical for solving RFCO problem than DP. The detailed discharge processes and water level processes of some typical schemes (the 15-th scheme in Table 7 and 3-th scheme in Table 8) are shown in Figs. 13 and 14, respectively.

In practical application, the flood forecasting may not be so accurate and need to be modified with time. Since MOCDE obtains trade-offs fast, we can use it to re-generate alternative schemes according to the modified flood flows in time. This enhances the engineering practicability of MOCDE. After getting a set of non-dominated schemes, decision makers could pick out a compromise scheme as the implementary scheme, according to actual requirements.

5 Conclusions

Reservoir flood control operation is a complex multi-objective optimization problem with a number of constraints. To solve this problem effectively, we proposed a novel multi-objective optimization algorithm, named multi-objective cultured differential evolution (MOCDE), which syncretizes the advantages of DE and cultural algorithm. MOCDE uses cultural algorithm as its framework and DE in its population space. DE converges fast but has the premature convergence problem when dealing with problems which have a large number of local optima. To overcome this problem and improve the algorithm's convergence speed, three knowledge structures are defined in belief space, according to the features of DE and multi-objective optimization. MOCDE updates these three knowledge structures by accepting excellent individuals in evolving population space and uses these knowledge structures to improve its searching efficiency. Tested by several typical benchmark problems, MOCDE shows its efficiency and robust for solving multi-objective optimization problems, especially the ability of avoiding premature convergence. Then we apply MOCDE to a case study—multi-objective flood control of Three Gorges Reservoir. It is found that MOCDE can provide decision-makers many alternative non-dominated schemes with uniform coverage and convergence to true Pareto optimal solutions.

Considering that MOCDE is not presented for RFCO specially, it also can be used as an efficient alternative technique to solve other practical multi-objective optimization problems. In the near future, we plan to use MOCDE to solve multi-objective generation scheduling problem of cascaded hydropower plants. Otherwise, comparing with maximum discharge (objective 2 in this study), a monetary scale would be more intuitionistic and comprehensive to represent downstream flood loss. However, to get the corresponding monetary loss of each discharge, a hydraulic model is needed to be embedded to our flood control model to calculate the submergence depth and flooded area and then count the monetary loss. This is also the further work that we plan to do.

Acknowledgements This work is granted by the National Basic Research Program of China (No. 2007CB714107), Key Projects in the National Science & Technology Pillar Program (No. 2008BAB29B08), and Special Research Foundation for the Public Welfare Industry of the Ministry of Science and Technology and the Ministry of Water Resources (No. 200701008). We thank the reviewers for their excellent comments on this paper.

References

Abbass HA, Sarker R, Newton C (2001) PDE: a Pareto-frontier differential evolution approach for multi-objective optimization problems. In: Proceedings of the Congress on Evolutionary Computation 2001 (CEC'2001), vol 2. IEEE Service Center, Piscataway, NJ, pp 971–978

Alexandre MB, Darrell GF (2008) Use of multi-objective particle swarm optimization in water resources management. J Water Resour Plan Manage 134:257–265

Becerra RL, Coello CAC (2006) Cultured differential evolution for constrained optimization. Comput Methods Appl Mech Eng 195:4303–4322

Chen L, McPhee J, Yeh WWG (2007) A diversified multiobjective GA for optimizing reservoir rule curves. Adv Water Resour 30:1082–1093

Cheng CT, Wang BD (1995) Fuzzy optimal model of reservoir flood operation by means of heuristic approach and human–computer iteration technique (in Chinese). J Hydraul Eng 26(11):71–76

Deb K (2001) Multi-objective optimization using evolutionary algorithms. Wiley, Chichester

Deb K, Jain S (2002) Running performance metrics for evolutionary multi-objective optimization. Technical Report 2002004, KanGAL, Indian Institute of Technology, Kanpur 208016, India

Deb K, Pratap A et al (2002) A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6:182–197

Iacoban R, Reynolds RG, Brewster J (2003) Cultural swarms: modeling the impact of culture on social interaction and problem solving. In: Proceedings of the 2003 IEEE Swarm Intelligence Symposium, Indianapolis, Indiana, USA, IEEE Service Center, pp 205–211

Janga Reddy M, Nagesh Kumar D (2006) Optimal reservoir operation using multi-objective evolutionary algorithm. Water Resour Manag 20:861–878

Janga Reddy M, Nagesh Kumar D (2007a) Multi-objective particle swarm optimization for generating optimal trade-offs in reservoir operation. Hydrol Process 21:2897–2909

Janga Reddy M, Nagesh Kumar D (2007b) Multi-objective differential evolution with application to reservoir system optimization. J Comput Civ Eng 21:136–146

Jin X, Reynolds RG (1999) Using knowledge based evolutionary computation to solve nonlinear constraint optimization problems: a cultural algorithm approach. In: Proceedings of the 1999 Congress on Evolutionary Computation, Washington DC, IEEE Service Center, pp 1672–1678

Karamouz M, Abesi O et al (2009) Development of optimization schemes for floodplain management: a case study. Water Resour Manag 23:1743–1761

Kim T, Heo JH, Jeong CS (2006) Multi-reservoir system optimization in the Han River basin using multi-objective genetic algorithms. Hydrol Process 20:2057–2075

Lei DM, Wu ZM (2005) Crowding-measure based multi-objective evolutionary algorithm (in Chinese). Chin J Comput 28(8):1320–1326

Li-Chiu C (2008) Guiding rational reservoir flood operation using penalty-type genetic algorithm. J Hydrol 354:65–74

Madavan NK (2002) Multiobjective optimization using a Pareto differential evolution approach. In: Proceeding of the Congress on Evolutionary Computation (CEC' 2002), vol 2. IEEE Service Center, Piscataway, NJ, pp 1145–1150

Mandal KK, Chakraborty N (2009) Short-term combined economic emission scheduling of hydrothermal power systems with cascaded reservoirs using differential evolution. Energy Convers Manag 50:97–104

Mehta R, Jain SK (2009) Optimal operation of a multi-purpose reservoir using neuro-fuzzy technique. Water Resour Manag 23:509–529

Mei YD (1999) Dynamic programming model and method of cascade reservoirs optimal operation for flood control (in Chinese). J Wuhan Univ Hydr Elec Eng 32(5):10–12

Nagesh Kumar D, Baliarsingh F (2003) Folded dynamic programming for optimal operation of multireservoir system. Water Resour Manag 17:337–353

Nagesh Kumar D, Baliarsingh F, Srinivasa Raju K (2009) Optimal reservoir operation for flood control using folded dynamic programming. Water Resour Manag. doi:10.1007/s11269-009-9485-3

Qian WY, Li AJ (2008) Adaptive differential evolution algorithm for multi-objective optimization problems. Appl Math Comput 201:431–440

Reynolds RG (1994) An introduction to cultural algorithms. In: Proceedings of the 3th annual conference on evolution programming, Sebalk, A.V. Fogel, World Scientific, River Edge, NJ, pp 131–136

Rolic T, Filipic B (2005) DEMO: Differential evolution for multi-objective optimization. In: Lecture notes in computer science. Springer, Berlin, pp. 520–533

Saleem SM (2001) Knowledge-based solution to dynamic optimization problems using cultural algorithms. Ph.D. thesis, Wayne State University, Detroit, Michigan

Schaffer JD (1985) Multiple objective optimization with vector evaluated genetic algorithms. In: Proceedings of the 1st international Conference on Genetic Algorithms, Hillsdale, NJ, USA

Schultz GA, Plate EJ (1976) Developing optimal operating rules for flood protection reservoirs. J Hydrol 28:245–264

Storn R (1996) On the usage of differential evolution for function optimization. NAFIPS'96, pp. 519–523

Storn R, Price K (1995) Differential evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical report TR-95-012. International Computer Science Institute, Berkeley

Unver OI, Mays LW (1990) Model for real-time optimal flood control operation of a reservoir system. Water Resour Manag 4:20–45

Wei CC, Hsu NS (2008) Multireservoir flood-control optimization with neural-based linear channel level routing under tidal effects. Water Resour Manag 22:1625–1647

Windsor JS (1973) Optimization model for reservoir control. Water Resour Res 9:1219–1226

Xue F, Sanderson AC, Graves RJ (2003) Pareto-based multi-objective differential evolution. In: Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003), vol 2. IEEE Press, Canberra, Australia, pp 862–869

Yuan XH, Zhang YC, Wang L et al (2008) An enhanced differential evolution algorithm for daily optimal hydro generation scheduling. Comput Math Appl 55:2458–2468

Zhong ZY (2003) Flood control plan and role of the Three Gorges Project of the Yangtze river. Yangtze River 34(8):37–40

Zitzler E, Laumanns M, Thiele L (2001) SPEA2: improving the strength Pareto evolutionary algorithm. Technical report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland