Comparative Study of Optimization Algorithms for The Optimal Reservoir Operation

Xinyuan Liu*, Yonghui Zhu and Lingyun Li, Lu Chen*

Key Laboratory of River Regulation and Flood Control of MWR, Changjiang River Scientific Research Institute, Wuhan 430010, China
College of Hydropower & Information Engineering, Huazhong University of Science & Technology, Wuhan, 430074, China.

Abstract-Apart from traditional optimization techniques, e.g. progressive optimality algorithm (POA), modern intelligence algorithms, like genetic algorithms, differential evolution have been widely used to solve optimization problems. This paper deals with comparative analysis of POA, GA and DE and their applications in a reservoir operation problem. The results show that both GA and DES are feasible to reservoir operation optimization, but they display different features. GA and DE have many parameters and are difficult in determination of these parameter values. For simple problems with mall number of decision variables, GA and DE are better than POA when adopting appropriate parameter values and constraint handling methods. But for complex problem with large number of variables, POA combined with simplex method are much superior to GA and DE in time-assuming and quality of optimal solutions. This study helps to select proper optimization algorithms and parameter values in reservoir operation.

1 INTRODUCTION

Reservoir management and operation are one of the most complex problems in water resources management (Simonovic 1987), primarily due to (1) the stochastic nature of stream flows, (2) the multi-objective nature of reservoir operation, (3) the multistage and dynamic nature of reservoir operation decision-making. It is very difficult to find a single model or technique that is universally accepted for operation of reservoir systems (Liu et al. 2011). Due to complexity of reservoir operation problems, lots of optimization algorithms have been introduced and applied to reservoir management and operation, including some traditional methods and modern intelligence algorithms. The performance of these algorithms also carried out a number of comparative studies based on some standard test functions or highly simplified engineering problems; however, these testing problems cannot reflect the complexity and diversity of the calculation of the actual engineering applications. For the optimal reservoir operation, there is still a lack of detailed and comprehensive discussion on the performance of algorithms. In this paper, some optimization algorithms, including traditional algorithm (e.g. POA) and modern intelligent algorithms (e.g. GA and DE), are selected to have a detailed comparative analysis on the performance of reservoir operation from aspects of parameter, operator, variable scale, and constraint handling methods, etc. The results can help to the selection of optimization algorithms for the reservoir operation.

2 DETERMINISTIC OPTIMIZATION MODELS OF RESERVOIR OPERATION AND CONSTRAINT HANDLING METHODS

As an illustrative example, China's Three Gorges Reservoir (TGR) is selected as a case study to evaluate the performance of optimization algorithms. The TGR is a quarterly regulated multipurpose reservoir, with a length of 660 km and a gigantic flood storage capacity of 22.15 × 109 m^3, and plays a very important role in the flood control of the Yangtze River. The practical operation of the TGR needs to consider many utilization objectives, such as flood control, power generation, shipping transport and so on.

2.1 Deterministic Optimization Models of Reservoir Operation.

To facilitate comparison of the algorithm performance, flood control is selected as the operation objective. For an inflow flood, the operation goal of the TGR is to discharge flood water as much as possible on the premise of ensuring the flood control safety of the dam and downstream, in order to vacate enough capacity for the following floods. In the flood control operation model, the decision variable is outflow at each operation stage, and the objective function is to maximize the average outflow with the constraints of mass balance requirement, reservoir storage limits, power generation limits and release requirement, i.e.

* Corresponding author:wishesliu@gmail.com，chen_lu@hust.edu.cn

$$\max \quad O_{avg} = \frac{1}{T}\sum_{t=1}^{T} O_t \tag{1}$$

$$s.t. \quad V_{t+1} = V_t + \left(I_t - O_t\right)\cdot \Delta t\,, \ t = 1,2,\cdots,T \tag{2}$$

$$V_{\min} \le V_t \le V_{\max}\,, \ t = 1,2,\cdots,T \tag{3}$$

$$P_{\min} \le AQ_{ot}H_t \le P_{\max}\,, t = 1,2,\cdots,T \tag{4}$$

$$O_t \le \min\left(O_{\max}^{dam,z_t}, O_{\max}^{down,z_d}\right), t = 1,2,\cdots,T \tag{5}$$

$$O_t = Q_{ot} + Q_{wt}\,, t = 1,2,\cdots,T \tag{6}$$

where T is the total number of time periods in calculation, and Δt is the length of time for each period. A is the coefficient of hydropower generation. I_t and O_t are the reservoir inflow and outflow for the period t, respectively. Q_{ot} and Q_{wt} are the release discharge for power generation and the spilled discharge for the period t, respectively. H_t is the average water head for the period t. $V_{\min}$ and $V_{\max}$ are the minimum and maximum water volume of reservoir, respectively, and V_t is the reservoir storage at the tth period. $P_{\min}$ and $P_{\max}$ the minimum and maximum power limits of reservoir, respectively. $O_{\max}^{down,z_d}$ is the maximum the water discharge of reservoir at the water level Z_t, and $O_{\max}^{down,z_d}$ is the maximum safety discharge in the downstream flood protection section at the water level of Z_d for flood control.

The 100-year design flood based on the 1954 typical flow is selected to evaluate the performance of optimization algorithms. The design flood lasted 30 days, and covered 120 times periods with 6 hours an calculation period. With n defined as the number of continuous periods from the first period, several optimization problems with different numbers of decision variables can be designed when n takes different values, such as, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110 and 120.

2.2 Constraint Handling Methods.

Complex constraints handling has been the emphasis and difficulty in the practical application of the various algorithms, and have a direct influence on the performance of algorithms. There are many constraints handling methods, such as discarding infeasible solutions, repairing infeasible solutions, penalty function, multi-objective method, and so on (Coello 2002), among which repairing infeasible solutions and penalty function are more feasible methods for the high-dimensional multi-constraint problems of reservoir operation. Repairing algorithm is problem-dependent, and needs to be designed according specific reservoir operation problem. Penalty function approach has to be introduced extra empirical parameters to the violation penalty of infeasible solutions to find the intersection of a line with the boundary of the feasible region.

3 PROGRESS OPTIMALITY ALGORITHM

Howson and Sancho (1975) presented the progress optimality algorithm (POA) for the solution of multi-state dynamic programming problems. In the algorithm, progressive optimality theory based on Bellman's principle of optimality was introduced, which indicates that each pair of decision sets is optimal in relation to its initial and terminal values (Howson 1973). Traditional algorithm is conceptually simple, iteratively applying a general two-stage solution to successive overlapping stages of the problem, and giving successive approximations that converge to the optimal path (Howson 1975). We consider that the objective function is

$$\sum_{j=1}^{N} \mathrm{g}(X_{j-1}, X_j) \tag{7}$$

where X_0, X_N are given. This is the equivalent to the two-point boundary value problems in dynamic programming. The algorithm solves the minimization of a two-stage problem

$$G(X_{j-1}, X_{j+1}) = \min_{X_j}[\mathrm{g}(X_{j-1}, X_j) + \mathrm{g}(X_j, X_{j+1})] \tag{8}$$

where X_j, j=1,2,3...N are vectors of stage j (Howson 1975).

At present research, the ways of discrete domain have been widely used to get an optimal value. However, it limits the accuracy and capacity of optimal search in some way. So this paper adopts simplex method to deal with two-stage problems of decision optimization search. Simplex method has been presented originally by Splendy et al. (1962). And then Nelder and Mead (1965) have done some improvements aiming at difficulties that they have met in accelerating search or searching in the valley and ridge. The improved method allow simplex to vary its shape, size and orientation to adapt itself to the local contour of the objective function (Shu et al. 1957). The method described for the minimization of a function of n variables, which depends on the comparison of function values at the (n+1) vertices of a general simplex, followed by the replacement of the vertex with the highest value by another point. The simplex adapts itself to the local landscape, and contracts on to the final minimum (Splendy et al. 1962).

4 GENETIC ALGORITHM

Genetic Algorithm (GA) was first introduced by Holland in 1975 .GA is a search heuristic to generate solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover. The main characteristics of GAs include: *1)* In the genetic algorithm, it is a practical matter about the components of genetic chromosome which need encoding firstly and replace the original mathematical expression (Yan et al. 2016). Therefore, GAs can solve discrete, non-convex, discontinuous problems without differentiation. *2)* These methods are inherently parallel,

using a distributed set of samples from the space (a population of strings) to generate a new set of samples (Goldberg 1989). *3)* GAs are probabilistic search procedures. These algorithm can automatically obtain and direct optimized search space and adjust the search direction adaptive without determined rules. Goldberg made a complete description about simple genetic algorithm in the book "Genetic algorithms in search, optimization, and machine learning" (Goldberg 1989). The simple GA, mainly adopting binary coding, simple arithmetic crossover and uniform mutation, has significant limitations in practical engineering problems. The crossover operator is the main operator and has a strong global search capability. For real-coded genetic algorithms, there have been proposed many crossover methods by improving simple arithmetic crossover, including: blend crossover (BLX) (Takahashi and Kita 2001), simulated blend crossover (SBX) (Deb and Agrawal 1995), unimodal normal distribution crossover (UNDX) (Ono et al. 1999) and simplex crossover (SPX) (Tsutsui 1999). The blend crossover (BLX) combines parental information in a component-wise manner in generating offspring and shows good search ability in optimizing separable objective functions (Wen 2006). The mutation operator is an auxiliary operator in genetic algorithm and has certain local search capability. Several mutation operators are widely used in real-coded genetic algorithm (e.g., uniform mutation operators, boundary mutation operators, non-uniform mutation operator, Gaussian mutation operators). The non-uniform mutation operator (Michalewicz 1996) is relatively simple and efficient, and its search space gradually narrowed with the increase of generation.

This paper mainly adopts a more efficient GA, which uses blend crossover (BLX-α) and non-uniform mutation, and analyzes its performance in optimal reservoir operation. α is a constant on the interval (0,1). In addition, the non-uniform mutation contains a parameter β with the integer value ranging from 2 to 5, which determines the degree of dependence of random number perturbation on generation. Both α and β need to be selected based on experience.

5 DIFFERENTIAL EVOLUTION

Differential Evolution (DE) is a very simple population based, stochastic function minimizer which is very powerful at the same time. It was first appeared as a technical report in 1995. DE operates through similar computational steps as a standard GA. However, unlike traditional GA, the defining characteristic of DE is the generation of offsprings (the so called trial vector) by employing the differential mutation operation followed by crossover of mutant vector with a predetermined parent vector (Thangavelu and Velayutham 2015). DE generates new parameter vectors by adding the weighted difference between two population vectors to a third vector. If the trial vector yields a lower cost function value than the target vector, the trial vector replaces the target vector in the following generation. The detail of DE is described by Price et al. (2006).

In order to classify the different kinds of DE model, the notation *DE/x/y/z*, is introduced here. *x* specifies the vector to be mutated which currently can be "rand" (a randomly chosen population vector) or "best" (the vector of lowest cost from the current population); *y* is the number of difference vectors used; *z* denotes the crossover scheme. The current variant is "bin" (Crossover due to independent binomial experiments) (Storn and Price 1997).

There are two main control parameters of DE: scale factor *F* and crossover rate *CR*. *F* is a constant number ranging from 0 to 2, which controls the diversity and convergence of the population by controlling the scaling degree of difference vector. And *CR* is a user chosen parameter on the interval (0, 1), which controls the degree of participation of each dimension in the individual parameters and the balance between global and local search ability.

6 GA AND ITS APPLICATION TO OPTIMAL RESEVOIR OPERATION

GA is a search heuristic to generate solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover. GA was described by Holland (1975). The simple GA, mainly adopting binary coding, simple arithmetic crossover and uniform mutation, has significant limitations in practical engineering problems. In this paper, a more efficient GA is introduced, which uses blend crossover (BLX) and non-uniform mutation based on floating-point encoding. The detail of BLX and non-uniform crossover are described by Michalewicz (1996) and Takahashi and Kita (2001), respectively.

There are several important parameters for the running of the proposed GA. The parameters α in the BLX and β in the non-uniform mutation are set to be 0.7 and 2, respectively, which was found to be more efficient by the author. For the problems with different number of variables, different crossover rates (*Pc*) and mutation rates (*Pm*) were tested in this study. The population size is 500, and the optimization process terminates when the optimal function value achieves the global optimal solution, the maximum generations is more than 50000, or the optimal function value changes less than 1.0. For each parameter pair of the decision variable number, the crossover rate and the mutation rate, the optimization program will run 10 times, of which the average value of the optimal function values and the generations are calculated as the optimal results of each parameter pair. All optimization programs are carried out on a computer with the CPU of Intel Core 2 Quad Q8200 Processor (2.33GHz) and the memory of 4GB.

The results of the problems with three different numbers of decision variables are shown in Fig. 1, Fig. 2 and Fig. 3, respectively. Fig.1 (a), Fig.2 (a) and Fig.2 (a) shows the best objective function value derived by GA with different crossover and mutation rates when the number of variables are 10, 30 and 50, respectively. Fig.1 (b), Fig.2 (b) and Fig.2 (b) shows the maximum generations with different crossover and mutation rates when the number of decision variables is 10, 30 and 50,

respectively. In these figures, the parameter pair of Pc and Pm with darker color block is better. The results show that the optimal function value and the maximum generation are more sensitive to Pm. Pc has more influence on the maximum generation but little to the optimal function value. In addition, the more decision variables, the smaller the probability of achieving global optimal solution. Larger mutation rate may need more generations but achieve worse solutions. It is generally recommended to not more than 0.1 for the mutation rate, and it should be smaller when the decision variables are more. The crossover rate is preferably from 0.6 to 0.8 for 30 decision variables, and smaller for 50 decision variables.

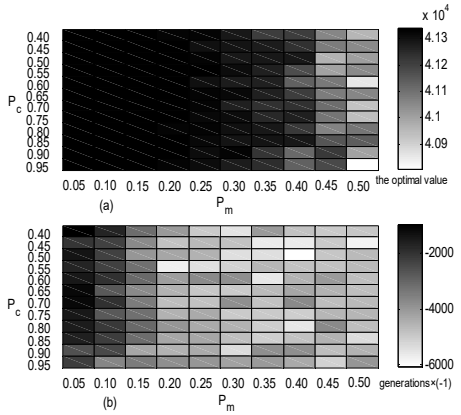

Fig. 2 GA's performance with different crossover and mutation rates (n=30)

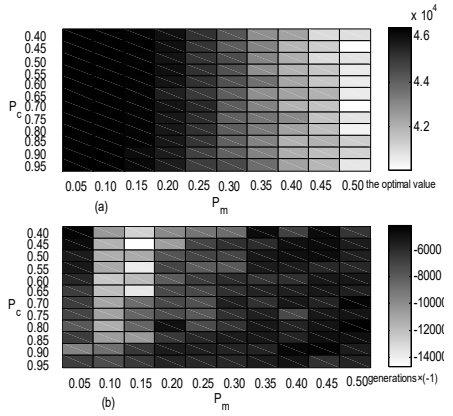

Fig. 3 GA's performance with different crossover and mutation rates (n=50)

7 DE AND ITS APPLICATION TO OPTIMAL RESERVOIR OPERATION

DE, developed by Price and Storn (1997), is also a stochastic population-based optimization algorithm, simple yet powerful. DE generates new parameter vectors by adding a weighted difference vector between two population members to a third member. If the resulting vector yields a lower objective function value than a predetermined population member, the newly generated vector will replace the vector with which it was compared in the following generation. The comparison vector can but need not be part of the generation process mentioned above. In addition the best parameter vector is evaluated for every generation in order to keep track of the progress that is made during the minimization process. The detail of DE is described by Price et al (2005).

There are several variants of DE, among which DE/BEST/1/EXP is more efficient for the reservoir operation problem in this paper. The differential weighting factor (F) and the crossover probability (CR) are two main parameters. F is a positive real number that controls the rate at which the population evolves. While there is no upper limit on F, effective values are seldom greater than 2.0. CR is a user-defined value that controls the fraction of parameter values that are copied from the mutant with a value of 0 to 1. Other parameter values are set to be same to the GA above.

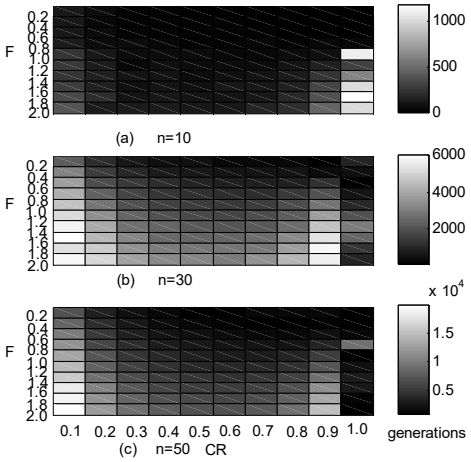

Fig. 4 DE's performance with different parameter pairs

For the problems with three different numbers of decision variables, the optimal global solutions are all achieved by DE. Fig.4 shows the average generation for each parameter pair. The parameter pair with darker block needs less generation and so is better. The results show that the performance of DE with small value of F and large value of CR is better. However, the performance may be poor when CR is more than 0.9. With the increase in the number of decision variables, F need to be increased and CR decreased gradually.

8 COMPARATIVE ANALYSIS OF OPTIMIZATION ALGORITHMS FOR THE RESERVOIR

GA and DE can achieve best results in application to reservoir operation, yet needing proper parameter values. Here, the performances of these modern intelligent algorithms are compared with a traditional algorithm POA.

8.1 POA in combination with simplex method.

POA is presented by Howson and Sancho (1975) for the solution of multi-state dynamic programming problems. It is a method of successive approximation using a general two-stage solution. The algorithm is computationally efficient and has minimal storage requirements. For each

two-stage solution, simplex method for function minimization is used. Simplex method was introduced by Splendy et al (1962) for tracking optimum operating conditions by evaluating the output from a system at a set of points forming a simplex in the factor-space, and continually forming new simplexes by reflecting one point in the hyperplane of the remaining points. The method was improved by Nelder and Mead (1965). The improved simplex adapts itself to the local landscape, and contracts on to the final minimum. The method is shown to be effective and computationally compact.

8.2 Parameter setting of optimization algorithms.

POA adopts penalty function in constraint handling. GA and DE adopt penalty function and repairing algorithm for unfeasible solutions. Parameters of GA and DE are all determined with quite good performance. The crossover and mutation rate are 0.70 and 0.05, respectively. The differential weighting factor and the crossover probability based on DE/BEST/1/EXP are 0.4 and 0.8, respectively. The population sizes of GA and DE are all 500, and the optimization process terminates when the optimal function value achieves the global optimal solution, the maximum generations is more than 50000, or the optimal function value changes less than 1.0. For each number of the decision variables, the optimization program will run 10 times, of which the average value of the optimal function values and the generations are calculated as the optimal results. Generation of POA is defined as the number of loop iterations.

8.3 Comparative analysis of optimization algorithms.

The results are shown in Table 1. It can be seen that POA combined with simplex method is superior to GA and DE combined with penalty function in performance. For each number of decision variables, POA can achieve the global optimal solutions. The performance degrades little with the increase of decision variables, while the probability to the global optimal solutions of GA and DE decreased obviously and time-assuming increases significantly with the increase of variables. Compared with GA, DE combined with penalty function has better solutions and less time-consuming, but hard to obtain feasible solutions for more than 70 decision variables. In combination with GA, repairing algorithm can decrease time-assuming greatly when the number of variables is smaller than 90, but may achieve poorer solutions when then number of variables is larger than 40. In combination with DE, repairing algorithm can also decrease time-assuming greatly and achieve better solutions than penalty function. Repairing algorithm is more efficient than penalty for DE without the limitation of number of variables, but for GA only when the number of variables is small. In addition, GA and DE combined with repairing algorithm are better than POA when the number of variables is less than 30.

9 CONCLUSIONS

GA and DE are all feasible in application to reservoir operation, but have many parameters and are difficult in determination of these parameter values. For simple problems with small number of decision variables, GA and DE are better than traditional algorithms (e.g. POA) when adopting appropriate parameter values and constraint handling methods. But for complex problem with large number of variables, POA combined with simplex method are much superior to GA and DE in time-assuming and quality of optimal solutions.

TABLE I COMPARSION OF SEVERAL OPTIMIZATION METHODS IN PERFORMANCE

Algorithms \ Number of variables		10	20	30	40	50	70	80	100	110	120
The optimal global solution		**36880**	**37940**	**41340**	**44505**	**46404**	**48574**	**49252.5**	**50202**	**50547**	**50835**
POA	Mean of optimal values	**36880**	**37940**	**41340**	**44505**	**46404**	**48574**	**49252.5**	**50202**	**50547**	**50835**
	generations	1	1	1	1	2	2	2	2	2	2
	Time-assuming(s)	0.03	0.06	0.14	0.17	0.28	0.41	0.50	0.47	0.62	0.73
GA (penalty function)	Mean of optimal values	**36880**	**37940**	**41340**	**44503**	**46401**	**48572**	**49249**	**50192**	**50532**	**50810**
	generations	31	162	1425	3838	6570	13857	16029	22875	26034	28711
	Time-assuming(s)	0.18	1.69	21.70	83.76	197	654	900	1700	2169	2610
DE (penalty function)	Mean of optimal values	**36880**	**37940**	**41340**	**44505**	**46404**	**48574**	—	—	—	—
	generations	9	39	629	890	1144	1690	—	—	—	—
	Time-assuming(s)	0.05	0.36	8.80	18.23	31.83	76.06	—	—	—	—
GA (repairing algorithm)	Mean of optimal values	**36880**	**37940**	**41340**	**44505**	**46397**	**48565**	**49224**	**50144**	**50471**	**50572**
	generations	1	1	263	1935	3878	7984	8443	15345	14579	11877
	Time-assuming(s)	0.01	0.03	4.24	41.22	109	337	435	1715	2755	2641
DE (repairing algorithm)	Mean of optimal values	**36880**	**37940**	**41340**	**44505**	**46404**	**48574**	**49252.5**	**50202**	**50547**	**50713**
	generations	1	1	17	52	99	158	205	158	153	1169
	Time-assuming(s)	0.01	0.01	0.29	1.09	2.58	5.92	8.68	8.82	9.75	74.95

ACKONWLEDGEMENTS

This study was financially supported by Natural Science Foundation of China (Grant No. 51409015; 51679094; 51339001); the National Key Research and Development Program of China (Grant No. 2016YFC0402305 and 2016YFC0402310); and the Public Welfare Scientific Research Funding Project of the MWR of China (Grant No. 201401011).

REFERENCES

1. C.A.C. Coello, Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art, Comput. Methods Appl. Mech. Eng. 191(2002)1245-1287.
2. D.E. Goldberg Genetic. Algorithms in search optimization and machine learning Boston: Addison-Wesley Longman Publishing Co. 1989
3. Deb K, Agrawal RB. Simulated binary crossover for continuous search space. Complex Systems, 1995, 9(6): 115-148.
4. M. Takahashi, H. Kita. A Crossover Operator Using Independent Component Analysis for Real-Coded Genetic Algorithm, in Proceedings of the 2001 Congress on Evolutionary Computation, (2001) 643-649
5. Holland J. Adaptation in natural and artificial systems. The University of Michigan Press, 1975, Ann Arbour.
6. H.R. Howson, N.G.F. Sancho, A two-stage algorithm for sequential decisions problems. INFOR II 2 (1973)163-176.
7. H.R. Howson, N.G.F. Sancho, A new algorithm for the solution of multi-state dynamic programming problems, Math.Programm. 8(1975)104-116.
8. J.H. Holland, Adaptation in Nature and Artificial Systems, University of Michigan Press, Ann Arbor, 1975.
9. J. Nelder, R.Mead, A simplex method for function minimization. Computer Journal, (1965)308-313.
10. K.V. Price, R.M. Storn, Differential evolution - a simple evolution strategy for fast optimization. Dr. Dobb's Journal, 22(1997)18-24.
11. K.V. Price, R.M. Storn, J.A. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, Springer-Verlag, New York, 2005.
12. Michalewicz Z. Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, New York, 1996.
13. Ono I, Kita H, Kobayashi S. A robust real-coded genetic algorithm using unimodal normal distribution crossover augmented by uniform crossover: effects of self-adaptation of crossover probabilities. Proceedings of the Genetic and Evolutionary Computation Conference. San Mateo: Morgan Kaufmann Publishers, 1999, 496~503.
14. Price K, Strom R, Lampinen J. Differential evolution: a practical approach to global optimization [M]. Springer Science & Business Media, 2006.
15. Shu-Kai S. Fan, Yun-Chia Liang, Erwie Zahara, A genetic algorithm and a particle swarm optimizer hybridized with Nelder–Mead simplex search. Computer and Industrial Engineering. 50(2006)401-425.
16. S. Simonovic, The implicit stochastic model for reservoir yield optimization, Water Resour Res., 23(1987)2159-2165.
17. Storn R, Price K. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces [J]. Journal of global optimization, 1997, 11(4): 341-359.
18. Takahashi M, Kita H. A crossover operator using independent component analysis for real-coded genetic algorithm. Proceedings of the 2001 Congress on Evolutionary Computation, 2001, pp. 643-649.
19. Thangavelu S, Velayutham C S. An investigation on mixing heterogeneous differential evolution variants in a distributed framework [M]. Inderscience Publishers, 2015.
20. Tsutsui S, Yamamura M, Higuchi T. Multi-parent recombination with simplex crossover in real coded genetic algorithms. Proceedings of the Genetic and Evolutionary Computation Conference. San Mateo: Morgan Kaufmann Publishers, 1999. 657~664.
21. Wen X, Xia Q, Zhao Y. An effective genetic algorithm for circularity error unified evaluation [J]. International Journal of Machine Tools & Manufacture, 2006, 46(14):1770-1777.
22. W. Splendy, G.R. Hext, F.R. Himsworth Sequential application of simplex design in optimization and evolutionary design. Technometrics (1962) 441-461.
23. X. Liu, S. Guo, P. Liu, L. Chen, X. Li, Deriving optimal refill rules for multi-purpose reservoir operation, Water Resour Manage 25 (2011)431-448.
24. Yan X, Liu H, Zhu Z, et al. Hybrid genetic algorithm for engineering design problems [J]. Cluster Computing, 2016, 13(9):1-13.
25. Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, New York, 1996.