Universal Journal of Industrial and Business Management 2(7): 164-172, 2014 http://www.hrpub.org
DOI: 10.13189/ujibm.2014.020704

Applying Differential Evolution to Aggregate Production Planning

Shih-ChangWang[1,*], Ming-Feng Yeh[2]

[1]Department of Business Administration, Lunghwa University of Science and Technology, Taoyuan, Taiwan
[2]Department of Electrical Engineering, Lunghwa University of Science and Technology, Taoyuan, Taiwan
*Corresponding Author: scwang@mail.lhu.edu.tw

Abstract Differential evolution (DE) has become a popular and promising technique in the recent decade. There are many successful applications implemented DE in scientific and engineering fields. However, we rarely found it applying to the field of highly constrained aggregate production planning (APP) problems. Besides, in the course of solving the APP problem, we discover that the most often used DE strategies have unsatisfactory performance to the problem. For alleviating the deficiency, in this paper, we propose an improved DE strategy and a constraint handling mechanism, called the winner-based constraint handling mechanism, for solving the highly constrained APP problem. In the computational study, some instances of the problem are experimented and analyzed to evaluate the performance of the approach with the most often used DE strategies. The experimental results show that the improved DE strategy with the winner-based constraint handling mechanism can provide particular performance for solving the APP problem than other DE strategies.

Keywords Aggregate Production Planning (APP), Differential Evolution (DE), Constraint Handling Mechanism

1. Introduction

Aggregate production planning (APP) is an important technique in Operations Management. It is a mid-term capacity planning that determines a principle of workforce, production, inventory, subcontract, backlog, etc., over a specific time horizon which ranges from 2 to 12 months, to satisfy fluctuated demand requirements(Al-E-Hashem, Aryanezhad, & Sadjadi, 2012; Graves, 2002; Stevenson, 2009). In recent decades, as the advance in heuristic technique and modeling approach, the APP problems discussed have become quite complex and large scaled. Paiva and Morabito (2009) propose an optimization model to support decisions in the APP problem of the sugar and ethanol milling factory. The model is a mixed integer programming formulation based on the industrial process selection and the production lot-sizing model. Sillekens, Koberstein, and Suhl (2011) present a mixed integer linear programming model for the APP problem of flow shop production lines in the automotive industry. In contract to traditional approach, the model considers discrete capacity adaptions which originate from technical characteristics of assembly lines, work regulations and shift planning. Its solution framework containing different primal heuristics and preprocessing techniques embedded into a decision support system. Zhang, Zhang, Xiao, and Kaku (2012) build a mixed integer linear programming model, which characterize an APP problem with capacity expansion in a manufacturing system. The system includes multiple activity centers. They use a heuristic method based on the capacity shifting with linear relaxation to optimize the model. Ramezanian, Rahmani, and Barzinpour (2012) consider an APP problem of the multi-period, multi-product, and multi-machine system with setup decisions. They develop a mixed integer linear programming model for the system. Due to the NP-hard class of the model, they use the genetic algorithm (GA) and the tabu search approach for solving the model.

From literature review, there is a trend in the research community to solve the APP problem by using modern heuristic optimization techniques. This is due to the time-consuming and unsuitability of classical techniques to the problem in many circumstances.

In the last decade, Differential evolution (DE) algorithm has become a promising technique. There are also many successful applications implemented by DE to solve many practical problems. DE was first proposed by Storn and Price in 1995 while trying to solve the Chebyshev polynomial fitting problem (Storn & Price, 1995). It is a simple but powerful stochastic global optimizer that stems from the genetic annealing algorithm developed also by Price. With a randomly initialized population, DE employs simple mutation and crossover operators to generate new offspring, and then utilizes a selection technique to determine whether the offspring will replace their parents into next generation. Since its initiation, DE has proved to be a very efficient and robust technique for function optimization. It has been successfully used to solve various problems in scientific and

engineering fields, such as pattern recognition(Swagatam Das & Konar, 2009; Maulik & Saha, 2009), power dispatch (Chiou, 2009; Varadarajan & Swarup, 2008), control systems(Iruthayarajan & Baskar, 2009; Tang, Xue, & Fan, 2008), and many others (S. Das & Suganthan, 2011).

Despite many successful DE applications implemented in scientific and engineering fields, however, there are not many in the field of management, especially applying to the highly constrained APP problem. Moreover, in the course of solving the problem, we find that the most often used DE strategies have limited ability and inefficiency that need to be improved in dealing with the problem. The reason may be attributed to that DE is originally introduced to solve unconstrained and continuous optimization problems. Its process implies the unrestricted and continuous exploration and exploitation in the search space which may have limited ability to the highly constrained APP problem.

In this study, we first explore the importance of the APP problem, some significant researches and applications are also presented. To the optimization of the APP problem, we find that DE algorithms were less used applying to the problem in researches. We also discover that the most often used DE algorithms have some imperfections in optimizing the highly constrained APP problem. Therefore, in the succeeding context, an improved DE strategy and a constraint handling mechanism called the winner-based constraint handling mechanism are proposed to alleviate the deficiency of DE algorithms for solving the problem. In the final, some test instances of the APP problem, from small-sized to large-sized, are randomly generated to evaluate the performance of the improved DE with the winner-based constraint handling mechanism. The methods are also experimented and compared with the most often used DE strategies, DE/rand/1, DE/best/1 and DE/target-to-best. The statistical result shows that the improved DE strategy with the winner-based constraint handling mechanism possesses particular quality in convergence, accuracy, and reliability for solving the APP problem than the most often used DE strategies.

The rest of the paper is organized as follows: in section 2, the proposed approach of the improved DE strategy with the winner-based constraint handling mechanism for solving the constrained APP problem is revealed. The evaluation experiments and discussions for the approach with the most often used DE strategies are displayed in section 3, some conclusions are addressed in the final of section 4.

2. Improved DE with Winner-Based Constraint Handling Mechanism

2.1. Improved DE

DE is a kind of stochastic population-based search methods. Their operations imply the existence of unrestricted and continuous explorations in search space. Accordingly, there is no guarantee that the global optimum of the highly constrained APP problem will be found.

Our improved DE strategy is inspired by the promising results that used the exponential probability distribution to generate the coefficient of PSO (Krohling & dos Santos Coelho, 2006). We think that generating random numbers for the scale factor F in DE by using the exponential probability distribution is worth adopting. It can provide a good compromise between the probability of having a large number of small perturbations and a small probability of having large perturbations. The density function of the exponential probability distribution is given in (1). In this function, it is obvious that we can control the variance by changing the location parameter a and the scale parameter b. The variance effect of the parameters a and b to the function is shown in Figure 1. The algorithm of generating F by using the exponential probability distribution function is presented in Algorithm 1.

$$f(x) = \frac{1}{2b}\exp(-\frac{|x-a|}{b}), \quad -\infty < x < \infty, \quad a, b > 0 \qquad (1)$$

Algorithm 1. Generating F by the exponential pdf

1	*u1 = rand[0, 1] ;*	% uniform random number [0, 1]
2	*u2 = rand[0, 1] ;*	
3	**if** *u1 > 0.5*	
4	*x = a + b*ln(u2) ;*	% natural logarithm
5	**else**	
6	*x = a - b*ln(u2) ;*	
7	**end**	
8	$F_{\exp}$ *= abs(x) ;*	% absolute value

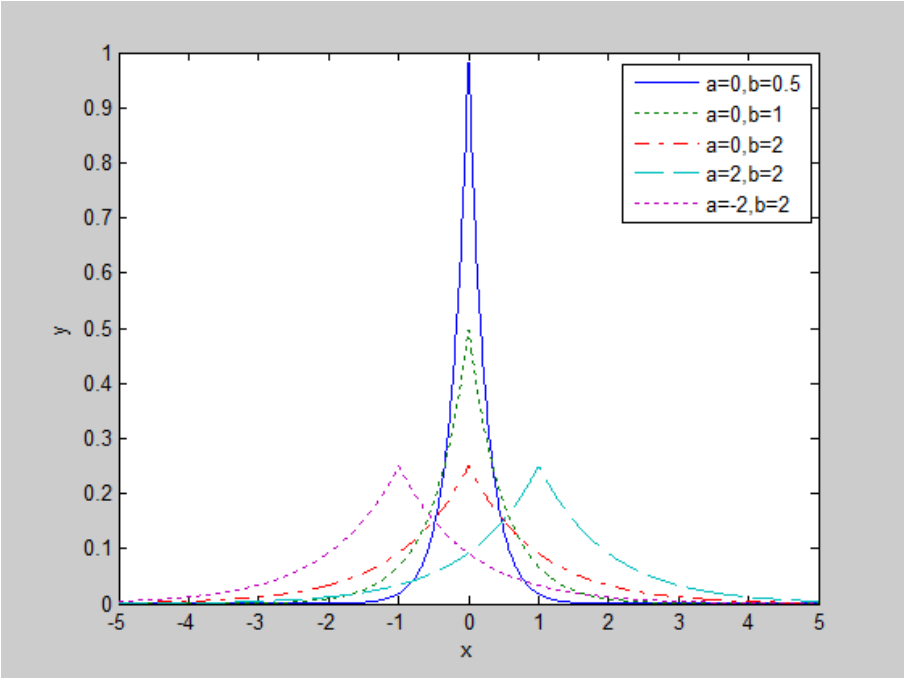

Figure 1. The variance effect of *a* and *b* to the exponential distribution

2.2. Winner-Based Constraint Handling Mechanism

The original APP model with many equality and inequality constraints presented in Wang and Yeh (2014) can be reformed as a new unconstrained model whose new objective function is expressed as $F(x)$ by incorporating a penalty function $P(x)$:

Minimize

$$F(x) = f(x) + P(x) \tag{2}$$

where the penalty function $P(x)$ with the penalty parameter *c* be:

$$P(x) = c\left\|g^{+}(x)\right\|^{2} + \frac{c}{2}\left\|h(x)\right\|^{2} \tag{3}$$

and

$$g^{+}(x) = (g_1^{+}(x),\ ...,\ g_p^{+}(x))^{T},$$

$$\text{With}\quad g_i^{+}(x) = \max(g_i(x),\ 0) \tag{4}$$

$$h(x) = (h_1(x),\ ...,\ h_q(x))^{T} \tag{5}$$

Although the penalty function approach is a common and popular constraint handling technique to deal with constraints in optimization problems, however, the performance of the approach is significantly affected by the choice of the penalty parameter *c*. For getting good results it's ability is limited by the fine-tuning of the penalty parameter that is problem-dependent (Deep & Dipti, 2008). In addition, when incorporating a penalty function into the objective function as the fitness function for evolutionary algorithms, it is particularly important to maintain the diversity in the population and to be able to keep solutions both inside and outside the feasible region (Coello Coello, 2002; Mezura-Montes & Coello Coello, 2003). Several studies have shown that despite its popularity, the penalty function approach, even when used with dynamic penalty parameters, tend to have difficulties to deal with highly constrained search spaces and to deal with problems in which the constraints are active in the optimum (Coello Coello, 2002; Runarsson & Xin, 2000).

Therefore, instead of the popular approach, we propose a mechanism to handle constraints for the APP model without using the penalty parameter at all. It is called the winner-based constraint handling mechanism. The mechanism is a decision-making scheme. It is based on the score of a well-defined scoring function used to decide a winner among candidate solutions when dealing with constrained search spaces in the improved DE, as follows:

The APP model with many equality and inequality constraints can be formally expressed as the formulation:

Minimize

$$f(x) \tag{6}$$

subject to

$$g_i(x) \le 0\ ,\ i = 1,\ ...,\ p \tag{7}$$

$$h_j(x) = 0\ ,\ j = 1,\ ...,\ q \tag{8}$$

$$\forall\ x \ge 0$$

It is relatively common to transform the equality constraints (8) into the form of inequality constraints:

$$\left|h_j(x)\right| - \varepsilon \le 0 \tag{9}$$

where ε is the tolerance of a very small positive value allowed.

Therefore, all constraints and boundaries can be expressed as:

$$l_k(x) \le 0,\ k = 1, 2, ..., r \tag{10}$$

We incorporate a closeness function, defined by (11), to measure the closeness of a solution to the feasible region. A solution x in the APP model is called a candidate, if the function value of a candidate compared with another is larger, we can regard it as closer to the feasible region than the other one.

$$c(x) = \sum_{k=1}^{r} \min(0, -l_k(x)) \tag{11}$$

Thus, a scoring function measuring the degree of candidates in our winner-based constraint handling mechanism can is defined by:

$$score(x) = \begin{cases} c(x) & if\ c(x) < 0 \\ \exp(-f(x)) & o.w. \end{cases} \tag{12}$$

where $score(x) \le 1$, $c(x)$ is the closeness function, $f(x)$ the objective function of the APP model, and exp() the exponential transformation function.

The scoring function is used as decision-making criterion in the winner-based constraint handling mechanism, which selecting winners of candidates into next generation in the improved DE. By calculation of the scoring function, if both candidates compared, then the one that has a higher score wins and can be chosen into next generation. In addition, if the score of a candidate is smaller than 0, it means the candidate violates at least one constraint and is classified as infeasible; otherwise as feasible. The procedure of the improved DE with the winner-based constraint handling mechanism employed to the optimization of the APP model is depicted in Algorithm 2.

In the improved DE with the winner-based constraint handling mechanism, based on the scoring function after iterations, the winner will tend to the feasible region rather than the infeasible, and tend to feasible solution with a better solution gradually. In addition, both the objective and constraint violation of the APP model are considered, forgetting the annoying penalty parameters to adjust at all.

Algorithm 2 The procedure of the proposed approach

1	randomly initialize a population	//*NP D*-dimensional
2	**do** {	parameter vectors
3	**for***i*=1 to *NP*	
4	randomly select $r_1 \neq r_2 \neq r_3 \neq i$;	
5	j_{rand} =*randint[1, D]* ;	
6	**for***j*=1 to *D*	//generate trial vectors
7	**if***rand[0, 1]*<=*Cr*or*j*== j_{rand}	
8	generating F_{exp} ;	//crossover
9		//by using Algorithm 1
10	$u_{ji} = x_{jr_1} + F_{exp} * (x_{jr_2} - x_{jr_3})$;	
11		//mutation
12	**else**	
13	$u_{ji} = x_{ji}$;	
14		
15	**end**	
16	**end**	
17	**end**	
18	**for***i*= 1 to *NP*	//choose winner
19	**if** $score(\mathbf{u}_i) > score(\mathbf{x}_i)$	
20	$\mathbf{x}_i = \mathbf{u}_i$;	
21	**end**	
22	**end**	
23	} **while** (termination criteria not met)	

3. Computational Study

3.1. Evaluation Experiments

In order to know the performance of the improved DE with the winner-based constraint handling mechanism for solving the APP problem, the approach was experimented and compared with three most often used DE strategies, DE/rand/1, DE/best/1 and DE/target-to-best. The test instance of the APP problem is illustrated through the extension of the Red Tomato Tools, a manufacturer of gardening equipment presented in Wang and Yeh (2014).We made use of artificial datasets that 4test instances of the APP model, from small-sized to large-sized, were randomly generated. These instances and DE strategies with the winner-based constraint handling mechanism were coded by MATLAB R2012a and executed on a laptop PC with Intel Core 2 Dual 2.0 GHz CPU, 4 GB RAM, and 64-bit Windows 7 operating system. For establishing the benchmark of comparative performance, each instance was also coded by LINGO optimization solver to discover the global optimum as lower bound (LB). In order to present the diversity of the experimented outcomes between LB and the best objective value derived from each strategy, we defined a quality measurement, called percent deviation, denoted as %*Dev* as follows:

$$\%Dev = \frac{BOV - LB}{LB} \times 100\% \quad (10)$$

Where BOV is the best objective value of an instance experimented from a DE strategy, and LB the lower bound of the instance.

In these DE evolutionary experiments, the population members of a trial for each DE strategy were set to 8 times of the parameter number of the trial. The crossover rate *Cr* and the scale factor *F* were set to 0.9 and 0.85 respectively, as suggested. While in the improved DE strategy, we used the exponential probability distribution function with the parameters of *a*=0 and *b*=0.5 to generate the scale factor *F*. Termination criteria of a trial were set to 5% of above its corresponding lower bound or reached to 5,000 iterations. The experimental results of an instance for each DE strategy were collected in average within 10 trials of execution. The experimental statistics included were the stop iterations, the best objective value occurred, the mean and the standard deviation of the best objective values collected from each iteration when the values are feasible (score > 0),the iteration at which the scores begin to greater than 0, and the percent deviation. Both convergent variations of the best objective values and the scores over iterations for each DE strategy are presented from Figure 2 to Figure 5. The statistical result and the comparative performance in average are shown in Table 1 and Table 2.

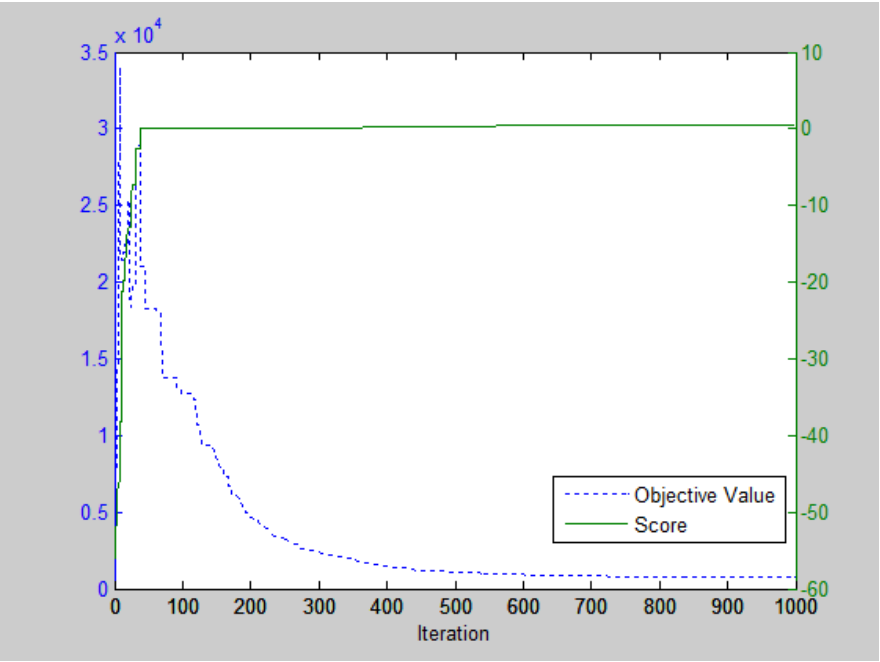

Figure 2. The convergent variations over iterations for improved DE strategy

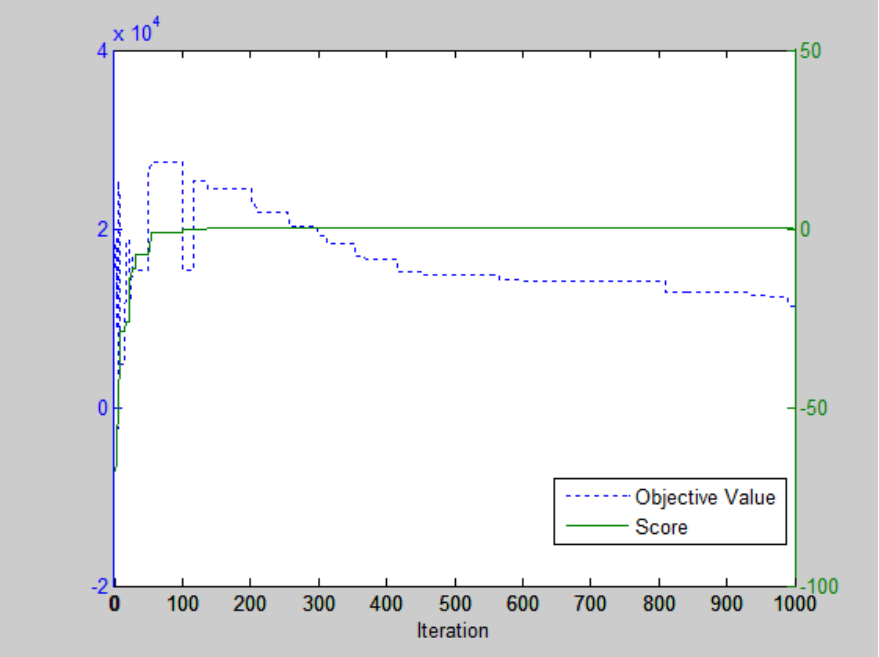

Figure 3. The convergent variations over iterations for DE/rand/1strategy

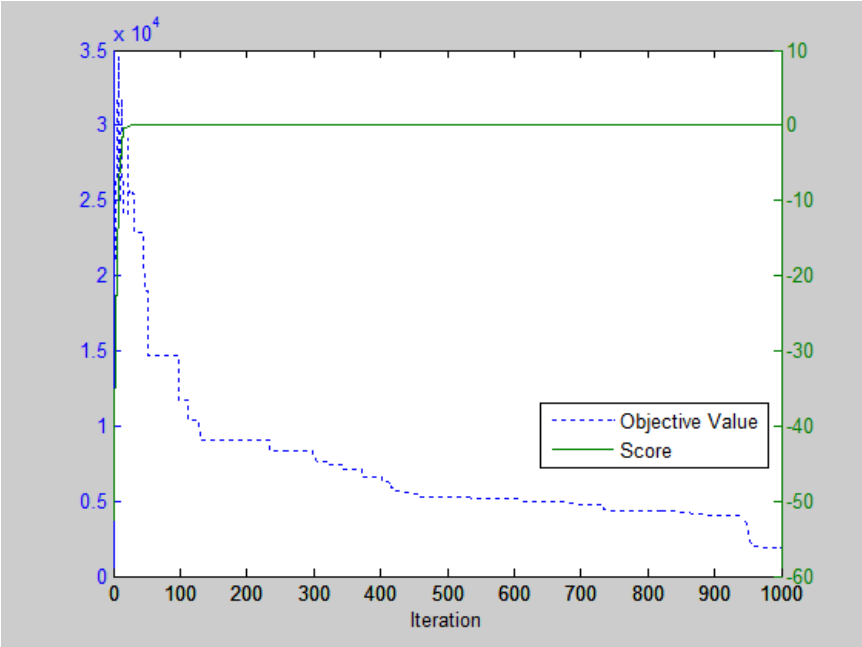

Figure 4. The convergent variations over iterations for DE/best/1 strategy

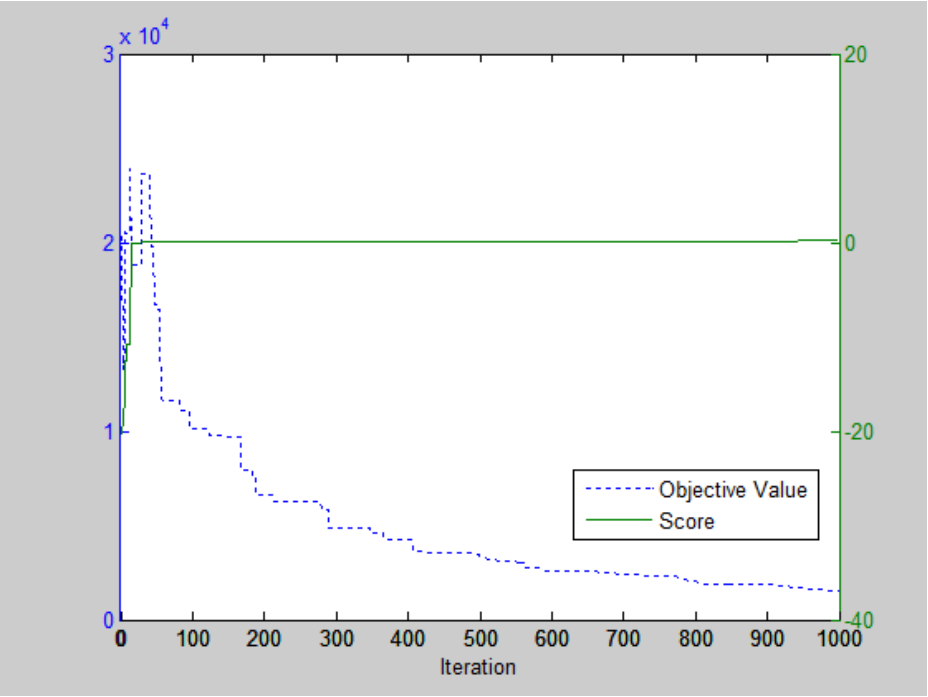

Figure 5. The convergent variations over iterations for DE/target-to-best/1 strategy

Table 1. The statistical results of the evolutionary experiments

	Test Instance	**1**	**2**	**3**	**4**	
	Periods	3	6	9	12	
	Worker Type	1	2	3	4	
	Num.of Para.	24	48	72	96	
	Num.of Constr.	12	42	90	156	
DE Strategy	**LB**	152	304	456	612	**Average**
Improved DE	Iter. (Stop)	1,162	2,457	3,291	4,523	2,858
	Best	158	315	473	635	395
	Mean	1,479	4,314	5,173	5,650	4,154
	Std.Dev	2,787	7,943	10,827	13,465	8,755
	Iter. (Score>0)	46	73	143	156	105
	%Dev	3.8%	3.6%	3.7%	3.8%	3.7%
DE/rand/1	Iter. (Stop)	5,000	5,000	5,000	5,000	5,000
	Best	2,488	32,594	81,402	102,693	54,794
	Mean	5,955	36,120	NaN	NaN	-
	Std.Dev	3,737	5,386	NaN	NaN	-
	Iter. (Score>0)	37	757	>5,000	>5,000	-
	%Dev	1,536.6%	10,621.8%	17,751.2%	16,679.9%	11,647.4%
DE/best/1	Iter. (Stop)	3,227	5,000	5,000	5,000	4,557
	Best	158	2,036	12,344	31,920	11,615
	Mean	1,517	10,248	28,301	51,237	22,826
	Std.Dev	2,793	10,041	13,861	16,133	10,707
	Iter. (Score>0)	17	96	223	555	223
	%Dev	3.9%	569.7%	2,607.0%	5,115.7%	2,074.1%
DE/target-to-best/1	Iter. (Stop)	3,045	5,000	5,000	5,000	4,511
	Best	157	1,534	9,752	24,527	8,993
	Mean	1,474	8,822	25,414	44,233	19,986
	Std.Dev	2,567	8,443	13,853	15,340	10,051
	Iter. (Score>0)	17	74	207	598	224
	%Dev	3.5%	404.8%	2,038.5%	3,907.7%	1,588.6%

Table 2. Comparative performance in average

DE Strategy	Iterations	Best	Mean	Std.Dev	Iter.(Score>0)	%Dev
Improved DE	100%	100%	100%	100%	100%	100%
DE/rand/1	175%	13,865%	-	-	-	313,523%
DE/best/1	159%	2,939%	549%	122%	213%	55,830%
DE/target-to-best/1	158%	2,275%	481%	115%	214%	42,762%

The experimental results from Figure 2 to Figure 5 and Table 1 to Table 2 show that the improved DE with the winner-based constraint handling mechanism can provide remarkable performance for optimizing the APP problem than the other 3 DE strategies, discussed as follows:

1) From Figure 2 to Figure 5, the convergent variations of the best objective values and the scores over iterations show that at the beginning of iterations, the best objective values for each DE strategy have a high degree of volatility. As their scores are smaller than 0, which means at least one constraint is still violated. Even many times they got small values enough but the solutions are still infeasible. At this stage, the winner-based constraint handling mechanism guided the DE algorithms substantially exploring better solutions in the infeasible regions while toward the feasible regions.
2) Soon after more iteration, the scores become greater than 0, which means the DE algorithms are evolving into the feasible regions. The winner-based constraint handling mechanism will handle exploiting better solutions without escaping from the feasible regions, leading the solutions stably and gradually evolving toward the global optimum. That is the important feature of the winner-based constraint handling mechanism.
3) It is apparent in Table 1 and Figure 2 to Figure 5 that all DE strategies can explore into the feasible regions in less than 5,000 iterations except the DE/rand/1. The reason for the DE/rand/1 may attribute to that the great randomness of it needs more efforts to explore in a highly constrained (90 and 156 for the test instance of 3 and 4), large multi-dimensional space (72 and 96 also). That may limit the effectiveness and ability of its exploration. Besides, the improved DE can quickly explore and evolve into the feasible regions better than others to a large degree. It is at about 105 iterations in average, better than the other DE strategies: DE/best/1 is at 223 iterations, DE/target-to-best/1 at 224, and DE/rand/1 is the worst. The performance of the improved DE for exploring feasible regions is times better than DE/best/1 (213%) and DE/target-to-best/1 (214%) in average (see Table 2), especially to the large-sized problems (see Table 1).
4) After evolving into the feasible regions, the improved DE can also quickly and accurately explore better solutions toward the global optimum in less than 5,000 iterations, at 2,858 iterations in average. It is also superior to the others: DE/best/1 at 4,557, DE/target-to-best/1 at 4,511, and DE/rand/1 is still the worst. The performance of the improved DE for exploring the global optimum is better than DE/rand/1 (175%), DE/best/1 (159%) and DE/target-to-best/1 (158%) in average. It is worth mentioning that the improved DE can explore inside the acceptance level of the corresponding lower bound in less than 5,000 iterations for all size of the problems, while 2 DE strategies only can do for the small-sized problems (see Table 1). We can say that the improved DE gains a good convergence to the optimum than others.
5) Apparently, as the result of the exponential probability distribution function with the scoring function for constraint handling incorporated into the DE algorithm, the other statistics in Table 2: Best, Mean, Std. Dev, and %Dev have also shown that it can gain greater performance than the other 3 DE strategies.

Therefore, we can conclude that the improved DE strategies with the winner-based constraint handling mechanism possesses particular quality in convergence, accuracy, and reliability for solving the APP problem than the most often used DE strategies of DE/rand/1, DE/best/1, and DE/target-to-best/1.

4. Conclusions

In this study, we first explore the importance of the APP problem, some significant researches and applications are also presented. To the optimization of the APP problem, we find that DE algorithms are less used applying to the problem in researches. We also discover that the most often used DE algorithms have some imperfections in optimizing the highly constrained APP problem. Therefore, in the succeeding context, an improved DE strategy and a constraint handling mechanism called the winner-based constraint handling mechanism are proposed to alleviate the deficiency of the DE algorithms for solving the problem. In the next, some test instances of the APP model described in Wang and Yeh (2014), from small-sized to large-sized, are randomly generated to evaluate the performance of the improved DE with the winner-based constraint handling mechanism. The improved DE strategy is experimented and compared with the most often used DE strategies, DE/rand/1, DE/best/1 and DE/target-to-best. The statistical result shows that improved DE strategy with the winner-based constraint handling mechanism can possess particular quality in convergence, accuracy, and reliability for solving the APP problem than the most often used DE strategies.

REFERENCES

[1] Al-E-Hashem, S. M. J. M., Aryanezhad, M. B., & Sadjadi, S. J. (2012). An efficient algorithm to solve a multi-objective robust aggregate production planning in an uncertain environment. International Journal of Advanced Manufacturing Technology, 58(5-8), 765-782.

[2] Chiou, J. P. (2009). A variable scaling hybrid differential evolution for solving large-scale power dispatch problems. Generation, Transmission & Distribution, IET, 3(2), 154-163. doi: 10.1049/iet-gtd:20080262

[3] Coello Coello, C. A. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Computer Methods in Applied Mechanics and Engineering, 191(11–12), 1245-1287. doi: http://dx.doi.org/10.1016/S0045-7825(01)00323-1

[4] Das, S., & Konar, A. (2009). Automatic image pixel clustering with an improved differential evolution. Applied Soft Computing, 9(1), 226-236. doi : http://dx.doi.org/10.1016/j.asoc.2007.12.008

[5] Das, S., & Suganthan, P. N. (2011). Differential Evolution: A Survey of the State-of-the-Art. Evolutionary Computation, IEEE Transactions on, 15(1), 4-31. doi: 10.1109/TEVC.2010.2059031

[6] Deep, K., & Dipti. (2008). A self-organizing migrating genetic algorithm for constrained optimization. Applied Mathematics and Computation, 198(1), 237-250. doi: http://dx.doi.org/10.1016/j.amc.2007.08.032

[7] Graves, S. C. (2002). Manufacturing planning and control. In P. M. Pardalos & M. G. C. Resende (Eds.), Handbook of applied optimization. New York 10016 USA: Oxford University Press.

[8] Iruthayarajan, M. W., & Baskar, S. (2009). Evolutionary algorithms based design of multivariable PID controller. Expert Syst. Appl., 36(5), 9159-9167. doi: 10.1016/j.eswa.2008.12.033

[9] Krohling, R. A., & dos Santos Coelho, L. (2006, 0-0 0). PSO-E: Particle Swarm with Exponential Distribution. Paper presented at the Evolutionary Computation, 2006. CEC 2006. IEEE Congress on.

[10] Maulik, U., & Saha, I. (2009). Modified differential evolution based fuzzy clustering for pixel classification in remote sensing imagery. Pattern Recognition, 42(9), 2135-2149. doi: http://dx.doi.org/10.1016/j.patcog.2009.01.011

[11] Mezura-Montes, E., & Coello Coello, C. A. (2003, 8-12 Dec. 2003). Adding a diversity mechanism to a simple evolution strategy to solve constrained optimization problems. Paper presented at the Evolutionary Computation, 2003. CEC '03. The 2003 Congress on.

[12] Paiva, R. P. O., & Morabito, R. (2009). An optimization model for the aggregate production planning of a Brazilian sugar and ethanol milling company. Annals of Operations Research, 169(1), 117-130.

[13] Ramezanian, R., Rahmani, D., & Barzinpour, F. (2012). An aggregate production planning model for two phase production systems: Solving with genetic algorithm and tabu search. Expert Systems with Applications, 39(1), 1256-1263.

[14] Runarsson, T. P., & Xin, Y. (2000). Stochastic ranking for constrained evolutionary optimization. Evolutionary Computation, IEEE Transactions on, 4(3), 284-294. doi: 10.1109/4235.873238

[15] Sillekens, T., Koberstein, A., & Suhl, L. (2011). Aggregate production planning in the automotive industry with special consideration of workforce flexibility. International Journal of Production Research, 49(17), 5055-5078.

[16] Stevenson, W. J. (2009). Operations Management (10th ed.): McGraw-Hill/Irwin.

[17] Storn, R., & Price, K. (1995). Differential Evolution- A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces.

[18] Tang, H., Xue, S., & Fan, C. (2008). Differential evolution strategy for structural system identification. Computers & Structures, 86(21–22), 2004-2012. doi: http://dx.doi.org/10.1016/j.compstruc.2008.05.001

[19] Varadarajan, M., & Swarup, K. S. (2008). Differential evolution approach for optimal reactive power dispatch. Applied Soft Computing, 8(4), 1549-1561. doi: http://dx.doi.org/10.1016/j.asoc.2007.12.002

[20] Wang, S.-C., & Yeh, M.-F. (2014). A modified particle swarm optimization for aggregate production planning. Expert Systems with Applications, 41(6), 3069-3077. doi: http://dx.doi.org/10.1016/j.eswa.2013.10.038

[21] Zhang, R. Q., Zhang, L. K., Xiao, Y. Y., & Kaku, I. (2012). The activity-based aggregate production planning with capacity expansion in manufacturing systems. Computers & Industrial Engineering, 62(2), 491-503

.