2019 12th Asian Control Conference (ASCC)
Kitakyusyu International Conference Center
Japan, June 9-12, 2019

A PSO-LP Cooperative Algorithm for Mixed Integer Nonlinear Programming

Ye YANG, Jinhou HAN, Chen CHEN, Jiarong XU, Zuwei LIAO, Xinggao LIU, Jinshui CHEN, Jiangang LU*

Abstract— **Mixed integer nonlinear programming (MINLP) problems widely exist in various aspects of an oil refinery, from crude oil supply chain, crude oil scheduling, production process decision-making, to optimal design of heat exchanger network or hydrogen network. Due to the NP-hard nature, many MINLP problems are difficult to solve efficiently and reliably. In this paper, a cooperative algorithm based on an improved particle swarm optimization algorithm (PSO) and linear programming algorithm (LP) is proposed to solve the complex MINLP problems. The PSO-LP cooperative algorithm includes an outer module running the improved PSO algorithm, and an inner module running LP algorithm, for example, the simplex algorithm. The outer module transforms the MINLP problem into a LP problem and passes it on to the inner module. The inner module feeds back the LP results to the outer module to update the MINLP problem. The above procedure iterates continuously to get the optimal solution of the MINLP problem. The proposed PSO-LP cooperative algorithm is verified using various MINLP problems. Results show that the proposed algorithm is computationally more accurate and more reliable than the existing ones.**

I. INTRODUCTION

Mixed integer nonlinear programming (MINLP) problems widely exist in various aspects of an oil refinery [1], from crude oil supply chain, crude oil scheduling, production process decision-making, to optimal design of heat exchanger network or hydrogen network.

There are two main algorithms for solving the MINLP problem: deterministic algorithm and heuristic algorithm. When the objective function exhibits strong nonlinearity and non-convex feasible region, deterministic algorithms [2] is hard to guarantee global optimal solutions or even impossible to solve. On the other hand, although the heuristic algorithm has the ability of global optimization in theory, when the feasible region is non-convex, the algorithm is easy to fall into the local optimal value which limits its application.

The general mixed-integer nonlinear integer programming problems (MINLP) as follows:

$$\begin{aligned} &minimize\ f(x,y) \\ &Subjected\ to\ h(x,y)=0; g(x,y)<0 \\ &x\in R^n, y\in I^m \end{aligned} \tag{1}$$

where x is defined as n-dimensional continuous variable, y is defined as m-dimensional integer variable.

In this paper, the PSO-LP cooperative algorithm is proposed to solve MINLP problems. This algorithm divides mixed integer nonlinear programming problem into outer layer problem and inner layer problem, the outer problem uses an improved particle swarm algorithm and the inner problem uses a linear programming algorithm. This algorithm combines the advantages of the heuristic algorithm and the deterministic algorithm that can reduce outer algorithm variables, as well as faster convergence speed and less computation time.

The rest of the paper is organized as follows. Section 2 introduces the standard PSO and its features. Section 3 gives a PSO-LP cooperative algorithm for solving MINLP. Numerical experiments and analysis are shown in Section 4. Finally, the conclusions are given in Section 5.

II. THE STANDARD PSO

Particle Swarm Optimization (PSO) is a global search algorithm based on group intelligence proposed by *Kennedy* and *Eberhart* [3] in 1995. Like other evolutionary algorithms [4], PSO is based on the concept of population and evolution and it can search for optimal solutions through competition and collaboration among individuals.

A. Principle of standard PSO algorithm

The PSO initializes a random particle swarm in the solution space, assuming that the position of *l*st particle in k dimension solution space is $x_i=(x_{i1},x_{i2},x_{i3},...,x_{ik})^T$, and its velocity is defined as $v_i=(v_{i1},v_{i2},v_{i3},...,v_{ik})^T$. According to the current optimal position of each particle $p_i=(p_{i1},p_{i2},p_{i3},...,p_{ik})^T$ and the current optimal position of all particles $p_g=(p_{g1},p_{g2},p_{g3},...,p_{gk})^T$, its velocity and position can be updated, formulas are as follows:

$$v_i^{t+1}=wv_i^t+c_1r_1^t(p_i^t-x_i^t)+c_2r_2^t(p_g^t-x_i^t) \tag{2}$$

*Research supported by the National Natural Science Foundation of China (No. 61590925, No. U1609212, No. U1509211), the National Key Research and Development Plan of China (No. 2017YFC0210403), and the Robotics Institute of Zhejiang University under Grant K11901.

Ye YANG, Han Jinhou, Chen CHEN, Jiarong XU, Xinggao LIU, Jinshui CHEN are with the State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, Zhejiang, China.

Zuwei LIAO is with the State Key Laboratory of Chemical Engineering, College of Chemical and Biological Engineering, Zhejiang University, Hangzhou 310027, Zhejiang, China.

Jiangang LU is with the State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, Zhejiang, China; he is also with the Robotics Institute of Zhejiang University, Yuyao Technology Innovation Center, Ningbo 315400, Zhejiang, China (corresponding author; e-mail: lujg@zju.edu.cn).

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (3)$$

Where r_1 and r_2 are random number between 0 and 1, c_1 and c_2 are learning factor which enables the particles to move closer to their own optimal position and the optimal position of the group, w is inertia weight.

B. Parameter selection of PSO

The parameters of PSO mainly include inertia weight, learning factor, maximum speed, etc.

1) Inertia Weight

The inertia weight w is used to control the influence of the previous velocity of the particle on the current velocity. The appropriate selection can make the particle balance the global search ability and the local search ability. The commonly used dynamic inertia weight is the linear decreasing weight strategy [5]:

$$w^m = w_{\max} - (w_{\max} - w_{\min}) * \frac{m}{L_{\max}} \quad (4)$$

Here, m is the current iterations, w_{max} is the initial weight and w^{min} is the inertia weight of getting maximum number of iterations, L_{max} is the maximum number of iterations.

2) Learning Factor

Reasonably setting the learning factor can reduce the probability of falling into local optimum and speed up the particle groups convergence. The learning factor c_1 can adjust the step size of the flight to its best position while the learning factor c_2 can adjust the step size of the flight to the best position of the group.

3) Maximum Speed

The particle velocity determines the particle's ability of exploration and development. IThe maximum speed can be adjusted by the inertia weight. When the maximum speed is small, the inertia weight is near 1, and when the maximum speed is large, the inertia weight is around 0.2.

C. Standard PSO Algorithm Flow

The standard PSO algorithm flow is:

1) Randomly initialize the particle swarm, randomly initialize the position and speed within the initialization range;

2) Calculate the fitness function value of each particle;

3) Compare the current fitness function value of each particle with the fitness function value of the best position experienced. If it is better, use it as the individual historical optimal value of the particle, and use this position to update the individual history optimal location

4) Compare the fitness function value of the current individual historical optimal position of each particle with the fitness function value of the best position experienced in the group. If it is better, use it as the global optimal position, and use this position to update the global optimal position.

5) Update the position and velocity of the particles according to formulas (2), (3), and (4).

6) Exit the algorithm when the termination condition is satisfied, otherwise return to step 2). The general termination condition is to reach a satisfactory fitness function value or reach the maximum number of iterations.

III. PARTICLE SWARM OPTIMIZATION ALGORITHM AND LINEAR PROGRAMMING ALGORITHM COOPERATIVE METHOD

Particles swarm optimization algorithm and linear programming algorithm cooperative method (PSO-LP). It divides mixed integer nonlinear programming problem into outer problem and inner layer. The outer problem uses an improved particle swarm algorithm and the inner problem uses a linear programming algorithm. Fig.1 shows the structure of the algorithm for the two-layer solution strategy.

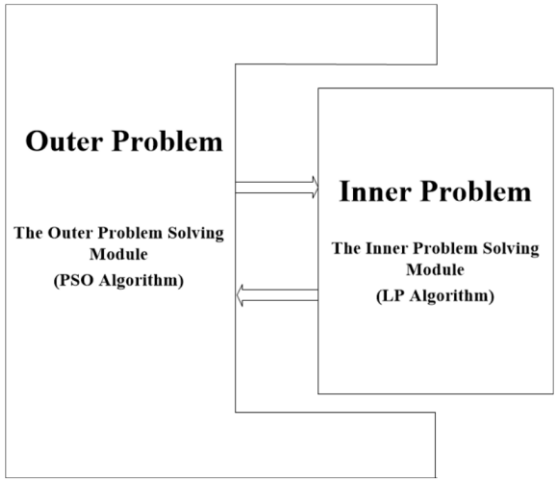

Figure 1. Double layer structure of PSO-LP cooperative algrithm

In order to better describe the core idea of the PSO-LP cooperative algorithm, we transfer general MINLP problems into a more specific form:

$$\begin{aligned} &\min f(x_1, x_2, x_3, y) \\ &s.t. \begin{cases} g(x_1, x_2, x_3, y) \le 0 \\ h(x_1, x_2, x_3, y) = 0 \end{cases} \\ &x_1 \in X_1, x_2 \in X_2, x_3 \in X_3, y \in Y \end{aligned} \quad (5)$$

where x_1 represents strong nonlinear continuous variable, $X_1 \subset R^{l_1}$, x_2 represents strong linear continuous variable, $X_2 \subset R^{l_2}$, x_3 represents weak nonlinear continuous variable, $X_3 \subset R^{l_3}$, here l_1, l_2, l_3 show each variable dimension which is an integer greater than or equal to 0. y is defined as integer variable, $Y \subset R^{l_4}$, l_4 is variable dimension which is an integer greater than 0. $g(x_1, x_2, x_3, y) \le 0$ and $h(x_1, x_2, x_3, y) = 0$ are inequality and equality constraint equations separately with the dimension number of p, q.

A. Variable Partition Strategy

Mixed integer nonlinear programming (MINLP) problem variables include integer variable y, linear variable x_2, nonlinear variable (strong nonlinear variable x_1, such as exponential logarithmic variable; weak nonlinear variable x_3, such as double product term, etc.) These variables are divided into L_1 and L_2, which correspond to PSO algorithm of the outer layer and LP algorithm of the inner layer respectively. The partitioning strategy is:

1) Divide the integer variable y, the strongly nonlinear continuous variable x_1 into L_1, and divide the linear continuous variable x_2 into L_2.

2) Divide the remaining weak nonlinear continuous variables x_3 into L_1 one by one, and observe the remaining variables of x_3. If the remaining x_3 variables are linear variables, divide them into L_2. Set variables of x_3 be divided into L_2 called x_{3L2} and number is l_{3L2}. Similarly, the variable assigned to L_1 is x_{3L1} and the number of variables is l_{3L1}, after grouping it shows as follows:

$$L_1 = \{x_1, x_{3L_1}, y\},\ \ L_2 = \{x_2, x_{3L_2}\}$$

Assuming that the number of L_1 variable is n_1, the number of L_2 variable is n_2, so $n_1=l_1+l_4+ l_{3L1}$, $n_2=l_2+ l_{3L2}$.

B. Improvements in the PSO of outer layer

- *Dual fitness functions*

In this paper, the dual fitness function and the tolerance of particles that do not satisfy the restrictions are introduced in improved PSO algorithm to solve constraint problem with massive equation. The dual fitness functions are $f_{1,l}^m$ and $f_{2,l}^m$, the system tolerance for default particles is g. $f_{1,l}^m$ is improved objective function and $f_{2,l}^m$ is default function of defined particle, they are defined as following formulas:

$$f_{1,l}^m = f(x_{1,l}, x_{2,l}, x_{3,l}, y_l,) \tag{6}$$

parameters here can be external environment parameters or objective function parameters or constraint parameters.

The robustness of the solution has been considered and uncertain set of $x_l = [x_{1,l}, x_{3L_1,l}]^T$ is determined:

$$D_{x_l}^m = \{x_l^m \mid x_l^m \in [\overline{x_l^m} - e_1, \overline{x_l^m} + e_2]\} \tag{9}$$

where x_l^m is a nominal value, e_1 and e_2 are parameters of uncertainty. When calculating the improved objective function of the m*th* iterative particle, randomly choosing parameters for C times from the uncertain set of x_l, then finding the mean as the improved objective function value for each particle:

$$f_{1,l}^m = \frac{1}{c}\sum_{j=1}^{c} f_{1,l}^m(x_{l,j}^m) \tag{10}$$

$x_{l,j}^m$ is defined as the random value of uncertain set $D_{x_l}^m$. In simple terms, for the parameter fluctuations existing in the industry, the robustness of the solution can help exclude some sharp points with poor robustness in the solution space.

- *Double mutation strategy*

$$f_{2,l}^m = \sum_{i=0}^{p_1}(k_{g,i} * \max(g(L_{1,l}), 0)) + \sum_{i=0}^{q_1}(k_{h,i} * |\, h(L_{1,l})\,|) \tag{7}$$

where m is the current iteration number, $g(L_1) \leqslant 0$ is the inequality constraint equation sets containing the L_1 variable, the number of equation set is p_1, $k_{g,i}$ is the default coefficient of inequality constraint equation sets. $h(L_1)=0$ means equality constraint equation sets with L_1 variables, the number of equations is q_1, and $k_{h,i}$ is the default coefficient of this kind of equation sets.

The pros and cons of particles are judged by comparing two fitness functions, and the rule of judgment is Deb criterion [6]. The Deb criterion gives the criterion for the superiority and inferiority of the particle under the constraint of the objective function. However, the judgment of the particle near the boundary is too absolute. For the case where the optimal solution is in the boundary, the Deb criterion can only make the particle converge from the feasible solution side to the optimal solution without using the spatial information on the side of the infeasible solution. Therefore, this paper improves the Deb criterion and gives the function representation of the system tolerance.

The improved criterion is as follows: when $f_{2,l_1}^m \leqslant g$ and $f_{2,l_2}^m \leqslant g$, the smaller particle of f_{1,l_1}^m and f_{1,l_2}^m is superior; otherwise, the smaller particle of f_{2,l_1}^m and f_{2,l_2}^m is better; where g is the tolerance of the system to the default particle, and the iteration formula is:

$$g^m = \left(\frac{2+\alpha}{3+\frac{s}{N}} - \frac{2+\alpha}{3} * \frac{m}{M}\right) * g^{m-1} + \delta \tag{8}$$

where s is the number of particles that satisfy $f_2^m > 0$, α is a set value and δ is a small positive number.

- *Linear robustness*

The robustness [7] of the optimization problem means that under the disturbance of certain parameters, the solution of the optimized problem can maintain certain characteristics. The

Outer improved PSO algorithm uses particle mutation probability function:

$$P = \mu + \mathrm{Re} * \sigma \tag{11}$$

μ and σ are disturbance rate adjustment parameters and *Re* is a parameter which implies whether there is a significant optimization of the particle swarm optimization. The optimal value of the *mth* generation particle swarm is defined as $O(m)$, and ε is a threshold variable that measures whether optimal value has a significant change and is a positive number, if$|O(m)- O(m\text{-}1)| \leqslant \varepsilon$, It means that the optimal value of the *mth* generation particle swarm is not significantly optimized, *Re* is set *Re*+1, otherwise, the optimal value does not change significantly, then *Re* equals 0.

Improved PSO algorithm compares the particle's own optimal position *pBest*[*l*] and the group optimal position *gBest* each time, and compares P with the random value Pr in (0,1). When *Re* increases continuously, the probability of P being greater than the random value Pr will increase; when $P \geqslant Pr$,

the current *gBest* is saved and the following particle mutation is performed:

Particle mutation is a double mutation strategy, which has strategy 1 and strategy 2 respectively. Re_{p1} is the number with no optimal optimization of the optimal value after the particle has undergone the mutation strategy, and M_{p1} is its threshold value, which is used for the selection of the mutation strategy. Re_{p2} and M_{p2} have the similar definition.

When $Re_{p1} \leqslant M_{p1}$, choosing the first mutation strategy:

$$V_l^m = V_l^m + V_l^m * \alpha_{1v} * \theta * \frac{M-m}{M} \tag{12}$$

$$L_{1,l}^m = L_{1,l}^m + L_{1,l}^m * \alpha_{1L} * \theta * \frac{M-m}{M} \tag{13}$$

V_l^m is the velocity of *lth* particle and $L_{1,l}^m$ is the position of *lth* particle, α_{1v} is mutation coefficient of velocity, α_{1L} is mutation coefficient of position, m is current iterations and M is total iterations, θ is a random number obeying the standard normal distribution $N(0,1)$.

When $Re_{p1} > M_{p1}$, choosing the second mutation strategy:

$$V_l^m = \alpha_{2v} * \theta * \frac{M-m}{M} \tag{14}$$

$$L_{1,l}^m = L_{1,gBest}^m + \alpha_{2L} * \theta * \frac{M-m}{M} \tag{15}$$

α_{2v} is mutation coefficient of velocity, α_{2L} is mutation coefficient of position, $L_{1,gBest}^m$ is the global optimal particle position, after mutation strategy, Re_{p1} is reset to 0 and $Re_{p1}=Re_{p1}+1$.

After the particle mutation, the linear programming algorithm of the inner layer is used to solve the parameter $L_{2,l}^m$, then recalculating the fitness value of the particle group. Updating *pBest*[*l*] and *gBest*, and replace the worst-fitted particles with current *gBest*, and reset *Re* to 0.

● *Termination condition setting*

The termination condition is set to two, one is that if iteration number is greater than the maximum number, the algorithm is terminated, and the other is Re_{p2} exceeds M_{p2}, and the algorithm terminates. This can ensure that the algorithm does not take too long and help end the algorithm early after the algorithm converging.

C. PSO-LP Cooperative Algorithm Step

Combined with the flow chart of PSO-LP cooperative algorithm shown in Fig2, the specific algorithm steps are as follows:

Step1: The variables in the MINLP problem are divided into two parts, L_1 and L_2, which respectively correspond to the outer simplified mixed integer nonlinear problem and the inner layer linear programming problem.

Step2: Initializing the outer improved PSO algorithm, the number of initial populations is N; the total number of iterations is M; the velocity boundary [V_{min}, V_{max}] and the position boundary [L_{1min}, L_{1max}] are determined; Particles uniformly generate the initial position $L_{1,l}^0$ in the space of m_1 dimension and randomly generate initial velocity V_l^0, where l = 1, ..., N.

Step3: After outer improved PSO solving operations, the problem is transformed into a linear programming problem. The inner layer problem solving module is used to solve the parameter L_2, and the solution of the parameter L_2 is $L_{2,l}^m$, where m is the current number of iterations, the initial solution is $L_{2,l}^0$.

Step4: Outer improved PSO update the particle's own optimal position $pBest^m[l]$ and the group optimal position $gBest^m$ according to the values of [$L_{1,l}^m$, $L_{2,l}^m$] and $f_{1,l}^m$, $f_{2,l}^m$ which are related to the robustness of solutions.

Step5: Calculate the particle mutation probability function value P, then comparing with a random value Pr in (0,1), if $P \geqslant Pr$, start particle mutation in the group, different mutation strategy is used to update the parameters according to the relationship between Re_{p1} and M_{p1}. After particle mutation, the the updated parameters $L_{2,l}^m$, $f_{1,l}^m$, $f_{2,l}^m$, $pBest^m[l]$, $gBest^m$ can be obtained. If $P \leqslant Pr$, go directly to *step6*.

Step6: It is judged whether the termination condition $m>M$ or $Re_{p2}>M_{p2}$ is satisfied, where M_{p2} is the threshold number of times of performing the second particle mutation strategy. If the termination conditions are not satisfied, let $m=m+1$. According to the particle's own optimal position *pBest*[*l*] and the group optimal position *gBest*, update the velocity V_l^m and the position $L_{1,l}^m$, and return to *step2*, if any termination condition is met, the algorithm ends.

IV. NUMERICAL EXPERIMENTS AND ANALYSIS

To evaluate the performance of the proposed algorithm, an experiment on a benchmark function has been conducted, the benchmark functions [8] as follows:

$$\min f(x_1, x_2, y) = -0.7y + 5(x_1 - 0.5)^2 + 0.8$$
$$s.t. \begin{cases} -\exp(x_1 - 0.2) - x_2 \leq 0 \\ x_2 + 1.1y \leq -1 \\ x_1 - 1.2y - 0.2 \leq 0 \\ 0.2 \leq x_1 \leq 1, -2.22554 \leq x_2 \leq -1, y \in \{0,1\} \end{cases} \tag{16}$$

It is obvious that strong nonlinear continuous variable is x_1; weak nonlinear continuous variable is x_2; integer variable is y. In line with PSO-LP cooperative algorithm, variables in the problem are grouped as follows:

$$L_1 = \{x_1, y\}, L_2 = \{x_2\}$$

The number of elements in the group L_1 is 2, number of elements in the group L_2 is 1. After dividing variables into two groups, several parameters in the algorithm are set, such as initial swarm number is N=200, total number of iterations is M=25; velocity boundary is [V_{min},V_{max}]=[-0.01,0.01],position boundary is [L_{1min},L_{1max}]=[{0,0,0,0},{10,10,1,1}]; particle

initial position is generated evenly as $L_{1,l}^{0}$,particle initial velocity is V_l^0 ,l=1,…,200; the tolerance of the system to the default particle g=0.1, α=0.05 ; μ,σ in particle mutation probability function are set 0.2 and 0.03 separately. Taking parameters into the model and the results can be obtained by MATLAB.

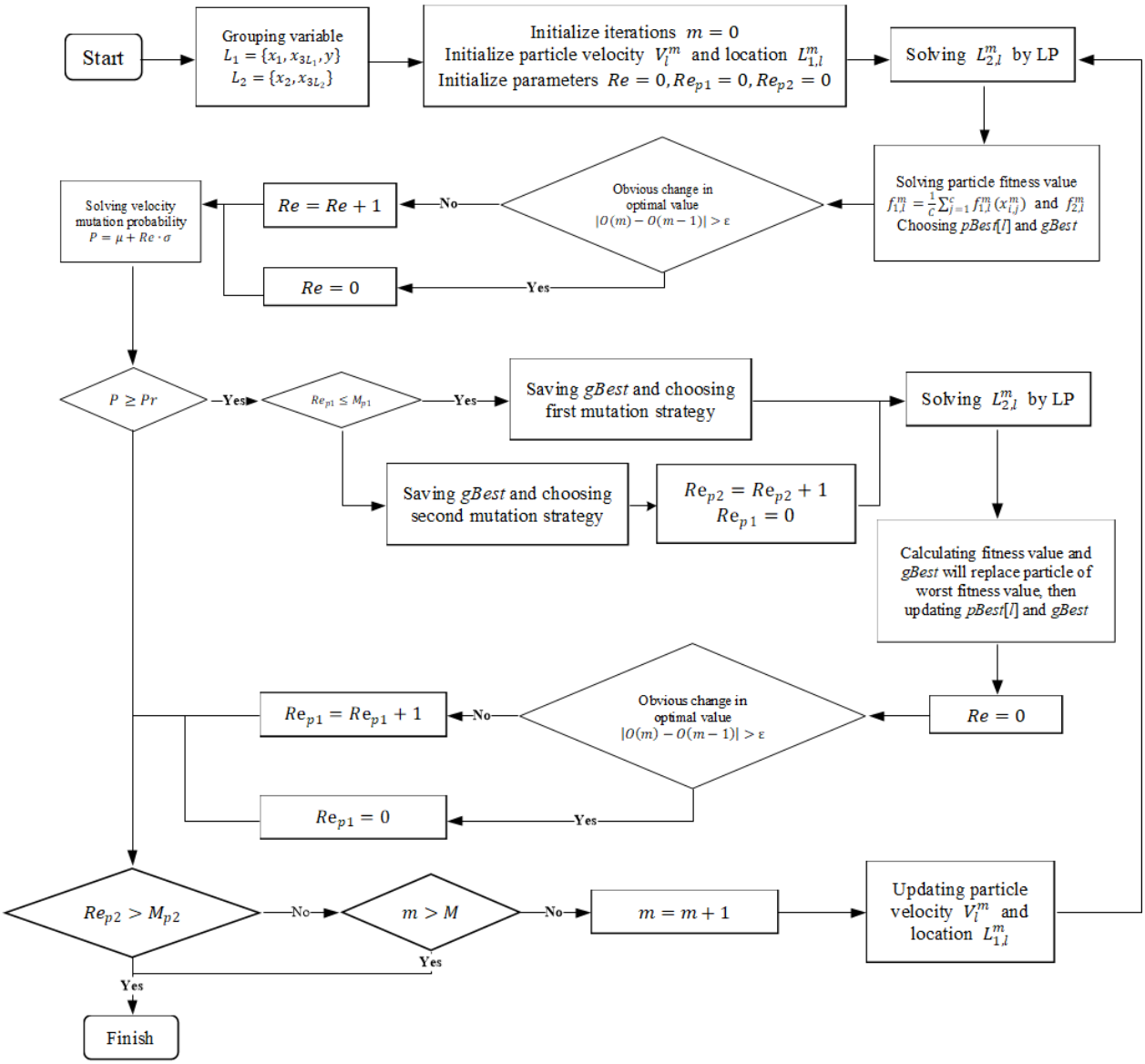

Figure 2. Flow chart of PSO-LP cooperative algrithm

The theoretical solution is [x_1,x_2,y]=[0.94194,-2.1,1],and the objective function value f=1.07654, while the optimal value given by PSO-LP is [v_1,v_2,v_3,v_4,v_5]=[0.9419,-2.1,1], the objective function value f=1.07654727. Comparing the two results, PSO-LP cooperative algorithm is basically the same as theoretical values.

According to the results, the average number of iterations of PSO-LP cooperative algorithm is 6386. The optimal values of each iteration have been collected and drawn in Fig.3. It is easy to show objective function value has already very close to the optimal value after the tenth iteration and tend to be stable. This indicates PSO-LP cooperative algorithm can quickly converge and find the optimal solution, which greatly improves the efficiency of the solution.

To verify the stability of the algorithm accuracy, the PSO-LP cooperative algorithm has been run independently 30 times, the results are shown in the Fig.4. Then the relative error between the running result and the theoretical value has been calculated, it shows the maximum relative error is 0.03%, which strongly illustrates the accuracy and stability of PSO-LP cooperative algorithm.

In order to prove that PSO-LP cooperative algorithm has advantages over existing algorithms. Two algorithms which achieve great results in solving MINLP problems have been selected to be compared: Differential evolution algorithm with chaotic local search (CLSDE) and Particle swarm optimization (PSO [10]). CLSDE [9] is an improved differential evolution algorithm which enhances the local search ability.

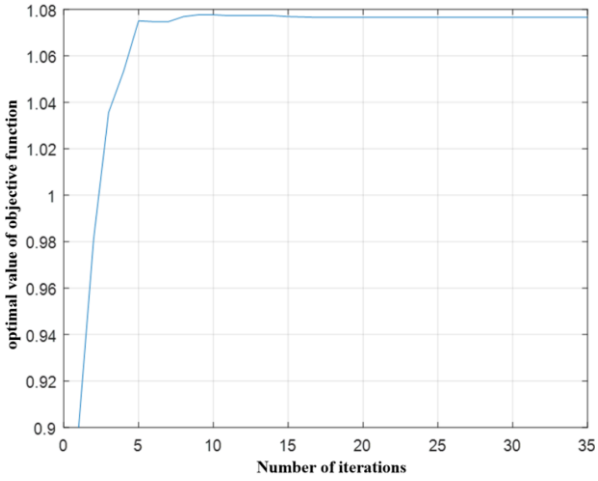

Figure 3. Objective function optimal value under different iterations

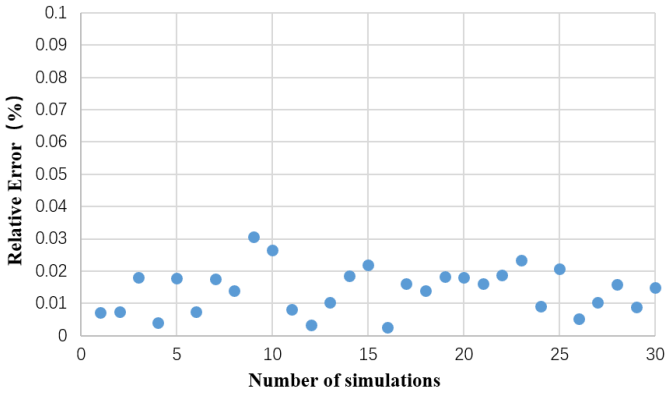

Figure 4. Relative error between actual result and theoretical value

All three algorithms (PSO-LP, CLSDE, PSO) mentioned above are used to solve the benchmark functions [8] at the same time. Compare the optimization result of POS-LP cooperative algorithm with other algorithms among the following four factors: the best result f_b ,the worst result f_w , the average value of all results $\overline{f}$ and the standard deviation σ_f . The results of the comparison are shown in the Table 1.

From the table it is obvious that POS-LP cooperative algorithm has a better performance in the listed algorithms, its standard deviation is the smallest among all algorithms; the optimal value and the average value are basically the closest to the theoretical optimal value. As an optimization algorithm of PSO algorithm, it successfully improves the accuracy of the algorithm on the basis of the original PSO.

TABLE I. OPTIMIZATION RESULTS OF DIFFERENT ALGORITHMS

	f_b	f_w	$\overline{f}$	σ_f
CLSDE[9]	1.0769	1.0787	1.0777	5.8905e-4
PSO[10]	1.076568	1.140703	1.079864	9.948e-3
PSO-LP	1.076545	1.076870	1.076693	7.54995e-5

*theoretical optimal f=1.07654

V. CONCLUSION

In this paper, an improved particle swarm optimization algorithm and linear programming algorithm (PSO-LP) cooperative algorithm is proposed for the MINLP problem. The outer layer of the algorithm uses the improved particle swarm algorithm (PSO) module, and the inner layer uses the linear programming algorithm (LP) module, the optimal solution to the MINLP problem is obtained through iterative updating of the inner and outer modules.

In improved PSO module, a dual fitness function is introduced to filter the particles that do not satisfy the constraint and select the optimal particles to enhance the algorithm's ability to search for the boundary. The particle mutation probability function and the double mutation strategy are introduced to prevent the particles from falling into local optimum in the early stage and enhance the local optimization ability in the later stage. Then the PSO-LP cooperative algorithm is applied to a standard test function for verification, and the computational complexity, correctness rate and results are analyzed and compared with other algorithms. The results show that the algorithm has better global search ability than other algorithms, proving the effectiveness and stability of the algorithm for solving the MINLP problem.

REFERENCES

[1] Karuppiah, Ramkumar, Kevin C. Furman, and Ignacio E. Grossmann. "Global optimization for scheduling refinery crude oil operations." *Computers & Chemical Engineering*, vol. 32, issue 11, pp. 2745-2766, 2008.

[2] Barhen, Jacob, Vladimir Protopopescu, and David Reister. "TRUST: A deterministic algorithm for global optimization." *Science*, vol. 276, issue 5315, pp. 1094-1097, 1997.

[3] Eberhart, Russell, and James Kennedy. "A new optimizer using particle swarm theory." In Conf. *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*. IEEE, 1995.

[4] Back, Thomas. *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford university press, 1996

[5] Shi, Yuhui, and Russell C. Eberhart. "Parameter selection in particle swarm optimization." *International conference on evolutionary programming*. Springer, Berlin, Heidelberg, 1998.

[6] Deb, Kalyanmoy. "An efficient constraint handling method for genetic algorithms." *Computer methods in applied mechanics and engineering*, vol. 186, issue 2-4, pp. 311-338, 2000.

[7] Coit, David W., and Alice E. Smith. "Reliability optimization of series-parallel systems using a genetic algorithm." *IEEE Transactions on reliability*, vol. 45, issue 2, pp. 254-260, 1996.

[8] Chanthasuwannasin, Manatsanan, Bundit Kottititum, and Thongchai Srinophakun. "A Mixed Coding Scheme of a Particle Swarm Optimization and a Hybrid Genetic Algorithm with Sequential Quadratic Programming for Mixed Integer Nonlinear Programming in Common Chemical Engineering Practice." *Chemical Engineering Communications*, vol. 204, issue 8, pp. 840-851, 2017.

[9] Tan, Yue, and G. Z. Tan. "Differential Evolution Algorithm with Chaotic-local-search Strategy [J]." *Journal of Chongqing Institute of Technology (Natural Science)*, vol. 5, issue 12, 2009.

[10] Kennedy, James. "Particle swarm optimization." *Encyclopedia of machine learning*. Springer, Boston, MA, 2011, pp. 760-766.