Contents lists available at YXpublications

International Journal of Applied Mathematics in Control Engineering

Journal homepage: http://www.ijamce.com

Review of Methods for Solving Large Scale Mixed Integer Programming

Yue Chen[a], Liangliang Sun[a,*], Baolong Yuan[a], Xiaoxi Tian[a], Qinghua Shi[a]

[a] *Faculty of Information and Control Engineering, Shenyang Jianzhu University, Shenyang, 110168 China*

ARTICLE INFO

Article history:
Received 15 March 2021
Accepted 18 May 2021
Available online 21 May 2021

Keywords:
Large scale mixed integer programming
Lagrangian relaxation
Heuristic algorithm
Approximation algorithm

ABSTRACT

Based on the basic mathematical features of mixed-integer programming, the problem of slow convergence and low solution quality is found in most of the algorithms based on the basic mathematical features of mixed-integer programming and the practical application of the existing solution theory. On this basis, we further analyze the adaptability of existing Lagrangian relaxation algorithms to large-scale mixed-integer programming problems, and evaluate the advantages and disadvantages of different iterative strategies based on Lagrangian relaxation framework in terms of the stability, convergence and solution quality of the algorithms, and confirm the superiority of Lagrangian relaxation algorithms for solving large-scale mixed-integer programming.

1. Introduction

A mixed-integer programming problem is an optimization problem in which some of the decision variables must be integers, i.e., an optimization problem that optimizes a multivariate function with integer variables subject to a set of equations or inequalities. According to its model and characteristics, it can usually be divided into linear mixed-integer programming and nonlinear mixed-integer programming. The difficulty of solving nonlinear mixed-integer programming problems is much greater than that of linear mixed-integer programming problems, so this paper focuses on the solution of large-scale linear mixed-integer programming problems. Among them, the linear mixed integer programming problems have the following forms.

$$\min c^T x + h^T y \qquad (1)$$
$$\text{S.t. } Ax + Gy \le b$$
$$x \in \mathbb{Z}_+^m, y \in \mathbb{R}_+^n \ [1]$$

The mixed integer programming problem can solve the optimization problem of discrete variables, and it has a wide range of applications--the model has penetrated into most research fields such as engineering technology research, social sciences, and economics research. Especially for large-scale combination optimization problems such as production scheduling problems, electromechanical combination optimization and transportation network optimization problems, the mixed integer programming model can commendably describe the characteristics of such questions, and the calculation technology is simple and has strong applicability. In ordinary daily life, the mixed integer programming model can also solve the problem of overall arrangement of daily work and improving work efficiency. Therefore, the study of large-scale mixed integer programming models has pretty practical significance. However, with the in-depth research in various fields of science and technology, the research content is more specific and detailed, and the decision variables and constraints of the mixed integer programming model are also more complicated and calculation difficult. This requires some special methods to solve such problems. Next, this article will introduce several commonly used mixed integer programming optimization algorithms and their improved algorithms.[2]

2. Introduction to existing mixed integer programming optimization problem methods

Since Cook [3] introduced the concepts of NP and complete NP in 1971, the mixed integer programming problem has been extensively studied, and it has three main phases as follows.

2.1 Precise algorithm

The exact algorithm has been extensively studied in the past decades, and its core idea is to solve mixed-integer programming optimization problems using fast exponential-time algorithms, which are essentially exponential-level violent search algorithms. [3] This algorithm is usually applicable to small-scale mixed integer solution problems at around 10 nodes, and the more complex the

* Corresponding author.
E-mail addresses: sunliangliang@sjzu.edu.cn (L. Sun)

problem, the longer the running time of the exact algorithm, and so far it still has not broken the upper bound of 2^n on running time [3]. For the NP-

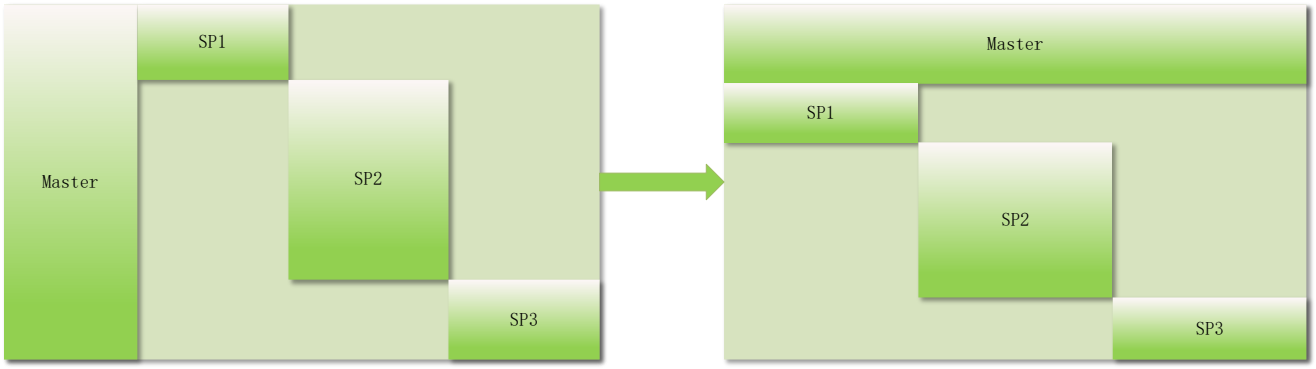

Fig. 1 The relationship between bender decomposition and Dantzig-Wolfe decomposition

-hard problem, Impagliazzo et al [5] pointed out that sub-exponential algorithms do not exist. Although the solution scale of the exact algorithm is small, the final solution it seeks must be optimal, and the common exact algorithms are the cut-plane method and the branch-and-bound method.

（1）Benders decomposition

The Benders decomposition was proposed by Jacques F. Benders in 1962 [4] and is commonly used to solve extreme value problems containing integer and continuous variables. The basic concept is to fix the values of the integer variables (complex variables) and simplify the optimization problem to obtain a general linear programming problem (generally called subproblem). The subproblem takes the complex variables as parameters and uses the cut-plane method to find the extreme values of the main problem and the set of values of the covariates of the general linear programming problem (usually called the subproblem) containing feasible solutions. That is, only a small number of constraints are considered at the beginning for the master problem, and the natural cut-plane clusters containing the above-mentioned set of characteristics are derived by solving the dual problem [5] until the optimal solution is approximated. Moreover, this part of the constraints may have significant "diagonal block structure", as shown in Figure 1.

（2）Branch delimitation method

The idea of branch-and-bound method comes from binary trees, and its kernel lies in "branching" and "bounding". The "branching" is similar to the framework of "divide and conquer": the algorithm takes the necessary constraints and divides the integer mixed programming problem into the original problem and the linear programming problem without integer constraints, and adds additional conditions to the non-integer solutions of the linear programming problem without integer constraints. In order to form two branches (i.e. two subproblems), additional conditions are added to the adjacent integer solutions of non integer solutions of linear programming problems without integer constraints. The feasible solution space of the two branches can contain all the feasible solutions of the original objective function.[9] When a linear programming problem without integer constraints does not meet the integer constraints of the original problem, then its optimum must be the boundary value of the optimum of the original problem, while any feasible solution of the objective function is its boundary value. [6]

（3） Dantzig-Wolfe decomposition (DW decomposition)

The essence of DW decomposition is the column generation algorithm, and the DW decomposition is obtained by flipping the above Benders decomposition. The same as Benders decomposition is that DW decomposition puts complex constraints in the main problem and simple constraints in the subproblem constraints, but unlike Benders, DW decomposition reduces the constraints and increases the variables of the main problem by using the requested poles as the base variables of the main problem.

2.2 Approximation algorithm

Approximation algorithms are the approximate optimal summary, which require some techniques such as greedy strategy, restriction, division, cut-off, and relaxation to solve the problem according to the specific problem. This type of algorithm has a limit on the size of the solution and the results obtained are not optimal. [7]

（1）Forbidden search algorithm

The forbidden search algorithm is a stochastic search algorithm [8]. The forbidden search algorithm constructs a taboo table during the search process to store the found locally optimal solutions, using the pardon criterion that the forbidden solutions with promising excellent results are pardoned, which avoids falling into local optimum solutions and gains more search space [9].

（2）Greedy algorithm

The greedy algorithm is a type of optimal search algorithm and a local search algorithm. Its core is to use the method of constructing the optimal solution step by step, that is, to make an unchangeable and optimal decision to a certain extent at each stage. The criteria that determine the assignment and selection of variables at each step are called greedy criteria, also known as greedy factors [10] . The solutions obtained by the greedy algorithm are usually locally optimal, and the global optimal solution can only be obtained by repeated search of multiple initial solutions [11].

2.3 Heuristic algorithm

Different from the above-mentioned algorithms, heuristic algorithm is a parallel processing algorithm designed based upon the phenomena and properties of biology, physics and artificial intelligence, such as simulated annealing algorithm, ant colony algorithm, artificial neural network algorithm, etc. Heuristic algorithms have been widely used in combinatorial optimization problems because they do not require problem specificity, global optimality, and generality [12].

（1）Simulated annealing algorithm

Because the annealing process of solids is similar to the combinatorial optimization process, research scholars introduced the principle of annealing of solids into the study of combinatorial optimization [13][14]. Similar to the annealing principle of solids, the simulated annealing algorithm first sets a large initial solution and

the initial value of the control quantity (the higher temperature is taken as the initial temperature); iteratively calculates, compares the gap of the objective function, gradually makes the value of the control parameter smaller (in the process of decreasing the temperature parameter), and uses the sudden jump property of probability to find the best answers of the original problem in the solution space. The simulated annealing algorithm is a stochastic optimization-seeking algorithm with a serial structure, which can effectively avoid the problem of falling into local minima and converging to local optima [15].

（2）Ant colony algorithm

A population-based heuristic bionic algorithm, the ant colony algorithm, was proposed by Dorigo Macro et al. by simulating the behavior of ant colonies in the biological world [16]. Information is transferred and exchanged between individual ants through pheromones, and ants prefer to move in the direction of high concentration of that substance, i.e., the more ants pass on a certain path, the higher the probability that the path will be chosen by the later. Similar to the movement of ants, the ant colony algorithm continuously adjusts the structure of candidate solutions based on known information in the adaptation phase; in the collaboration phase candidate solutions exchange information and use the evolutionary process of the population composed of candidate solutions to produce the optimal solution [17]. This algorithm is highly applicable to complex optimization problems such as TSP and QAP, but it also suffers from slow convergence, long search time, and limitation by local optimal solutions [16].

（3）Artificial neural network algorithm

Artificial neural network algorithm is an algorithm for parallel processing by simulating the mathematical model of a neural network with multiple neurons that can be connected with tuned connection weights [16]. The artificial neural network algorithm is essentially a multi-level processing system: the input data enter from the input layer, forward and backward propagation is performed in the implicit layer, and the results are output from the output layer [19]. It has massive parallel processing, distributed information storage [18] good adaptability and strong learning ability for large-scale mixed integer programming optimization problems. The convergence speed and the quality of the optimal solution of the artificial neural network algorithm are directly linked to the selection of the algorithm parameters, while a better formula for parameter selection has not been given yet.

The exact algorithms can be classified into three types: Benders decomposition, branch-and-bound, and Dantzig-Wolfe decomposition. From a global perspective, the Benders decomposition algorithm has a small global gap and is easy to obtain the optimal solution. However, its variables must be clearly partitioned; the optimal solution is tested until an optimal solution is obtained that satisfies all constraints. For branching delimited problems, it has good adaptability to pure integer problems and mixed integer programming problems. However, the time and space complexity of the algorithm is 2^n at a time. for the Dantzig-Wolfe decomposition, it is suitable for solving some mixed integer programming problems with a specific structure. However, there is little related theory in China.

Among the approximation algorithms, the forbidden search algorithm has a strong local development ability and faster convergence speed, and the greedy algorithm has a high quality of local optimal solutions. However, the search results of the forbidden search algorithm completely depend on the initial solution and domain mapping relationship, and the global development ability is weak. The greedy algorithm also has some problems, such as weak global exploitation ability, and the results may not be optimal solutions.

For heuristic algorithms, we can discuss the advantages and disadvantages of the following three algorithms. The simulated annealing algorithm has high efficiency, good parallelism, good robustness and other good performance. However, it is highly dependent on the initial value and easy to sink into local optimal solutions. The ant colony algorithm has effective robust and strong global search capability, but its population diversity and convergence speed are contradictory, and it is easy to be trapped in the local optimal solution and difficult to exit from the local optimal solution. For artificial neural network algorithm, it has strong fault tolerance, strong self-learning and self-adaptive ability. However, its slow convergence speed and strong dependence on data cannot explain the feasibility of the optimal solution.

Based on the above analysis, this paper summarizes the solution methods of mixed integer programming problems. The comparative analysis of various algorithms shows that the current solution methods of large-scale mixed integer programming are characterized by the difficulty of getting out of local optimum solutions, strong data dependence, incompatible convergence speed and solution quality. Therefore, how to avoid local optima and improve the quality and operational efficiency of the solution becomes an urgent problem to be solved.

3. A Rasch relaxation method for solving large-scale mixed-integer programming problems

3.1 Model optimization under the Rasch framework

In 1970, Held.M was the first to successfully apply the Lagrangian relaxation algorithm to compute the postman problem and the minimal tree problem. In 1974, Geoffrion et al. named this Lagrangian relaxation-based algorithm Lagrangian Relaxation. 1974, Held.M described the subgradient method for Lagrangian multipliers which provides a good solution for the calculation of the Lagrangian multiplier [20].

Lagrangian relaxation algorithms, agent relaxation algorithms and pairwise relaxation algorithms are often used to provide a suitable theoretical lower bound (upper bound) for solving the minimization (maximization) problem: in general, Lagrangian relaxation algorithms are used to solve a suitable theoretical lower bound for solving the minimization problem; to solve the maximization problem, the maximization problem is first transformed into a minimization problem and then solved by Lagrangian relaxation algorithms. The essence of the Lagrangian relaxation algorithm is to reduce the number of constraints by changing the objective function but not changing its linearity, and the objective function is always linear during the whole process. In other words, the Lagrangian relaxation algorithm treats a large-scale mixed-integer programming problem as a series of simple subproblems combined by boundary constraints. Using this feature, the Lagrangian relaxation algorithm linearly relaxes away (removes some of the hard constraints) from some NP-hard combinatorial optimization problems by introducing a multiplier (i.e., Lagrangian multiplier) and absorbing it into the original problem's objective function to simplify the problem. At this time, the selection of Lagrange multipliers, the iterative approach used and the correction shortening used for different constraints will

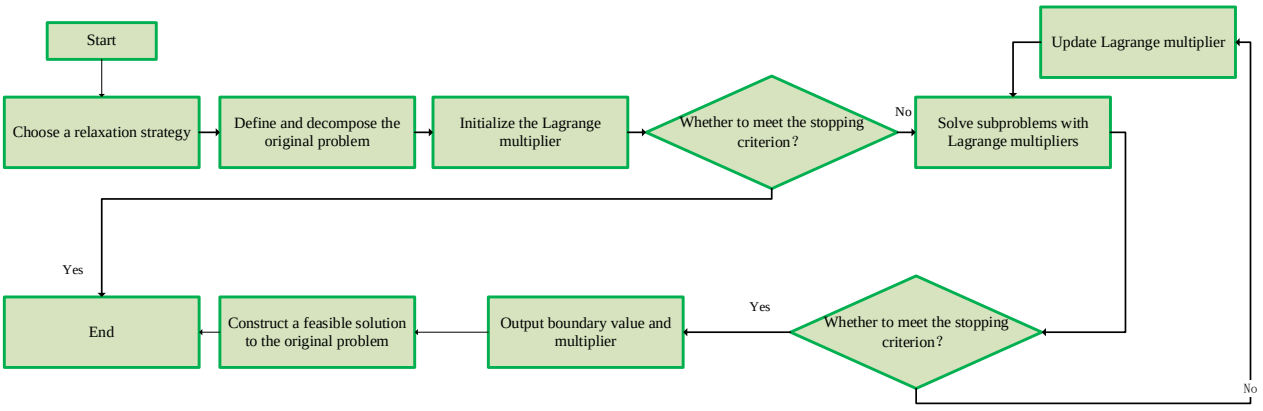

Fig. 2 Basic flow chart of Lagrange relaxation algorithm

not only reflect the scale of violation of the current solution to the hard constraints, but also affect the quality of the solution. In turn, some effective algorithms such as sub-gradient optimization algorithm should be used to iteratively update the Lagrange multipliers continuously to accelerate the convergence speed, find the optimal solution or near-optimal solution, and improve the solution quality. (The basic flow chart of Lagrangian relaxation algorithm is shown in Figure 2.)

The Lagrangian relaxation algorithm has the following definitions and theorems.

Definition: If the problem $Z_R = \min_{x \in S_R} Z_R(x)$ is a relaxation of an integer programming, then it satisfies the following two properties.

$S \subseteq S_R, S_R$ denote a set of solutions; .

$c^T x \geq Z_R(x), \forall x \in S$, $Z_R(x)$ are real functions; [21]

（1）Choose a relaxation strategy to decompose the problem and solve it

Depending on the characteristics of the combinatorial optimization problem being solved, the complex constraints are defined differently. Therefore, it is necessary to choose the appropriate constraints for relaxation according to the specific conditions of the objective function. After the relaxation of the complex constraints, the feasible solution range will naturally be expanded while the original objective function is reduced in computational difficulty. Based on this, we should reasonably define the original objective function Lagrangian relaxation problem so that the decomposed subproblem is easier to solve by reducing the constraints of the original objective function while maintaining linearity.

After defining the original objective function Lagrangian relaxation problem, a suitable iterative algorithm is chosen to solve the optimal solution. In this process, it is worth noting that: since the iterative process is an infinite approximation to the optimal solution, we may only get the approximate optimal solution; the optimal solution of the relaxed subproblem may be contrary to the constraint that is absorbed into the objective function.

（2）Definition of the dual problem and the Lagrange multiplier update solution

As mentioned above: after the relaxation of the original objective function, the space of feasible solutions will be expanded; the optimal solution of the subproblem is not necessarily the optimal solution of the original combinatorial optimization problem; the optimal of the subproblem is not feasible. In order to avoid these potential problems and to better approximate the optimal solution of the original combinatorial optimization problem, the Lagrangian dual solution is usually used instead of the original solution method.

First, given a combinatorial optimization problem, i.e., the original problem (PP),assume that its problem description takes the form of

$$\text{(PP)} \quad Z_{PP} = \min c^T x \quad \text{S.t.} Ax \leq b \quad x \in z_+^n \qquad (2)$$

The corresponding relaxation problem (LP) is.

$$\text{(LP)} \quad Z_{LP} = \min c^T x + \lambda(Ax - b) \quad x \in z_+^n, \lambda > 0 \qquad (3)$$

The dyadic problem (DP) is.

$$\text{(DP)} \quad Z_{DP} = \max\{\min c^T x + \lambda(Ax - b)\} \quad x \in z_+^n, \lambda > 0 \qquad (4)$$

The objective function expression of the dual problem has the following relationship with the objective function of the original problem:

$$\text{PP:} \quad Z_{PP} = \min c^T x \qquad (5)$$

$$\text{DP:} \quad Z_{DP} = \max\{\min c^T x + \lambda(Ax - b)\} \qquad (6)$$

From the above comparison of the objective function value $\max(\min c^T x)$ and $\min\{\max[\min c^T x + \lambda(b - Ax)]\}$,it perceives that the objective function value of PP is the maximum of the DP objective function, which is a "maximum-minimal" problem; the objective function value of DP is the minimum value of the PP objective function, which is "minimum-maximum" problem. (As

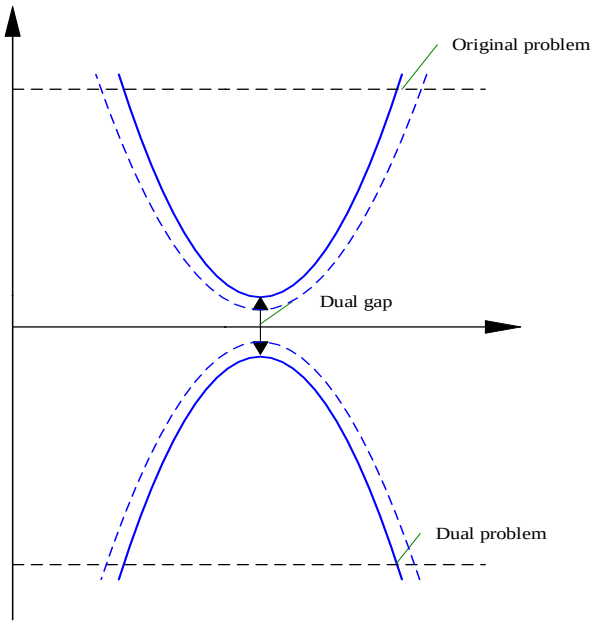

Fig. 3 Schematic diagram of the pairwise gap

shown in Figure 3 below)

At the same time, through the above relations, the following theorems can also be obtained:

Weak duality theorem: as described in the following equation, the solution of the subproblem is smaller than that of the original problem but larger than that of the dual problem. The difference between the value of the original problem and the value of the dual problem is called the absolute pairwise gap of the feasible solution.

$$Z_{DP} \leqq Z_{LP} \leqq Z_{PP} \tag{10}$$

Pairwise optimality theorem: if the absolute pairwise gap is 0 and the point where the maximum value of the objective function coincides with the minimum value of the pairwise problem, then the solution of the subproblem at this point is the optimal solution of the original-pairwise problem. (As shown in Figure 4 below)

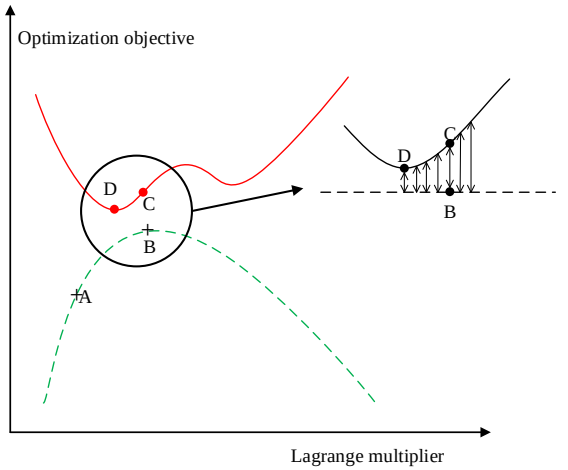

Fig. 4 Analysis of dual problem

3.2 Iterative optimization algorithm

Transforming the constraints into new general function residual conditions by introducing Lagrange multipliers to reduce the constraints of the original problem is the basic idea of the Lagrange multiplier method. [22][23] The accuracy of approximating the best answer depends on the precision of the multiplier selection, iterations, the selection of the optimal solution and feasible solutions to optimization problems. How to find the update direction of the Lagrange multiplier in a short time, simultaneously reduce iterations to gain on the optimal boundary has become an important issue that affects the efficiency of the Lagrange relaxation algorithm.

Later, some researchers combined the penalty function outpoint method and the Lagrange multiplier method, i.e., the Lagrange multiplier is introduced under the construction of the penalty function to construct the augmented Lagrange function. The augmented Lagrangian algorithm approximates the constrained optimal point of the original objective function by adjusting the penalty factor and the Lagrangian multiplier to derive an approximate optimal solution in terms of the convergence criterion [24].

（1）Subgradient descent method

The subgradient algorithm was first proposed in the Soviet Union in the 1960s [25]. The subgradient algorithm is essentially an improvement of the gradient descent method, which can handle a larger range of non-derivative convex functions efficiently compared to the gradient descent method. Define the subgradient as follows.

$$\partial f = \{g \mid f(x) \geq f(x_0) + g^T(x - x_0), \forall x \in domf, f: R^n \to R\} \tag{7}$$

Similar to the definition of gradient, if $f: R^n \to R$ is a convex set defined in a Euclidean space $\mathbb{R}^n$, then the vector v in that space is called the subgradient of the function $f(x)$ at point x_0. The set of all subgradients is called the subdifferential $\partial f(x_0)$. Since the subgradient obtains the most equivalent for any convex or nonconvex function $f(x)$ at the point $f(x)$:

$$f(x^*) = \min_x f(x) \Rightarrow 0 \in \partial f(x^*) \tag{8}$$

Namely , x^* is the optimal one when and only when 0 is an element of the subgradient set of $f(x^*)$. Therefore, the subgradient algorithm can be used to solve the large-scale mixed integer programming problem well. The basic mathematical thought of the subgradient algorithm is to replace the gradient with the subgradient, and use the negative direction of a subgradient as the search direction in each iteration to select the appropriate step size and compute the next iteration point. Suppose the number of iterations is k and the step length is s^k. , the subgradient method is used to update the multiplier.

$$\lambda^{k+1} = \lambda^k + s^k g^k \tag{9}$$

where g^k determined by the original problem and multipliers:

$$g^k = g(\lambda^k) = Ax(\lambda^k) - b = \sum_{i=1}^{n} a_i x_i(\lambda^k) - b \tag{10}$$

By theorem : $0 \leq L^* - L^k \leq (\lambda^* - \lambda^k) g(\lambda^k)$, it can be deduced that the step size satisfies.

$$0 \leq L^* - L^k \leq \frac{2(\lambda^* - \lambda^k)}{\left\|g^k\right\|^2} \tag{11}$$

From this, the domain of feasible solutions of the original function can be found as

$$x_i(\lambda^k) = \arg \min_{x_i^n \leq x_i \leq x_i^m} [J_i(x_i) + (\lambda^k)^T (a_i x_i)] \tag{12}$$

However, the sub-gradient algorithm does not have an explicit method for step selection (step size and search direction) and does not guarantee that the cost function is monotonically decreasing at each change. Moreover, due to the Markov property of the subgradient algorithm, [26][27] the current subgradient has no memory of historical gradients. Therefore, it is highly susceptible to oscillations and slow convergence when using sub-gradient to find the best.

From the perspective of step selection criterion, there are several commonly used step criteria such as constant-value step, decreasing step, divergence step, Polyak step, and dynamic step. The constant step criterion is simple in form, but the function value converges to a constant only at the iteration point. The multipliers under the divergent and decreasing step criterion converge to infinity while iteration process goes on , i.e., the subgradient iteration algorithms based on these two step convergence criteria are not linearly convergent and do not converge fast. The subgradient algorithm based on the Polyak step size criterion is linear, but it is highly dependent on the optimal solution of the objective function, which is not conducive to its generalized use. Therefore, Goffin et al [28] proposed a dynamic step criterion based on the Polyak step criterion. Compared with the Polyak step criterion, the improvement of the dynamic step criterion is that it obtains better convergence by adjusting the approximation of the optimal value. In addition, in order to assure the convergence of the subgradient algorithm, many scholars have also proposed the agent subgradient algorithm, neighborhood conjugate subgradient algorithm, incremental

subgradient algorithm, fuzzy subgradient algorithm, random subgradient algorithm, etc. from the perspective of step selection criterion and using historical subgradient information.

Instead of updating the multipliers after a complete iteration cycle, the proxy subgradient algorithm updates the multipliers after each subproblem is solved. In general, the algorithm avoids oscillations by performing multiplier updates in the gradient direction of the agent subgradient function, which allows the algorithm to add new information as soon as it encounters it. However, the agent sub-gradient algorithm uses a more frequent multiplier update strategy to fasten the convergence of the algorithm, which further increases the original computational complexity. In addition, if the starting multiplier is not chosen properly or the iteration step is too large, it is highly likely to cause the problem of lower bound violation for the agent pairwise [29].

Zhao et al [30] introduced the concept of fuzziness into the sub-gradient algorithm and proposed an algorithm that uses fuzziness to determine the gradient weights, which is usually called the fuzzy sub-gradient algorithm. In the fuzzy sub-gradient algorithm, it is usually considered that the smaller the difference between a certain historical sub-gradient direction and the currently obtained sub-gradient direction, the greater the weight in that historical sub-gradient direction is considered. Based on the above theory, the fuzzy sub-gradient algorithm extracts the weight coefficients of historical sub-gradients by using the affiliation function in the iteration process, and the result of linear weighting of these weight coefficients is used as the new iteration direction, which can make full use of all the historical gradient information and can better suppress the oscillation phenomenon. However, in practical application, the fuzzy sub-gradient algorithm has a certain chance to make the iteration step size decrease, and the short-term convergence effect of individual algorithms may be poor [31].

Along the sub gradient direction of the component function, the intermediate variables are used to adjust the sub gradient to ensure that the variables of each iteration are incrementally carried out for several times, that is, the incremental sub gradient iteration method is called incremental sub gradient method [32]. Since the iterative process of this algorithm involves sub-iterations of component functions, the difference in the order of component function iterations significantly affects the convergence rate of this algorithm. In order to find an optimal iteration order, scholars [33][34] introduced "random" into the subgradient algorithm and proposed the stochastic subgradient algorithm. It is proved by literature [35][36] that the convergence performance of randomly selected iteration order is better than that of the algorithm with fixed order of iteration.

The adjoint subgradient algorithm is an extension of the classical subgradient iterative algorithm, which is often used to find two non-integrable convex functions and problems. The algorithm makes full use of the additive structure of the primal problem in Hilbert space and sets the forward gradient step (display step) and the backward approximation step (implicit step) to compute the function nearest operator. This algorithm generates sequences with weak convergence and a more complicated iterative process. When the integrable convex function approximation operator is complex and the subgradient is simple to compute, the neighboring subgradient algorithm can find a more accurate solution [37]. The conjugate sub-gradient algorithm generally uses a linear combination of historical and present gradients to construct the search direction [38], and the previous search direction always remains conjugate to the current gradient direction [39]. Every time the search direction is determined, the error meter of the direction is updated to the minimum, that is, in the process of searching the optimal value, each search direction is searched only once. Therefore, the conjugate subgradient algorithm has the advantages of step convergence, high stability, high requirements for iteration termination results, only the previous search direction is required for each iteration, small computational memory required, and better effect for seeking the solution of large sparse matrices. The spectral distribution of the coefficient matrix determines the convergence efficiency of the conjugate subgradient algorithm and the convergence speed is slower as the eigenvalues are more dispersed and the condition number of the coefficient matrix is larger.

（2）Newton's method

The original Newton's method is designed to solve second-order continuous differentiable convex functions, and its basic idea is to do second-order Taylor expansion on the non-smooth function near the previous minimax estimate, and then find the current minimax estimate, so Newton's method has second-order convergence, and the convergence is better than the gradient descent method. One of the most important features of Newton's method is the use of generalized Jacobi matrix instead of derivatives to solve the maximization of the likelihood function or used to solve the function zero value. Its iterative formula is as follows.

$$x_{k+1} = x_k - H_K^{-1} \cdot g_k, k = 0,1,2,... \quad (13)$$
$$d_k = -H_k^{-1} \cdot g_k$$

Where H_k is the non-singular symmetric Hessian matrix, g_k is the gradient operator, and d_k is the search direction.

However, the iteration of Newton's method is a fixed-step iteration, so Newton's method is not stable in the convergence process, and the iteration sequence can diverge when the objective function is a non-quadratic function [40-42].

In order to ensure that the iteration sequence converges during the iterative process, some scholars have modified the Newton method, i.e., by setting parameters to let the algorithm perform a linear search along the Newton direction to find the next iteration point [43]. This iterative algorithm is called damped Newton's method, which has the feature of large range convergence, and can obtain the globally optimal point. The specific formula is as follows.

$$\bar{x} = x + \alpha p \quad (14)$$
$$p = -[F'(x)]^{-1} F(x)$$

Where x is the historical iterative approximation, and $\bar{x}$ is the present iterative approximation, p is the Newton correction, and $F(x)$ i s the Jacobi matrix at$x$ [26]. The parameterα is the damping factor, which ranges from $\alpha \in (0,1]]$, andα satisfying.

$$\|F(\bar{x})\| = \|F(x) + \alpha\beta\| < (1 - \mu\alpha)\|F(x)\| \quad (15)$$
$$\mu \in (0,1)$$

The damped Newton algorithm also has some drawbacks: the search direction is not easily determined and may not be the optimal descent direction. When the Hessian matrix is singular, the search direction cannot be determined. Also, when the objective function is not quadratic, the optimal solution cannot be solved in one iteration, and the inverse matrix of the Hessian matrix needs to be solved repeatedly, which is not suitable for solving large-scale mixed-integer programming problems [44].

To solve the problems that the search direction of the damped Newton method cannot be determined and the computational complexity is large, scholars use the proposed Newton method matrix instead of the Hessian matrix to approximate the curvature of the objective function and the first-order derivative to construct the

objective function. The proposed Newton's method was first proposed by Davidon in 1959, and the first result was given by Powell in 1971 [45]. The commonly used formulas for the proposed Newton matrix are th e Broyden rank-one correction formula, the symmetric rank-one correction formula, the DFP correction formula, and the BFGS correction formula [46] [47]. Similar to the most rapid descent method, the proposed Newton method requires only the amount of gradient change to obtain a super linear convergence. Based on this, the proposed Newton method is one of the effective methods for solving unconstrained, constrained, and large-scale optimization problems. Despite the small computational complexity of the proposed Newton method and the fast convergence speed of the Newton method, when the global convergence of the proposed Newton method is not stable, the linear search is not suitable [48].

Unlike the Newton method in which the nonlinear equations are expanded to Taylor series, the Gaussian Newton method uses Taylor series expansions to approximate the nonlinear regression function instead, and corrects the regression coefficients several times to approximate the best regression coefficients of the original model to ensure that the residual sum of squares of the original model is minimized. In other words, Gaussian Newton's method is based on the original Newton's method by introducing a reduced order condition and thus neglecting the second order partial derivatives [49]. This approach simplifies the Hessian and second-order matrix calculations while still ensuring the original second-order convergence, and is one of the classical practices for solving nonlinear least squares. In general, the local convergence of the Gaussian Newton method is slow or appears to be non-convergent if the residuals are not convergent to zero. In addition, if the Jacobi matrix is not of full rank, the Gaussian Newton method cannot be used to solve the problem. In order to further improve the global convergence of Newton's method, another scholar [50] combined the trust domain technique with Newton's method and proposed the trust domain Newton's method. This algorithm has no positive definite requirement for the second-order derivative matrix and can obtain better global convergence.

（3）Augmented Lagrangian multiplier method

In 1998, Reockafellar and Wets introduced the augmented Lagrangian relaxation function to solve optimization problems [49][50]. The augmented Lagrangian algorithm transforms a series of stationary value problems containing complex constraints into a series of unconditional generalized stationary value problems by introducing Lagrangian multipliers is the basic idea of the Lagrangian multiplier method [51]. In order to ensure the smoothness of the objective function even after the relaxation, the non-negative constraint of the inequality constraint in the KKL condition [52] can be removed theorem and the inequality constraint is redefined to obtain the augmented Lagrangian function as follows:[53]

$$L(x,\lambda,\sigma)=c^{T}x-\sum_{i=1}^{m}\left\{\lambda_i c^{T}x-\frac{1}{2}\sigma_i\times[c^{T}x]^{2}\right\}-\sum_{j=m+1}^{n}L_j(x,\lambda,\sigma)$$

$$L_j(x,\lambda,\sigma)\begin{cases}\lambda_j(\mathrm{Ax}-\mathrm{b})-\frac{1}{2}\sigma_i\times[Ax-b]^{2} & (\lambda_j-\sigma_j c^{T}x>0)\\ \frac{1}{2}\frac{\lambda_j^{2}}{\sigma_j} & \text{其它}\end{cases} \tag{16}$$

Where, σ_i,σ_j is the penalty factor, and, $\lambda_i\lambda_j$ is the Lagrangian multiplier. Further, there is the iterative formula as follows.

$$\lambda_j^{k+1}=\lambda_j^{k}-\sigma_i^{k}c^{T}x\quad(i=1,2,\dots,m) \tag{17}$$

$$\lambda_j^{k+1}=\max\{\lambda_j^{k}-\sigma_j^{k}(\mathrm{A}x^{k}-\mathrm{b})\}$$

When satisfied $\sigma^{k}=\sqrt{\sum_{i=1}^{m}[c^{T}x]^{2}+\sum_{j=m+1}^{n}[\min((\mathrm{A}x^{k}-\mathrm{b}),\frac{\lambda_j^{k}}{\lambda_j^{k}})]^{2}}$, the iteration stops.

In essence, the augmented Lagrangian algorithm is an extension of the Lagrangian algorithm. Unlike the Lagrange multiplier method, the augmented Lagrange algorithm uses the augmented functional form to keep the values of the penalty function within a reasonable range, i.e., it does not require increasing the penalty parameter to positive infinity to ensure convergence and does not cause pathological problems [54]. To further improve the accuracy of the solution, the Lagrangian function can be minimized unconstrained so that the Lagrangian multiplier infinitely approximates the KKT multiplier for which extreme values are available. Since the optimal KKT multiplier is not available in the practical case, the solution may have some deviations. To further improve the accuracy of the solution, Di.Pillo and Grippo [55][56] added the first-order necessary optimality condition as a penalty term to the original function to improve the quality of the solution. Although the convergence of the augmentation functions proposed by Di.Pillo and Grippo was better, these functions had to use the extremal function, which led to integrability at the infinity point. Later, Aiping Jiang and other scholars introduced the NCP function into the augmented Lagrangian function to solve the problem of integrability of the augmented function at infinity [57].

3.3 Comparison of different iterative strategies

Subgradient descent is the most basic and commonly used optimization method. However, many literatures pointed out [58-60] that the conventional sub-gradient algorithm when the previous s^{k-1} forms an obtuse angle with the current s^{k} (as shown in the figure.5), constitutes a sawtooth phenomenon, making no significant improvement in the accuracy of the search direction from λ^{k-1} to λ^{k+1} in two iterations . That is, the closer the sub-gradient algorithm is to the target value, the smaller the step size is, the slower the search speed is, and the phenomenon of "zigzag" oscillation may also appear (as shown in the figure.5). In response to this oscillation phenomenon, conjugate subgradient algorithm [61], neighborhood subgradient method [62], stochastic gradient descent method, batch gradient descent method and other algorithms are often introduced at this stage to improve the solution efficiency, but due to the limitation of convergence speed and solution quality, these iterative algorithms still have limitations for mixed integer programming problems with large-scale samples and strongly convex bounded conditions.

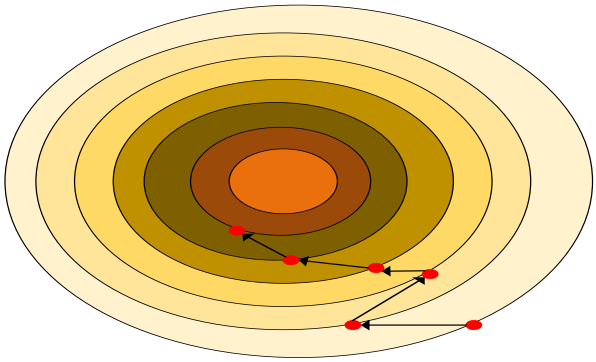

Fig. 5 Sub-gradient descent method search iteration schematic

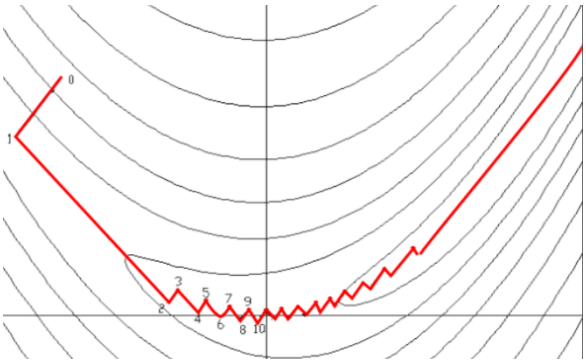

Fig. 6 Sub-gradient iteration algorithm oscillation phenomenon graph

The essence of Newton's method is to use $f(x)$ the first few Taylor series to find the best answers of the equation [63]. Newton's method overcomes the problem of inefficient convergence of the subgradient algorithm and is thus widely used in practical mixed-integer programming problems. In essence, compared to the subgradient descent method, Newton's method is quadratic convergence, and thus Newton's method has good global convergence [64]. However, iteration of Newton's method depends on the previous step of the sought $f'(x)$ in turn, i.e., each iteration of the search requires the solution of the Jacobi matrix of the objective function, which has a large computational complexity. To reduce the computational complexity of Newton's method, domestic and foreign scholars have used algorithms such as the proposed Newton's method [65] and the damped Newton's method to reduce the dependence on the derivative of the objective function. However, such algorithms cannot guarantee the stability and convergence of the search process well, e.g., the exact safety factor cannot be found when the given search area does not contain the most dangerous sliding surface [66].

The compatibility and loss point asymptotic validity of the augmented Lagrange multiplier method [67] effectively overcome the numerical instability and slow convergence of the traditional Lagrange multiplier method [68]. The augmented Lagrange multiplier method adds the quadratic penalty term related to the constraint to the original Lagrange multiplier method, and overcomes the undesirable effects caused by the quadratic penalty term tending to infinity by recursively correcting the Lagrange multiplier. However, the accuracy of the augmented Lagrange multiplier method depends on the KKL condition that the optimal multiplier is known, while the optimal KKL multiplier is unknown in practical mixed-integer programming problems, and is no better method for unconstrained minimization of infinite degree of augmented Lagrangian function [69]. Thus, the multiplicative Lagrangian multiplier method still has large limitations in practical applications.

To address the above problem, the original algorithm can be improved by considering the logarithmic method, i.e., adding a barrier function to the original objective function.

$$B(x) = -\sum_{j=1}^{r} \ln\{-g_j(x)\} \tag{18}$$

$$B(x) = -\sum_{j=1}^{r} \frac{1}{g_j(x)}$$

And a one-step Newton iteration method is used to decompose the minimum obstacle function in the Newton direction. We expect that in the future this algorithm will be better adapted to large-scale mixed-integer programming problems while improving the stability and convergence speed of the algorithm.

4. Application examples

A typical application of the Lagrangian relaxation algorithm is the optimal scheduling of production lines.

The overall planning problem is described as follows: with workpieces i（$i \in \{1,2,\dots,n\}$）, a total of g production processing links, lin kj has L_j machines; processing time is p_{ij}, weight is w_i; transportation time of adjacent workpieces is $\beta_i t_{i,i+1}$; the completion time of the i workpiece on the j processing link is denoted as Q_{ij}; x_n is a 0-1 variable, when the i workpiece is being processed on the j processing link at moment $x_n = 1$, otherwise $x_n = 0$ [70]

$$\text{Min} \sum_{i=1}^{|\delta|} w_i Q_{ij} \tag{19}$$

The following constraints are satisfied:[71]

（1） Machine capacity constraint

The machine capacity constraint means that the number of processes that can be processed on a production line per unit of time cannot exceed the number of machines available in that time period L_j. If a workpiecei Each processing step of the workpiece requires b_{ij}. Each processing step of the workpiece requires one machine. The constraint is as follows.

$$\sum_{i\in\delta} b_{ij} x_{tij} \le L_j, (t = 1,\dots,T; i = 1,\dots,m; j = 1,\dots,m) \tag{20}$$

（2） Process priority constraints

Only when the processing of the j process is completed and the workpiece i is transported to the $j+1$ process, the $j+1$ process can start processing, and the moment of arrival of the workpiece is recorded as α_i.

$$Q_{ij} + \beta_i t_{i,i+1} + p_{i+1,j} \le Q_{ij+1} (j = 1,..,s-1; i = 1,\dots,m) \tag{21}$$

$$g_{1i} \ge \alpha_i, (i = 1,2,\dots,m)$$

（3） Processing time constraint

The machining time constraint is the workpiece i in the j process.

$$\sum_{t=1}^{T} x_{tij} = p_m, (i = 1,\dots,m; j = 1,\dots,s) \tag{22}$$

（4） Variable range constraint:

$$x_{tij} \in \{0,1\} \tag{23}$$

$$J_{ij}, Q_{ij} \in \{1,2,\dots,T\}$$

Let the number of workpieces $i = \{80,120,150\}$, the number of machines is 3, the number of machining stages is 3, the machining time and weights in$[1, L_j]$, obeys uniform distribution . The dynamic arrival time obeys a uniform distribution at [1,5] and the two adjacent processing stages transport time is [4,6] obeys a uniform distribution. The Lagrangian relaxation algorithm is programmed using Matlab, and when the program is stopped after running for 60s, the true pairwise gap is obtained as follows.

Let the target value of the optimal pairwise gap be J^L , and the best upper bound is ϕ^u, then the resulting agent pairwise gap is

$$\frac{(\phi^u - J^L)}{J^L} \times 100\% \tag{24}$$

After solving the Lagrangian relaxation algorithm, the difference between the true agent-pair gap and the agent gap is in the range of

[0.95,2.68] The difference between the agent-pair gap and the true agent gap is extremely small, which indicates that the Lagrangian relaxation algorithm has a superior convergence performance in solving large-scale mixed-integer programming problems from the average performance point of view. Meanwhile, the average pairwise gap of 12.44% is obtained in 60s computation time, which indicates that the Lagrangian relaxation algorithm is extremely applicable to the large-scale mixed integer programming problem, i.e., the Lagrangian relaxation algorithm can obtain a good solution in a shorter time. In addition, as the production scheduling problem scales further (e.g., the number of workpieces increases, etc.), the pairwise gap increases, but the number of iterations decreases and the quality of the solved solution is better than that of other solution methods. Therefore, the Lagrangian relaxation algorithm has better solving capability compared with the conventional methods for solving the production scheduling problem.

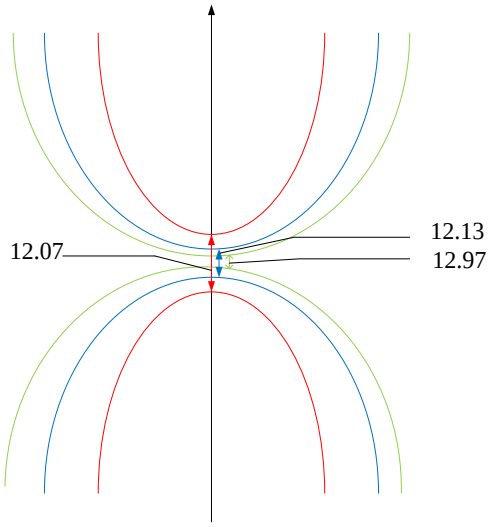

Fig.7 Calculation results

5. Future Outlook

The Lagrangian relaxation algorithm uses the introduction of penalty terms to decompose the large-scale problem into a set of simpler subproblems, and then determines the direction of the Lagrangian multiplier update according to the degree of constraint destruction, coordinates the contradiction between the subproblems, and iterates cyclically to approach the optimal solution. Since the Lagrangian relaxation algorithm can effectively simplify the model, the algorithm has strong modeling ability and is not only applicable to discrete optimization problems, but also to solve continuous optimization problems. In addition, the Lagrangian relaxation algorithm is flexible and can improve a lower bound (upper bound) for the optimal solution of the target problem, narrowing the range of the search for the optimal solution. Taking production scheduling as an example, the Lagrangian relaxation algorithm provides a reliable approximate solution while providing more information about the entire production environment. When different conditional constraints are relaxed, the optimal multipliers reflect the actual situation of the constraints: when the machine capacity constraints are relaxed, the optimal multipliers reflect the machine operation in each time period; when the process constraints are relaxed, the optimal multipliers reflect the sensitivity of the production environment factors to the processing time. Thus, it is more convenient for decision makers to grasp the actual situation and make reasonable allocation of resources [72]. Data experiments show that the Lagrangian relaxation algorithm converges quickly and can obtain a feasible near-optimal solution in a short running time, saving decision time. Based on all the advantages, the extension of its Lagrangian relaxation algorithm can not only obtain good time, resource, and economic benefits in the process industry [73], but also simplify the computational complexity and improve the operational efficiency in other industries such as distribution network state estimation [74], network fault location [75], and multi-sensor multi-objective tracking [76].

At the present research stage, the Lagrangian relaxation algorithm has made good progress for solving large-scale mixed-integer programming problems, obtaining many theoretical results and achieving good benefits in practical applications. However, there is still a large amount of work that needs further depth and improvement.

(1) For how to reduce the complexity of large-scale mixed-integer programming problems, the core idea of Lagrangian relaxation algorithm is still to start from problem decomposition, and rarely consider starting from model simplification directly. If some unimportant constraints can be combined, removed and simplified, the model can be simplified into a "rough model", and the rough solution can be obtained on the basis of the Lagrangian relaxation algorithm combined with the theory of sequential optimization, and then the optimal solution can be approximated step by step according to the need to reduce the complexity of the goal function from the root and improve the quality of the result . In addition, for the present, the research on the solution methods of large-scale mixed integer programming problems under uncertainty is still very limited, while in practical application, there are more uncertainties such as machine failures. Further, further research is needed for the Lagrangian relaxation algorithm for solving large-scale mixed integer programming problems under uncertainty scenarios [77].

(2) The Lagrangian multiplier update mechanism needs to be improved. In Lagrangian relaxation algorithm obtains a new feasible solution by adjusting the multipliers, and then compares this new feasible solution with the best feasible solution obtained in the previous stage. If the new feasible solution is optimal, the new feasible solution is set as the optimal feasible solution, otherwise the feasible solution of the previous stage is maintained and the optimal solution is approached gradually. However, if a better feasible solution is not obtained by adjusting the multipliers several times, the angle formed by two adjacent subgradients will keep increasing, and the more violent the oscillation is, which slows down the convergence of the algorithm. Therefore, a correction to the multiplier update mechanism can effectively improve the solution efficiency of the algorithm [78]. Although at this stage of research, the incremental Lagrangian relaxation algorithm proposed by Peter B. Luch [79] and Dangqing [80] makes timely use of known information to speed up the update frequency of multipliers, the incremental Lagrangian relaxation algorithm is less studied at this stage and still needs further research.

(3) From the analysis of the algorithm examples in the previous section, the key to the solution efficiency of the Lagrangian relaxation relaxation algorithm lies in the solution efficiency of the pairwise problem. Therefore, for some problems with large inherent gaps, the range accuracy of the feasible solutions obtained is not

high, i.e., the quality of feasible solutions is not high. Therefore, how to shorten the pairwise gap based on the existing Lagrangian relaxation framework is a proven way to improve the solution efficiency of Lagrangian relaxation algorithm.

(4) In addition, the meaning of the Lagrange multiplier being zero is less discussed at this stage, and it is usually assumed that when the Lagrange multiplier is zero, the constraint being relaxed by the multiplier actually does not work as a constraint. That is, there may be three cases: the first case is that the feasible solution possesses this property before this constraint; the second case is that the original additional constraint is the natural boundary condition of the original function [81], i.e., the resident value equation of the variational theorem is mistakenly taken as the boundary constraint; the third case is that the initial self-varying function has no binding effect on the selected value range, i.e., the added constraint cannot narrow the self-varying function's The third case is that the initial range of the self-varying function is not constrained, i.e., the added constraints cannot narrow the range of the self-varying function. For the present, the first two cases have been discussed more, while the problem described by the third case has only recently been taken seriously by scholars.

(5) The performance theory of Lagrangian relaxation algorithm needs to be improved. At present, the efficiency of the Lagrangian relaxation algorithm solution is mainly measured by the comparison of simulation results, and there is no guarantee for the quality of the solution under the worst conditions. In addition, it is also impossible to specifically find out the computational complexity and space complexity required to increase the accuracy of the near-optimal solution from 99% to 99.99%, i.e., the existing analytical theory of Lagrangian relaxation algorithm cannot find out the marginal effect of the algorithm. To further improve the solution quality of the algorithm, the analytical theory of Lagrangian relaxation algorithm needs to be improved.

6. Conclusion

In this paper, we study and analyze the solution theory of existing mixed-integer programming problems, focusing on Lagrangian relaxation optimization algorithms for large-scale mixed-integer programming. By comparing different algorithms, the general defects of the existing solution algorithms are summarized. Through the study of existing Lagrangian relaxation algorithms, it is concluded that the basic idea is to replace the multipliers of all subproblems decomposed by the original objective function with the Lagrangian multipliers of a set of subproblems, thus reducing the computational complexity of the original objective function. The search direction is obtained more easily by calculating the approximate solution of the relaxation problem instead of the optimal solution at each iteration, thus demonstrating the feasibility of applying the Lagrangian relaxation algorithm to solving large-scale mixed-integer programming optimization problems. In addition, based on the analysis of the existing iterative algorithms of Lagrangian relaxation, it can be found that the Lagrangian relaxation technique has the following shortcomings: there is a "sawtooth phenomenon" in the multiplier updating process; the dependence on the previous iteration is too high; the computational complexity is large; and the KKL multipliers are not well determined. Based on the Lagrangian relaxation framework, further research work should be carried out on the basis of Newton's method, proxy gradient, genetic algorithm and other optimization techniques, including model supplementation, analytical expression and optimization strategies[82][83].

References

[1] Shanshan Pan. Study on the economic tuning problem of nonconvex power system based on linear mixed integer programming and nonlinear programming [D]. 2019...

[2] Zhang Mingjia. Research on the engineering application of mixed integer programming methods [D]. Huazhong University of Science and Technology, 2005.

[3]Cook S . The complexity of theorem-proving procedures.[C]// Proc Acm Symposium on the Theory of Computation. 1971.

[4] . F. Benders. Partitioning Procedures for Solving Mixed-Variables Programming Problems. Numerische Mathematik, Vol. 4, 1962.

[5] Huo F, Yi B. Analysis of classical Benders decomposition algorithm[J]. Science and Technology Information, 2010(30):141-142.

[6] Gouge. Comparison of cut-plane method and branch-and-bound method in integer programming[J]. Journal of Daxian Normal High School, 2005, 15(2):18-21.

[7] Vijay V. Vazirani. approximation algorithms [M]. Higher Education Press, 2010.

[8] Min-Hui Ji. Forbidden search algorithm[J]. Computer Knowledge and Technology, 2009(27):7748-7750

[9] Zhang XM, Zhang Y, Wang YT, et al. Optimization of multi-time transformer economic operation based on taboo search algorithm[J]. Power Grid Technology, 2010(07):115-119.

[10] Liu Y, Chen YW, Tan YJ. A greedy algorithm-based mission planning method for satellite ground stations [J]. Systems Engineering and Electronics Technology, 2003(10):61-63.

[11] Peng Shibao, Yuan Junquan, Xiang Jiabin. A new method of InSAR phase decoupling based on weighted iterative greedy algorithm[J]. Journal of Electronics and Information, 2008(06):60-64.

[12] Cong M.Y., Wang L.P.. Modern heuristic algorithm theory research[J]. High Tech Letters, 2003(05):105-110.

[13] Wei Yan, Xie Kaigui. Simulated annealing algorithm [J]. Journal of Mengzi Normal High School, 1999, 001(004):7-11.

[14] Haihong Wang, Li Liu. Degree Constrained Minimum Spanning Tree Problem Based on Partheno-genetic Algorithm and Simulated Annealing Algorithm[J]. International Journal of Applied Mathematics in Control Engineering,2019.2(2):188-193.

[15] Ni Qiu-Long, Huang Min-Xiang. Application of simulated annealing algorithm based on branch exchange in distribution network planning[J]. Journal of Power Systems and Automation, 2000, 12(004):31-35.

[16] Duan, Binhai, Wang, Daobo. Research on global convergence and improvement of ant colony algorithm [J]. Systems Engineering and Electronics Technology, 2004, 26(10):1506-1509.

[17] Yang, Yuanjie, Huang, Dao. Artificial neural network algorithm research and application [J]. Journal of East China University of Science and Technology (Natural Science Edition), 2002(05):115-118.

[18] Zhang X.C., Zhou J.J.. Improved artificial neural network algorithm(Ⅰ)-optimization of network structure and convergence criterion[J]. Computers and Applied Chemistry, 1995(3):186-191.

[19] GoemansM X,William son D P. Improved approximation algorithm s for maximum cut and satisfia 2bility problem s using sem idefinite programming[J]. J. A ssoc. Comput. M ach. , 1995, 42: 1115 2 1145.

[20] Xu Y. Research on 3PL transportation coordination and scheduling in MTO supply chain [D]. Huazhong University of Science and Technology, 2011. Xia G , Dong R , Xu J , et al. Linearized Model of Carrier-Based Aircraft Dynamics in Final-Approach Air Condition[J]. Journal of Aircraft, 2015:1-15.

[21] He Xiaoyu. Unit commitment optimization method based on improved Lagrangian relaxation method [D]. North China Electric Power University (Beijing), 2010]

[22] Hu, H. C.. On the Lagrange multiplier method and beyond [J]. Journal of Mechanics, 1985(05):40-48.

[23] Xuan H, Tang L X. Dynamic hybrid flowshop scheduling problem with multiprocessor tasks [J]. Computer Integrated Manufacturing Systems, 2007, 13(11): 2254-2260, 2288.

[24] Liu B, Xu XP, Li JZ. Augmented Lagrangian multiplier method for optimal design of horizontal well injection [J]. Drilling Technology, 2010(03):9-10+79-83.

[25] Zhao Tingting. Convergence of subgradient algorithms with dynamic step size [D]. 2019.
[26] Pawlak Z. Rough Sets: Theoretical Aspects of Reasoning about Data [M].Amsterdam: Kluwer Academic Publishers, 1991.
[27] Bazan J. A comparison of dynamic and non-dynamic rough set methods for extracting laws from decision systems [A]. Rough Sets in Knowledge Discovery[C].Herdelberg: Phyisca-Verlag, 1998. 321-365.
[28] Goffin J. L, Kiwiel K. C. ,Convergence of a simple subgradient level method [J]. Math. Program., 1999, 85(1):207-211.
[29] Kaskavelis, Christos, A, et al. Efficient Lagrangian relaxation algorithms for industry size job-shop. [J]. Iie Transactions, 1998.
[30] Zhao X and Ahn H. Convergence of a generalized subgradient method for nondifferentiable convex optimization.Mathematical Mathematical Programming. 1991.50(1):75-80
[31] Zhou Wei. Supply Chain Planning Optimization in Distributed and Uncertain Environments [D]. Tsinghua University.
[32] Zhang P. Study of several gradient algorithms for nonlinear optimization problems [J]. Journal of Mudanjiang Normal College (Natural Science Edition),2015(01):4-6.
[33] Yu. M. Ermoliev, On the stochastic quasi-gradient method and stochastic quasi-Feyer sequences, Kibernet., 2 (1969), pp. 73 -83.
[34] Yu. M. Ermoliev, Stochastic quasigradient methods, in Numerical Techniques for Stochastic Optimization, Yu. M. Ermoliev and R . J-B. Wets, eds., Springer-Verlag, Berlin, 1988, pp. 141-185.
[35] Geary A , Bertsekas D P . Incremental subgradient methods for nondifferentiable optimization[J]. 2007.
[36] Zhou Y, Hou Zhemei. Fast convergence of an improved stochastic subgradient method[J]. Journal of Computational Mathematics in Higher Education, 2007, 29(1):28-28.
[37] Cruz B , José Yunier. On Proximal Subgradient Splitting Method for Minimizing the sum of two Nonsmooth Convex Functions [J]. Set-Valued and Variational Analysis, 2017, 25(2):245-263.
[38] Hu, Binhai, Lin, Chun-Sheng, Gong, Shen-Guang. Correction of non-ideal orthogonal three-axis magnetic sensors based on conjugate subgradient method[J]. Data Acquisition and Processing, 2003, 18(001):88-91.
[39] Fletcher R,Reeves C.Function minimization by conjugate gradients[J].Computer Journal,1964,7(2):149-154.
[40] Dai Y H. A perfect example for the BFGS method [J]. Mathematical Programming, 2013,138(1-2):501-530
[41] Liu D C, Nocedal J. On the limited memory BFGS method for large scale optimization [J]. Mathematical Programming,1989,45(1-3):503-528
[42] Nocedal J. Upadating quasi-Newton matrices with limited storage [J]. Mathematical of computation,1980,35(151):773-782
[43] Zhang JJ, Li CHQ, Zhang LH. Two important parameters affecting the convergence of the damped Newton method [J]. Pure and Applied Mathematics, 2012(04):433-439.
[44] Shi-Zhen-Jun, Sun-Guo. Diagonal sparse Newton-like method for unconstrained optimization problems [J]. Systems Science and Mathematics, 2006, 26(1):101-101.
[45] Powell M J D . On the Convergence of the Variable Metric Algorithm [J]. J Inst Maths Applics, 1971, 7(1).
[46] Broyden C G. A class of methods for solving nonlinear simultaneous equations. Mathematic of Computation, 1965, 19(2):577-593
[47] Dennis J E. Toward a unified convergence theory for Newton-like memtholds. in:Nonlinear functional analysis and applications. London,1971,425-472
[48] Weijun Zhou. The proposed Newton method and its convergence [D]. Hunan University,2006.
[49] ZhaoShubo, LiLiWei, ZhaoShubo,et al. A note on Gauss-Newton method [J]. Journal of Natural Sciences, Harbin Normal University, 2016, 32(3):6-10.
[50] Xu, Cheng-Xian, Zeng, Yue-Lin. Global convergence of the relying domain type proposed Newton method [J]. Journal of Xi'an Jiaotong University, 1996, 030(012):108-116.
[51] Powell M J D . On the Convergence of the Variable Metric Algorithm[J]. J Inst Maths Applics, 1971, 7(1).
[52] Chen, Zhe. Approximate generalized Lagrangian functions in nonlinear programming [J]. Journal of Fuyang Normal College: Natural Science Edition, 2006(3):18-20.
[53] ZhaoShubo, LiLiWei, ZhaoShubo,et al. A note on Gauss-Newton method[J]. Journal of Natural Sciences, Harbin Normal University, 2016, 32(3):6-10.
[54] Luenberger D G. Linear and nonlinear[M] Programming. Massachusetts: Addision Wesley, 1984
[55] Feng, S. M.. A derivative-free generalized Lagrangian method for nonlinear optimization problems [D]. Dalian University of Technology, 2010.
[56] Pillo G D,Grippo L. An augmented Lagrangian for inequality constraints in nonlinear programming problems [J].Optimization Theorem and Applications,1982,36:495
[57] Di Pillo G , Grippo L . A New Class of Augmented Lagrangians in Nonlinear Programming[J]. Siam Journal on Control & Optimization, 2006, 17(5):618-628.
[58] Jiang A-P, Pu D-G, Duan X-B. A new class of Lagrangian multiplier methods with NCP functions [J]. Journal of Tongji University (Natural Sciences Edition), 2008(05):695-698.
[59] CameriniP M, Fratta L, Maffioli F. On improving relaxation methods by modified gradient techniques []. Mathematical Programming Study, 1975,3:26-34
[60] Braennlund U. A generalized subgradient method with relaxation step[]. Mathematical Programming, 1995,71:207-219 [6] Bazaraa M S, SheraliH D,Shetty C M. NonLinear Programming:Theory and Algorithms[]. John Wiley & Sons, Inc, 1993
[61] Sherali H D, Ulular O. A primal-dual conjugate subgradient algorithm for specially structured linear and convex programming problems []. Applied Mathematics and Optimization, 1989,20:193-221
[62] Hu, Binhai, Lin, Chun-Sheng, Gong, Shen-Guang. Correction of non-ideal orthogonal three-axis magnetic sensors based on conjugate subgradient method[J]. Data Acquisition and Processing, 2003, 18(001):88-91.
[63] Guo K, Yang HZ, Wang T. Convergence rate analysis of the neighborhood subgradient method for solving weakly convex problems[J]. Journal of Mianyang Normal College, 2018, 037(011):7-12.
[64] Xueqin Zhou. A discussion on the convergence order of Newton's method [C]// Proceedings of the 12th China Young Information and Management Scholars Conference. 2010.
[65] Ma, Changfeng. Second-order convergence of the damped Newton method for the double obstacle problem [J]. Economic Mathematics, 1998(Z1):88-94.
[66] Shi-Zhen-Jun, Sun-Guo. Diagonal sparse Newton-like method for unconstrained optimization problems [J]. Systems Science and Mathematics, 2006(01):101-112.
[67] Deng Naiyang, Xue Yi. Global convergence analysis of the proposed Newton method based on the new proposed Newton equation [J]. Journal of Beijing University of Technology, 1999, 25(004):6-12.
[68] Yulei Gao. Model selection:A generalized Lagrange multiplier method[D]. Dalian University of Technology, 2010.
[69] Di Pillo G, Grippo L. A new class of augmented Lagrangians in nonlinear programming[J]. SIAM JControl Optim, 1979, 17(5):618 -628.
[70] Du Xewu, Jin Zhen. An exact generalized Lagrangian function for inequality constrained optimization problems[J]. Journal of Shanghai Jiaotong University, 2006, 040(009):1636-1640.
[71]]Xuan H, Tang L X. Dynamic hybrid flowshop scheduling problem with multiprocessor tasks[J]. Computer Integrated Manufacturing Systems, 2007, 13(11): 2254-2260, 2288.
[72] Xuan-Hua, Feng-Ying, Li-Bing, et al. Improved Lagrangian relaxation for HFS multiprocessor task scheduling [J]. Control Engineering, 2017(10):5-8.
[73] Tao Jiping. Research on scheduling algorithm based on Lagrangian relaxation method [D]. Shanghai Jiaotong University, 2014.
[74] Xuan-Hua. Improved Lagrangian relaxation algorithm for scheduling mixed flow shop with limited transportation capacity[J]. Computer Integrated Manufacturing Systems, 2013, 19(007):1633-1639.
[75] Liu, R. S., Sheng, W. X., He, K. Y., et al. Distributed state estimation of complex active distribution networks based on Lagrangian relaxation technique[J]. Power System Protection and Control, 2017, 45(015):125-132.
[76] Qiu-Hua Zheng, Min Yao, Rushing Tao Qian. A new method for network fault localization based on Lagrangian relaxation and subgradient method [J]. Systems Engineering Theory and Practice, 2008, 28(11):155-164.
[77] Zhou L, He Y, Wang F, et al. Study of Lagrangian relaxation algorithm in multi-sensor multi-objective data interconnection[J]. Journal of Qingdao University: Natural Science Edition, 2004(04):29-33.
[78] Jin Feng, Wu Cheng. Research status and outlook of large-scale production scheduling problems [J]. Computer Integrated Manufacturing Systems, 2006, 12(2):161-168.
[79] Z. H. Yang, X. R. Yu. A new Lagrangian relaxation method and its applications [J]. Control and Decision Making, 1989, 000(001):18-22.

[80] Peter B. Luh, Debra J. Hoitomt. Scheduling of manufacturing systems using the lagrangian IEEE Transactions on Automatic Control, 1993, 38(7):1066-1079

[81] Danqing Yu, Peter B. Luh. A new lagrangian relaxation based method to improve schedule quality. IEEE International Conference on Intelligent Robots and Systems, 2003, 3:2303-2308

[82] Liangliang Sun, Liang Sun, Jiali Xi, Yuheng Song, Qiong Xia, Yiyang Liu. An Improved Optimization Strategy for Seed Liquid Preparation and Fermentation Process[J]. International Journal of Applied Mathematics in Control Engineering,2020.3(2):198-205.

[8] Wang Yang, Fanwei Meng. Modeling and Simulation of Magnetic Levitation System based on Lagrange Equation [J]. International Journal of Applied Mathematics in Control Engineering,2020.2(2):164-169.

Yue Chen studied in Shenyang Jianzhu University, her major is building electrical and intelligent.

Liangliang Sun receive his PhD degree in Control Theory and Control Engineering from Northeastern University, China, in 2015. He is currently an Assoicate Professor of Information and Control Engineering at Shenyang Jianzhu University. His current research interests include Intelligent Manufacturing Systems, planning, scheduling, and coordination of design, manufacturing, and service activities intelligence field.

Baolong Yuan obtained his M.S. study at the Shenyang Jianzhu University, Liaoning, China. He is an associate professor of Shenyang Jianzhu University. His research interest includes theory and application of intelligent control and computer control and communication technology.

Xiaoxi Tian obtained his graduate diploma at the Shenyang Jianzhu University, Liaoning, China. He is a lecturer of Shenyang Jianzhu University. His research interest includes application of intelligent control and IC Reliability.

Qinghua Shi obtained his Master`s Degree at the Shenyang Jianzhu University, Shenyang, China. He is a lecturer of Shenyang Jianzhu University. His research interest includes the design and manufacture of ceramic bearings.