

Article

Optimization of a Rural Portfolio Credit Granting System Using Improved Two-Dimensional Strip Packing Grouping Delay Problem

Huijun Huang¹ and Yuzhong Li^{2,*}¹ School of Politics and Public Management, Guangxi Minzu University, Nanning 530006, China² School of Computer Science and Engineering, Huizhou University, Huizhou 516001, China

* Correspondence: cslyz@hzu.edu.cn

Abstract: Rural preferential loans usually take the form of portfolio credits. From the perspective of public interest, the total delay time for obtaining loans is expected to be minimized. To use rural portfolio credits effectively, the two-dimensional strip packing grouping delay problem (2SPGDP) is improved to optimize the rural portfolio credit granting system. First, 2SPGDP is established by adding grouping constraints and the latest start time constraints to the two-dimensional strip packing problem, and the total delay is taken as the optimization objective. Second, based on the depth search reverse spanning tree (DSRST) and the insert spare space (ISS) method, the branch-and-bound reverse order insert algorithm (BB-RIA) is designed. Finally, the lag pruning operator (LPO) is designed to reduce lag. The improved model (2SPGDP) and BB-RIA-LPO algorithm are used to solve several classical two-dimensional strip packing problems and a specific rural portfolio credit case. Compared with the Bottom-Left and Branch and Bound Algorithm, our model and algorithm improve the success rate by 25% and reduce the total delay by 6%. The case of rural portfolio credit illustrates the operability and effectiveness of this method.



Citation: Huang, H.; Li, Y. Optimization of a Rural Portfolio Credit Granting System Using Improved Two-Dimensional Strip Packing Grouping Delay Problem. *Systems* **2022**, *10*, 193. <https://doi.org/10.3390/systems10050193>

Academic Editor: William T. Scherer

Received: 20 August 2022

Accepted: 18 October 2022

Published: 21 October 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: two-dimensional strip packing problem; grouping; deterministic algorithm; branch and bound; delay; rural portfolio credit

1. Introduction

Rural preferential finance is a requirement by the government regarding the public interest and social responsibility of rural financial institutions. Rural commercial banks will set favorable terms to enable villagers and rural small businesses to obtain preferential loans and increase their satisfaction with the government [1]. Rural preferential credit has many forms, among which “whole village credit” is a new rural portfolio credit method of “whole village wholesale centralized credit” established in the process of “batch credit, self-help lending, without guarantee”, aiming to meet the needs of the rural financial granting system [2–6]. In the rural portfolio credit plan project, the risk control mechanism is set for each village. When the nonperforming loan rate of a village is higher than 3%, village credit is stopped; if it is less than 3%, the loss will be shared by the guarantee company established by the government and the bank to ensure the progress of the planned project. If a strip box is set, the height is the total amount of rural portfolio credit of rural commercial banks, and the length is time. Due to the risk control mechanism, this paper does not consider the reduction of the total amount of loans caused by nonperforming loans, that is, the reduction of the height of the box. The loan items are grouped according to their villages to generate group constraints. The latest loan-obtaining time for each village is set to be the same. Each loan item is a rectangle, with the height and length representing the loan amount and loan term, respectively, and the final optimization target is the total loan-obtaining time delay. Then, the rural portfolio credit scheduling problem is abstracted as a two-dimensional packing problem with the goal of group constraints and total delay time, which is called

the two-dimensional packing grouping delay problem in this paper (2SPDGP). Solving such problems can solve the problem of capital occupation of rural commercial banks, achieve collective management through village credit, reduce risks, optimize the total delay time, improve satisfaction, and promote the sustainability of rural portfolio credit financial systems. Figure 1 shows the credit relationship between the government, the village committee, and the villagers. The government gives credit to the rural commercial bank, and the rural commercial bank gives credit to the village committee and directly makes and recovers loans to the villagers. The village committee and the rural commercial bank jointly manage the loan item of villagers.

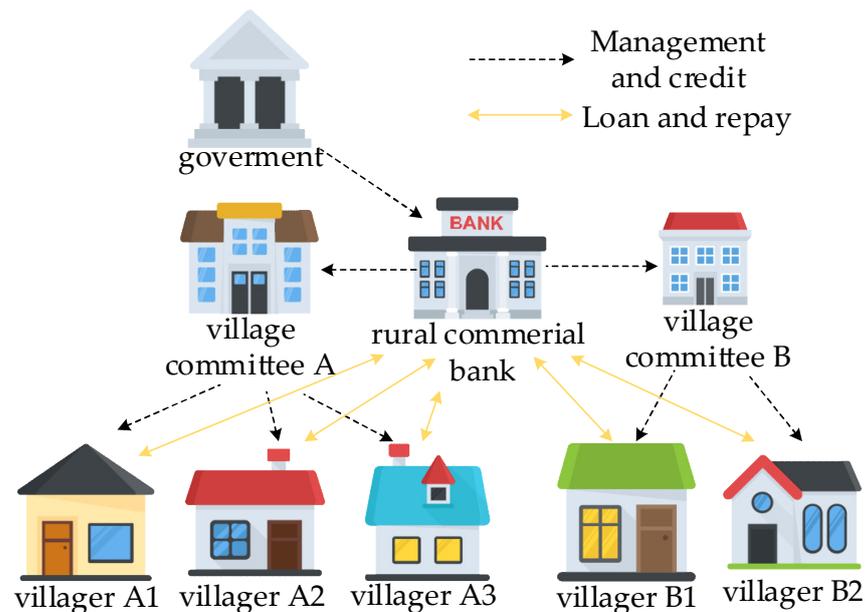


Figure 1. Rural credit relationship.

In recent years, many studies and applications have been conducted on the algorithm of the two-dimensional packing problem (2SPP) [7]. Deterministic algorithms mainly use models to construct accurate solutions or improve the branch pricing method to improve the optimization ability and efficiency of the algorithm. Cid-Garcia et al. [8] used the location and coverage method to obtain the two-dimensional, non-guillotine-constrained 2SP exact solution, generated a set of effective locations in the form of pseudo polynomials, and selected the best configuration of the strip box items by using the set coverage formula. Qi et al. [9] studied the relationship between the lower left coordinates of the rectangles and strip boxes and established linear integer programming models of nonrotating and rotating two-dimensional rectangular strip boxes to ensure that the two rectangles would not be placed repeatedly. Queiroz [10] presented a tailored branch-and-cut algorithm for the two-dimensional irregular strip packing problem with uncertain demand for the items to be cut. A two-stage stochastic programming model is developed, considering a discrete and finite set of scenarios. However, precise algorithms rarely study the problem of taking delay as the optimization goal. The heuristic algorithm is fast and can obtain the local optimal solution. Oviedo-Salas et al. [11] proposed an improved greedy random adaptive search process (GRASP) in which three loss functions are used as the alternative objective functions of the GRASP candidate list, and overlapping functions are used to ensure that the object is suitable for labelling; this method shows improved performance compared to other metaheuristic algorithms. Wei et al. [12] proposed an improved heuristic algorithm based on Best-Fit that used adaptive values to select the best rectangle suitable for the gap, providing an efficient implementation mode with $O(n \log n)$ time complexity for the improved heuristic algorithm of the best fit. By trying different sequences, a simple random local search was used to improve the results. Chen and Chen [13] presented a

corner increment-based algorithm for solving the two-dimensional strip packing problem without the guillotine constraint; it adopted heuristics in the corner increment strategy, segment tree, and multistart random local search. Chen et al. [14] proposed a variant of a hybrid heuristic algorithm for optimizing solutions to the strip packing problem; they adopted scoring rules for selecting rectangles and used red-black trees that store rectangle indices and random local search. A genetic evolutionary algorithm is often used to improve the efficiency of a solution. Li et al. [15] proposed the hybrid adaptive genetic algorithm (HAGA) for solving the NP-hard two-dimensional rectangular packing problem to maximize the filling rate of a rectangular sheet. Mondal and Tsourdos [16] presented a two-dimensional quantum genetic algorithm (2D-QGA), which is a new variety of QGA. Recently, reinforcement learning has been added to the two-dimensional packing problem. Neuenfeldt et al. [17] adopted supervised machine learning techniques to select multilabel dataset transformation for solving metal and paper roll shearing problems in industrial processes that improved the solution accuracy compared with other improved heuristic algorithms. Although many heuristic and evolutionary computing methods can improve the efficiency of the solution, these algorithms introduce random numbers, and the solution is uncertain in the same case. In the actual operation of bank portfolio credit, it is easy to cause disputes among lenders, so it is not applicable. There are also some applications in the field of economic management on the two-dimensional packing problem. Zheng et al. [18] study a realistic aircraft scheduling and parking problem with the goal of simultaneously determining the takeoff and landing time of each aircraft with consideration for wake vortex effect constraints and parking positions in the limited parking apron at a target airport. However, there are few applications in the field of bank credit. The 2SPP literature review also mentions the classification of various constraints. Vega-Mejia [19] provides an overview of recent optimization developments for integrating packing and routing problems to propose a simple classification scheme for realigning the optimization criteria and operational constraints, taking into consideration the issues of sustainability. Júnior et al. [20] conducted a literature review of current 2SP problems in practical problem constraints. These reviews discussed most of the constraints but did not address combinatorial constraints and the latest start length constraint. In addition, there is an exponential relationship between the computational cost and the problem size of the two-dimensional packing problem algorithm [21].

In summary, among the literature on 2SPP, there are few studies on the delay time objective in deterministic solution methods. Due to the uncertainty of the solution results under the same conditions, many heuristic evolutionary hybrid algorithms are not applicable to bank portfolio credit, which easily causes controversy. The application of 2SPP in the field of bank credit is rare. There are few studies on combinatorial constraints and fairness strategies. Starting from the bank portfolio credit problem, this paper abstracts the two-dimensional strip packing mathematical problem of grouping (combinatorial constraints) and delay optimization for research. First, the two-dimensional strip packing problem is modified based on the bank portfolio credit problem. By adding the group credit, the latest start time and the total delay corresponding to the bank portfolio credit problem as the grouping constraint, the latest start length, and the delay optimization goal, a two-dimensional strip packing grouping delay optimization problem (2SPGDP) is established. Second, based on the deep search reverse spanning tree (DSRST) and the insertion spare space (ISS) operator, a branch-and-bound reverse order insert algorithm (BB-RIA) is designed to cover the state of most of the decomposition space and effectively use the free space of the strip box. Third, a lag pruning operator (LPO) is designed for the goal of delay optimization, which can be used as an optimization criterion to obtain a better solution. Finally, the improved classical test examples and a specific case are used to verify the effectiveness of the algorithm.

2. Problem Description and Mathematical Model

In this section, we abstract the rural portfolio credit scheduling problem into the 2SPGDP scheduling problem and provide a simple example.

2.1. Problem Description

Table 1 lists the symbol descriptions for the problem and the meaning in the rural portfolio scheduling problem. If a strip box has a certain height H and infinite length, there are g rectangle groupings represented by $G_k, k \in \{1, 2, \dots, g\}$, each rectangle group has m_k rectangles, represented by $R_{ki}, i \in \{1, 2, \dots, m_k\}$, and the height and length of the rectangle R_{ki} are h_{ki} and l_{ki} , respectively. $h_{ki} < H, l_{ki}$ is a limited value and integer. R is the set of all rectangles, $R = \{R_{ki} | k = 1, 2, \dots, g, i = 1, 2, \dots, m_k\}$, which are the rectangular schedules in the strip box. The start and end lengths of each rectangular schedule are SL_{ki} and EL_{ki} , respectively. Each set of rectangles has the latest starting length constraint LSL_{ki} . If it is longer than this length, the penalty function will be calculated based on the delay length. The delay length is defined as the difference between the rectangle start length SL_{ki} and the latest start length LSL_{ki} after scheduling. If the delay length is negative, it means there is no delay. If it is positive, it is included in the delay, as shown in the following equation:

$$DL_{ki} = \begin{cases} SL_{ki} - LSL_{ki}, & \text{if } SL_{ki} - LSL_{ki} > 0 \\ 0, & \text{if } SL_{ki} - LSL_{ki} \leq 0 \end{cases} \quad (1)$$

Table 1. Symbol and parameter description and scheduling problem meanings.

Symbol	Description	Meaning in Rural Portfolio Credit Scheduling Problem
G_k	The group $k (k = 1, 2, \dots, g, k \in N+)$	Group k of portfolio credit project
R_{ki}	The i -th rectangle of Group $k (i = 1, \dots, m_k, i \in N+)$	The i -th item of Group k
h_{ki}	Height of the i -th rectangle of Group k	Loan amount of the i -th item
l_{ki}	Length of the i -th rectangle of Group k	Loan term of the i -th item
LSL_{ki}	The latest start length of the i -th rectangle of Group k	Latest loan obtaining time for the i -th item of Group k
GHC_k	Height constraint of Group k	The total credit of Group k
SL_{ki}	The start length of the i -th rectangle in Group k	Scheduling start time for the i -th item of Group k
EL_{ki}	The end length of the i -th rectangle in Group k	Scheduling end time for the i -th item of Group k
DL_{ki}	The scheduling delay length of the i -th rectangle in Group k	Scheduling delay time for the i -th item of Group k

The rectangle grouping constraint, defined as the sum of the heights of the same group of rectangles, cannot exceed the height of the constraint at the same length position, as shown in the following equation:

$$\forall \bigcap_{i=1}^{m_k} h_{ki} \neq \emptyset, h_{ki} \in [ST_{ki}, ET_{ki}) \cap I \quad (2)$$

$$\sum_{i=1}^{m_k} h_{ki} \leq GHC_k$$

The optimization goal of the scheduling problem is to calculate the minimum sum of all delay lengths under rectangular grouping constraints, as shown in Equation (3):

$$DL = \min \sum_{k=1}^g \sum_{i=1}^{m_k} DL_{ki} \quad (3)$$

The goal of the scheduling problem was to minimize the sum of the total scheduling delay times DL .

2.2. Simple Example

The following content will illustrate the process of abstracting the rural portfolio credit scheduling problem into 2SPGDP with a simple case: A bank’s total amount of rural portfolio credit is limited to 6 million yuan, and villages 1 and 2 participate in the portfolio credit project. In the first village, three villagers borrow money, with credit lines of two, two, and one million yuan, and the terms are two years, one year, and two years, respectively. The latest loan-obtaining time in this village is 0.2 years, and the credit for the whole village is 4 million yuan. In the second village, two villagers’ loan amounts are two million yuan and one million yuan, with terms of two years and one year, respectively. The latest loan-obtaining time for this village is 0.4 years, and the credit for the whole village is 3 million yuan. We abstract the village as a group k , the total credit $H = 6$, the villagers’ loan item is abstracted as rectangular R_{ki} , the loan amount is abstracted as h_{ki} , the loan term is abstracted as l_{ki} , the latest loan obtaining time is abstracted as LSL_k , and the credit of the village is abstracted as GHL_k . See Table 2 for details.

Table 2. Simple example parameters.

Rural Portfolio Credit	2SPGDP					
village	Group k	1	1	1	2	2
loan item	Rectangle R_{ki}	1	2	3	1	2
loan amount(million yuan)	Height h_{ki}	2	2	1	2	1
loan term(year)	Length l_{ki}	2	1	2	2	2
latest loan obtaining time (year)	Latest start length LSL_k		0.2			0.4
village credit limit(million yuan)	Height Constraint GHL_k		4			3

Figure 2 is a simple example of 2SPGDP scheduling. Compared with the classical packing problem, we rotate the box 90° clockwise and transform the width W and height H of the classical box into H and L described in this problem. According to the Bottom-Left algorithm, we first find the minimum length, then we find the coordinate point of the minimum height as the placement point and place the point at the bottom left corner of the rectangle at the placement point. We complete the scheduling according to the order and constraints in the table. In the figure, we can see that R_{22} does not place a point according to the BL algorithm because of the constraints of this group of heights GHL_2 .

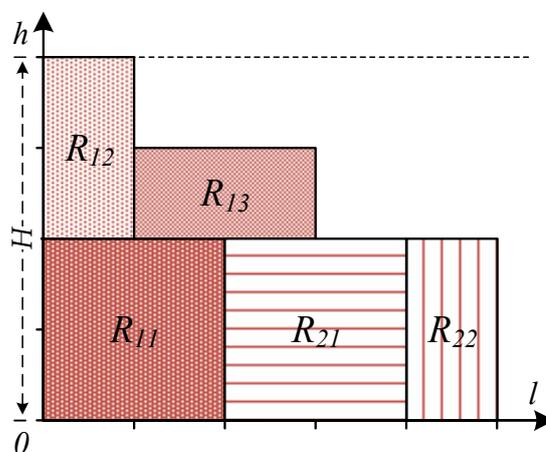


Figure 2. Simple example of 2SPGDP scheduling.

Finally, we determine all the scheduling start times $SL_{11}, SL_{12}, SL_{13}, SL_{21}$ and SL_{22} as 0, 0, 1, 2, and 4, respectively. According to Equation (3), we calculate the total delay

$$DL = \sum_{k=1}^g \sum_{i=1}^{m_k} DL_{ki} = 0 + 0 + (1 - 0.2) + (2 - 0.4) + (4 - 0.4) = 5.2$$

Therefore, there is a total delay time of 5.2 years when switching to rural portfolio credit.

2.3. Mathematical Model

This subsection presents a mixed-integer programming model for the two-dimensional strip packing grouping delay optimization problem (2SPGDP). This problem is based on five assumptions as follows [22]:

- (1) All rectangles must be packed into a strip box.
- (2) The sides of the rectangle must be parallel to the strip box, that is, right-angle filling.
- (3) The rectangle cannot be rotated.
- (4) Any two rectangles cannot overlap.
- (5) One size fits all must not be a requirement.

There are several rectangles $R(x_{ri}, y_{ri})$ in rectangle group G_r , and (x_{ri}, y_{ri}) is its minimum coordinate after scheduling, namely:

$$x_{ri} = \min\{p | p \in [x_{ki}, x_{ki} + l_{ki}), p \in I\}, y_{ri} = \min\{q | q \in [y_{ki}, y_{ki} + h_{ki}), q \in I\} \quad (4)$$

Additionally, I is a set of natural numbers. To schedule them in the strip box with height H and length L that has infinite length, all rectangles should be scheduled. The model aims to minimize the sum of the delay lengths. Since all rectangles must be scheduled, the description of the rectangles R_{ri} and R_{sj} according to Scheithauer’s mixed linear integer model based on mutual positions is defined through the binary variables $u_{ri,sj}$ and $v_{ri,sj}$. If $r = s$ then $i \neq j, i, j, r, s \in I$; then, the mathematical model is given by:

$$z = \min \sum_{k=1}^g \sum_{i=1}^{m_k} DL_{ki} \text{ s.t.} \quad (5)$$

$$0 \leq x_{ri} \leq L - l_{ri}, 0 \leq y_{ri} \leq H - h_{ri}, r, i \in I, \quad (6)$$

$$x_{ri} + l_{ri} \leq x_{sj} + \bar{L}(1 - v_{ri,sj}), y_{ri} + h_{ri} \leq y_{sj} + H(1 - v_{ri,sj}), i, j, r, s \in I, \text{ if } r = s \text{ then } i \neq j \quad (7)$$

$$u_{ri,sj} + u_{sj,ri} + v_{ri,sj} + v_{sj,ri} = 1, i, j, r, s \in I, \text{ if } r = s \text{ then } i \neq j \quad (8)$$

$$u_{ri,sj}, v_{ri,sj} \in \{0, 1\}, i, j, r, s \in I, \text{ if } r = s \text{ then } i \neq j \quad (9)$$

$$\forall \bigcap_{i=1}^{m_k} h_{ki} \neq \emptyset, h_{ki} \in [ST_{ki}, ET_{ki}) \cap I, \sum_{i=1}^{m_k} h_{ki} \leq GHC_k \quad (10)$$

$\bar{L} \in Z_+$ is the length of a feasible scheduling pattern, Z_+ is a positive integer, and I is a natural number. Condition (6) states that all rectangles $R_{ki}(x_{ki}, y_{ki})$ are placed in a horizontal strip box of height H and length L . According to Condition (5), the optimization goal is to minimize all delay lengths. Condition (8) ensures that the position relations between any two pairs of rectangles can only be one of the four directions, namely, the determinable variables $u_{ri,sj}, u_{sj,ri}, v_{ri,sj}$, and $v_{sj,ri}$, and only one of the four position relations can be equal to 1. Condition (7) indicates that all rectangles will not overlap horizontally or vertically; this model can also be taken as a Padberg-type model due to the use of binary variables. The grouping constraint scheduling policy has constraints on the sum of the scheduling lengths for all rectangles belonging to the same group. Grouping constraints are used to limit the total height of a group of items of the same length. Equation (9) indicates that the value of the binary variable of the relationship is either 0 or 1. Constraint (10) is a grouping

constraint. If all rectangles in the same group are scheduled at the same length, their sum at this length cannot be more than GHC_k .

It should also be considered that the number of binary variables is either too large to be solved by standard computer software for strip packing problem (SPP) instance problems or too computationally expensive to be solved. In any case, the model provides a lower bound of optimized length or an approximate solution.

The natural lower bound of the grouped SPP model can be deduced as follows. Due to the grouping constraint, regardless of the location of the rectangle, the natural lower bound of each group is given by:

$$LB_{G_k} = \sum_{i=1}^{m_k} h_{ki}l_{ki} / GHC_k \tag{11}$$

Then, the natural lower bound of all rectangular group scheduling can be estimated as follows:

$$LB_G = \sum_{k=1}^g LB_{G_k} / H = \sum_{k=1}^g \sum_{i=1}^{m_k} h_{ki}l_{ki} / H \tag{12}$$

2.4. Lower Bound of Group Fairness

Figure 3 shows an example of calculating the delay time score. In this paper, we consider the fairness policy in the same group; that is, intragroup fairness is considered. The latest start time LSL_k for each group is set to the first scheduled time of the tasks in the group.

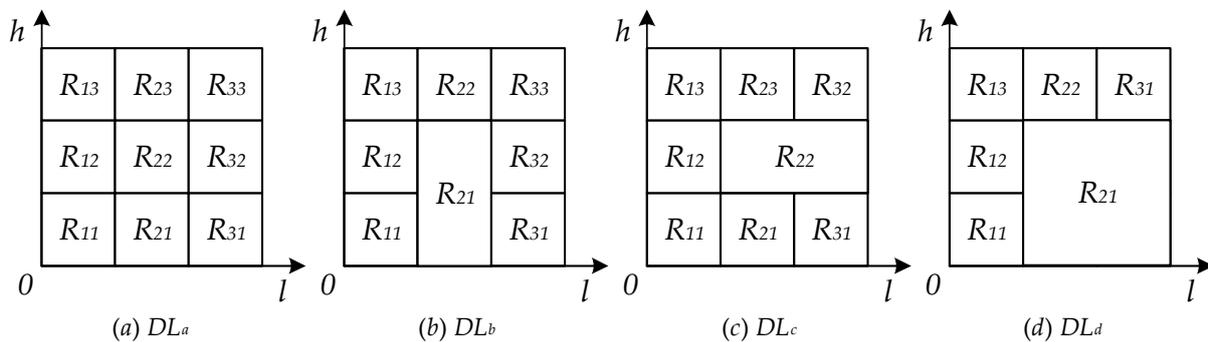


Figure 3. Example of calculating the delay time score.

Then, we can obtain the lower bound of the optimization target delay time by reasoning as follows. Our latest start time LSL_k is 0. According to Equation (5), we calculate the delay time in Figure 3a–d. The delay time is $DL_a = 3 \times (1 + 2) = 9$, $DL_b = 3 \times (1 + 2) - 1 = 8$, $DL_c = 3 \times (1 + 2) - 2 = 7$, and $DL_d = 1 \times 2 + 2 \times 1 = 5$.

Therefore, we derive the formula of arbitrary rectangular scheduling to reduce the calculation delay:

$$DL = \sum_{j=1}^{EL_{gmg}} j \times h_{ki} - \left(\sum_{j=SL_{ki}-LLC_k}^{EL_{ki}-LLC_{ki}} j \right) \times h_{ki} + (SL_{ki} - LLC_k) \tag{13}$$

Proof: Suppose there is a strip box of width W and unlimited length, each rectangle is a unit rectangle, and the latest start time of all rectangles is 0, then the maximum delay generated by all rectangles is:

$$DL_{UB} = \sum_{j=1}^{EL_{gmg}} j \times h_{ki} \tag{14}$$

When an arbitrary rectangle R_{ki} of the height h_{ki} and the length l_{ki} is scheduled, the maximum delay it can actually reduce is:

$$DL_{R_{ki}} = \left(\sum_{j=SL_{ki}-LLC_k}^{EL_{ki}-LLC_{ki}} j \right) \times h_{ki} \tag{15}$$

Finally, it must add its own delay at the front, and that delay is $(SL_{ki} - LLC_k)$, which concludes the proof.

$$DL = \sum_{k=1}^g DL_{G_k} = \sum_{k=1}^g \left[\sum_{j=1}^{EL_{gmg}} j \times h_{ki} - \left(\sum_{j=SL_{ki}-LLC_k}^{EL_{ki}-LLC_{ki}} j \right) \times h_{ki} + (SL_{ki} - LLC_k) \right] \tag{16}$$

3. Branch and Bound Reverse Order Insert Algorithm

To solve the portfolio credit-granting problem, based on the two-dimensional strip packing grouping delay optimization problem (2SPGDP), we design a deep search reverse spanning tree (DSRST) to generate a solution space for minimizing the sum of the latest start time lag of all rectangles and consider fairness in the same group at the latest start time. An insert spare space (ISS) is used instead of a skyline [12] to insert new rectangles. The lag pruning operator (LPO) is used to cut off the lagging branches. Based on the improvements above, we design the branch-and-bound reverse order insert algorithm (BB-RIA) and consider the equivalence and dominance of the classical pruning strategy to improve the quality and efficiency of the solution. These methods are described in detail in this section.

3.1. Deep Search Inverse Spanning Tree (DSRST)

The algorithm is based on a deep search reverse spanning tree (DSRST) to generate a search space. This method can cover the states in most search spaces and avoid generating solution space state explosion. Figure 4 shows the deep search inverse spanning tree (DSRST) of three variables; the variables are 1, 2, and 3, and 0 is set as a pseudo variable (node) for exchanging the first variable. Starting from the leftmost branch, Stages 0–3 are searched in-depth in the order of 0-1-2-3; the second branch generating the third branch is taken as an example. The second branch backtracks from Stage 3 and pushes variables to Stage 1. In contrast, variable 2 at the top of the stack does not appear in Stage 1, so it starts from variable 2 of Stage 1 to variable 1 of Stage 3 in the deep reverse back stack to complete the generation of the third branch, as shown by the dotted line in Figure 4.

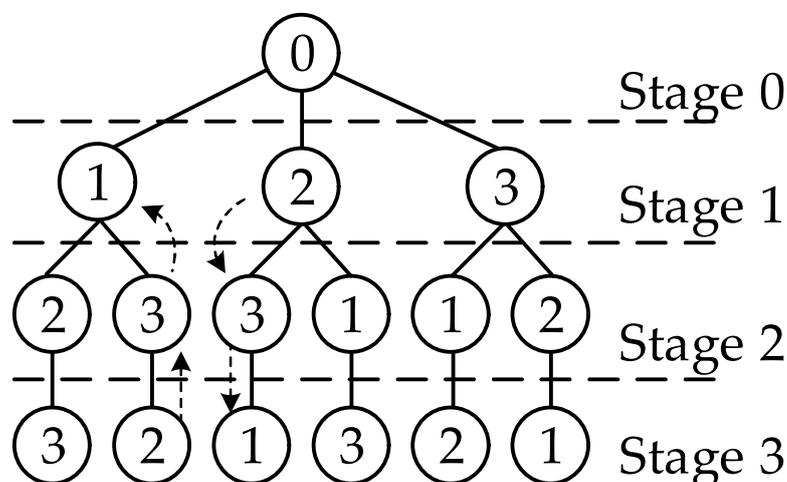


Figure 4. Deep search reverse spanning tree (DSRST).

3.2. Insert Spare Space (ISS)

The appropriate position in the strip box is selected, and the new rectangle is scheduled using the insert spare space (ISS) method. Compared with the skyline method, the frequency of searches increases, but the waste of space is also reduced, and a better solution can be obtained.

The rectangle placement point is set as (x,y) and $R_{ki}(x,y) \in I$ to define the rectangle coverage:

$$R_{ki}(x,y) := \{(p,q) \in R^2 : x \leq p \leq x + l_{ki}, y \leq q \leq y + h_{ki}\} \quad (17)$$

A packing mode is defined as:

$$A(\tilde{I}) := \{R_{ki}(x_{ki}, y_{ki}) : ki \in \tilde{I}\} \quad (18)$$

For a related mode of packing, $\tilde{I} \in I$ is a subset of the set of all of the rectangles. When all scheduled rectangles are placed left first and then right, this is called regularization or upper-left alignment. The contour of the $A(\tilde{I})$ mode is defined as:

$$U(A(\tilde{I})) := \{(x,y) \in R_+^2 : \exists ki \in \tilde{I}, x \leq x_{ki} + l_{ki}, y \leq y_{ki} + h_{ki}\} \quad (19)$$

To make the placement more compact, the placement point is designed as the contour corner point, which is the point aligned at the bottom left corner of the contour except for the origin (0,0):

$$S_{cor-pt} = \{(x,y) \mid (x,y), (x_1,y_1) \in U(A(\tilde{I})), \\ (x > 0 \cap y_1 = 0) \cup [(y_1 = y - 1) \cap x_1 > x]\} \quad (20)$$

According to the coordinate axes in Figure 4, two boundary corner points are defined as:

$$\begin{aligned} p_{bou-H} &= (x_{bou-H}, H), x \in U(A(\tilde{I})), x_{bou-H} = \max(x) \\ p_{bou-0} &= (x_{bou-0}, 0), x \in U(A(\tilde{I})), x_{bou-0} = \max(x) \\ x_{bou} &= \min(x_{bou-0}, x_{bou-H}) \end{aligned} \quad (21)$$

$X_{ex-ctr-0}$ and $X_{ex-ctr-H}$ are the horizontal coordinate values of the intersection of the outer contours $x = 0$ and $x = H$, respectively, that is, the height boundary of the strip box. X_{ex-ctr} is the minimum value of the two corner points. Then, we can define the set of outer contour placement points and inner contour placement points:

$$S_{pt-ex} = \{(x,y) \mid (x,y) \in S_{cor-pt}, x \geq x_{ex-ctr}\} \quad (22)$$

$$S_{pt-in} = \{(x,y) \mid (x,y) \in S_{cor-pt}, x < x_{ex-ctr}\} \quad (23)$$

Then, according to relevant definitions, it can be shown that the set of placement points of the outer contour is the set of placement points generated by the classical skyline, while the set of placement points of the inner contour is the set of points searched by the insert spare space method described in this subsection when the new rectangle is placed. Increasing the set of inner contour placement points increases the cost of the search calculation.

As shown in Figure 5, Rectangles R_1 , R_2 , and R_3 are set with widths of 2, 1, and 1 and heights of 1, 2, and 1, respectively. The scheduling sequence is 1–3. In the problem of regularization scheduling in this paper, the point with the shortest length is considered first, and then the point with the lowest height is considered the starting point to schedule the current rectangle. When the scheduling of rectangle R_3 is prepared, the skyline method seeks the scheduling placement point from the regularization position of the outer contour, that is, the position of the placement point is at (3,0), as shown in Figure 5a. The ISS method

starts from the regularization integer point (0,0) to search for the scheduling placement point; thus, the point (0,1) searched is the scheduling placement point, as shown in Figure 5b. Therefore, the ISS method can save space.

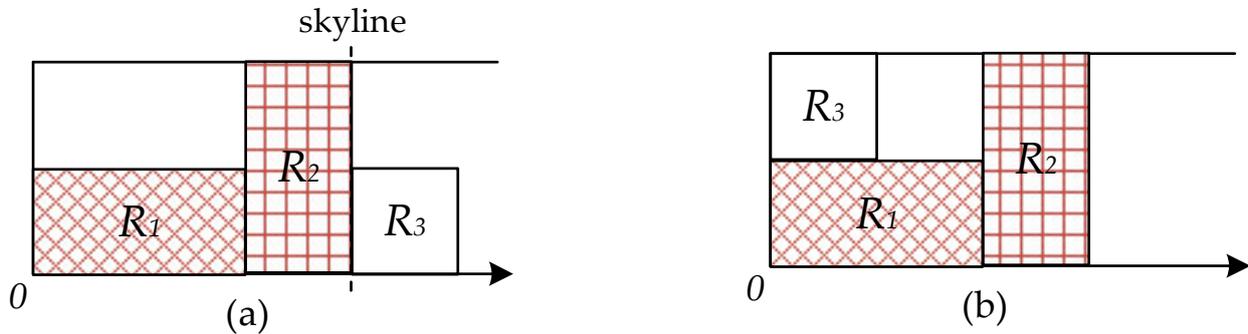


Figure 5. Skyline method and interpolation method: (a) Skyline method; (b) insert method.

3.3. Lag Pruning Operator (LPO)

For groups with different latest start times, it is necessary to consider the lag placement of some rectangles, that is, the shortest start time of some rectangles is later than the shortest start time of the current rectangle. However, in the strip box, the space is scheduled to be shorter than the length of the current rectangle. Therefore, we design a lag pruning operator (LPO) to improve the search ability, avoid being pruned in advance, and increase the search effect. R_{ki} is set to be the current scheduling rectangle; h_{ki} and l_{ki} are the height and length, respectively; LSL_k is the latest starting length.

S_a is the set of all rectangles in packaging mode $A(\tilde{I})$, while S_{lag} is the set of rectangles in packaging mode $A(\tilde{I})$ where there is a rectangle R_{ki} , whose start time is greater than that of other rectangles $SL_{ki} > SL_{pj}$, and the latest start length is less than that of other rectangles $LSL_k < LSL_p$. $|S|$ is the number of rectangles in the set and the lag factor Lf is defined as follows:

$$Lf = |S_{lag}/S_a|, S_{lag} = \{R_{ki} | R_{ki}, R_{pj} \in A(\tilde{I}), (LSL_{ki} < LSL_{pj}) \cap (SL_{ki} > SL_{pj})\} \quad (24)$$

Figure 6 is a schematic diagram of the lag relation and Rectangles R_1 , R_2 , and R_3 in the previous section, whose latest start lengths are 0, 1, and 2, respectively.

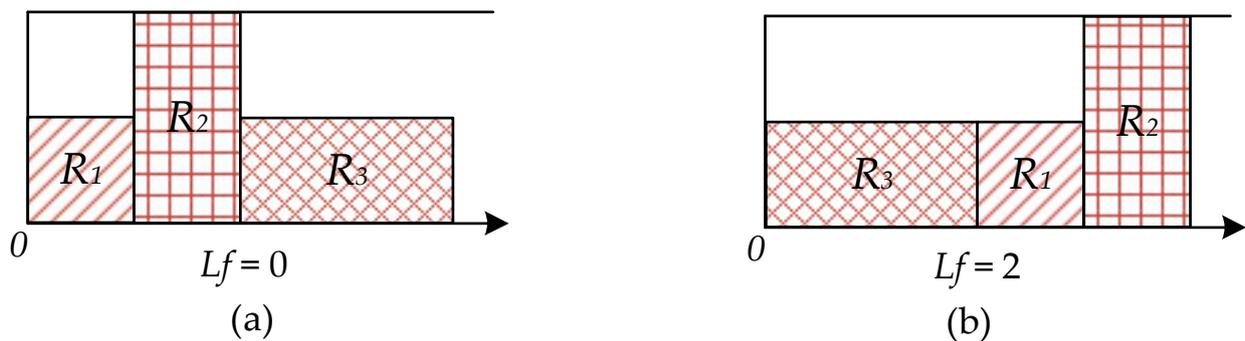


Figure 6. Non-lag and lags. (a) Non-lag; (b) lag.

Figure 6a shows that if the current scheduling rectangle is set to R_3 , the lag factor $Lf = 0$; if the current scheduling rectangle is set to R_2 , it is observed from Figure 6b that there are two rectangles with starting lengths longer than Rectangle R_2 ; thus, $Lf = 2/3$. In the algorithm, the lag pruning operator evaluates whether the sum of the lag factors of each rectangle is greater than the set lag limit (LL) under the current packing mode, and $LL \in (0,1)$. If it is greater than the lag limit, the pruning operation will be carried out.

3.4. Delay Equivalence and Dominance

In the branch-and-bound algorithm, equivalence and dominance relations are needed for pruning algorithmic spanning trees to reduce the computational cost. Relevant literature studies can be found in [22]. In this paper, equivalence and dominance are evaluated by the delay length as given by Equation (3).

3.5. Algorithm Flow

The operators introduced in this section will be used to generate the branch and bound reverse-order insert algorithm with the lag pruning operator (BB-RIA-LPO) algorithm. Table 3 shows the pseudocodes of the BB-RIA-LPO algorithm, where ks is the search depth, $count$ is the number of times that ks is assigned to 0, and $|R|$ is the number of all rectangles. Figure 7 shows the algorithm flowchart.

Table 3. Branch and bound reverse order insert algorithm with lag pruning operator (BB-RIA-LPO).

-
- ① Initialization, the loop begin, $ks = 0$, $count = 0$.
 - ② If the current strip box placement point queue is empty, then find all placement points and put them in the placement point queue.
 - ③ Have all placement points in placement point queue been scheduled? If yes, go to ④, backtracking; otherwise, set the current placement point as the scheduling point and go to ⑦.
 - ④ Go back to the previous layer, $ks = ks - 1$.
 - ⑤ Check if $ks = 0$, count the number of times that ks is assigned to 0 and record it in the variable $count$.
 - ⑥ If the $count < |R|$, then $count = count + 1$, and go to ②, the next loop; otherwise, stop and end the algorithm.
 - ⑦ Whether to arrange all rectangles depends on whether the rectangle queue is empty. If yes, go to ⑧; otherwise, go to ② for the next loop.
 - ⑧ Will the lag operator be set? If yes, check the lag. Otherwise, switch to ⑨.
 - ⑨ Pre-dispatch and check over the limit; if the height of the strip box is exceeded, go to ② for the next loop.
 - ⑩ Are the group constraints met? If not, go to ② for the next loop.
 - ⑪ Is equivalence and dominance satisfied? If not, go to ② for the next loop.
 - ⑫ Regarding dispatching rectangles, calculating object values L_{opt} and recording paths, is it at the bottom? If yes, go back to $ks = ks - 1$ and go to ② for the next loop; otherwise, search down, $ks = ks + 1$.
-

The two-dimensional strip packing grouping delay optimization problem (2SPGDP) and the corresponding latest start time LSL_{ki} , grouping constraint GHC_k , rectangle height h_{ki} , and length l_{ki} are based on group fairness that set the latest start length equal to the same group $LSL_{ki} = LSL_k$, ($i = 1, 2, \dots, m_k$). The partial solution space is generated by a deep search inverse spanning tree. Using the branch-and-bound method, the placement point and the dispatch rectangle are found. Additionally, they are put into the placement point queue and the rectangle queue and wait to be scheduled. After grouping the constraint operator, equivalence, and dominance operator pruning, the sum of all scheduling rectangle delays is calculated according to Equation (3).

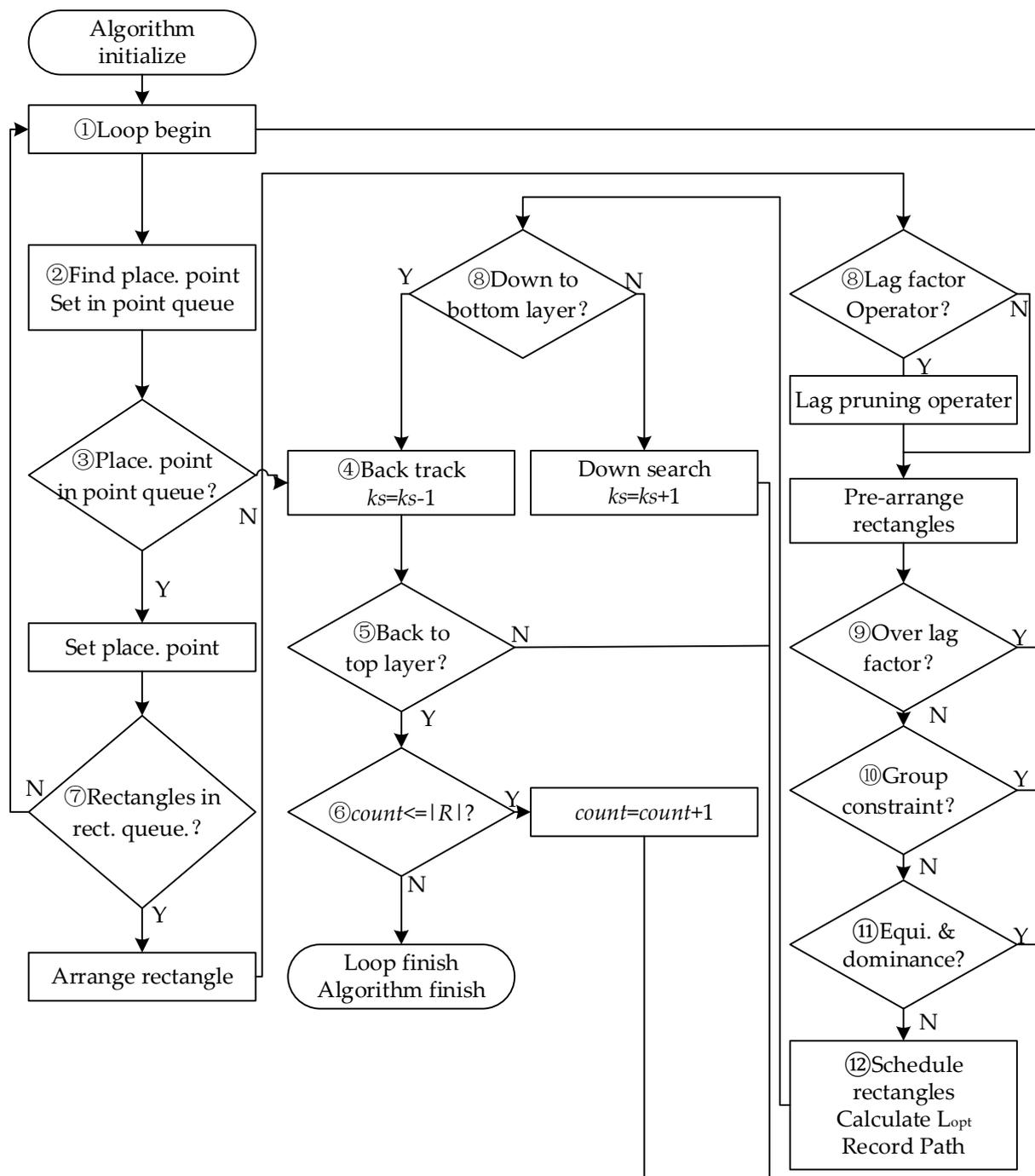


Figure 7. Flowchart of the branch-and-bound reverse-order insert algorithm with the lag pruning operator (BB-RIA-LPO).

4. Numerical Experiment

To test the performance of the algorithm, we divided this section into three subsections. In the first subsection, classical test instances were used; in the second subsection, improved grouping constraint test instances were used; and in the third subsection, the influence of the lag limit (LL) on the algorithm was studied. The algorithms we use include BL, BL0, NFDH, branch and bound (BB), and BB-RIA-LPO with lag factors. BB-RIA-LPO is abbreviated as the improved branch and bound (IBB). Among these, the BB algorithm is based on the rectangle combination length to perform equivalence and dominance pruning operations, while the IBB algorithm is based on the calculated delay (DL) to perform

equivalence and dominance pruning operations with the lag pruning operator. In this study, a Lenovo P340 graphics workstation, CPU Intel Core I9-10900/16G DDR4 memory, and MATLAB R2021b were used.

4.1. Performance of Classical Test Instances

In this subsection, we use classical test instances to evaluate the packing performance of the algorithm. All test instances are from the website of the European Operations Research Institute [23]. All of the grouping constraints $GHC_k = H$ and all the latest start lengths $LSL_{ki} = L_{best}$, which is the optimal length of the test instances, are used to evaluate whether our algorithm can solve the two-dimensional strip packing problem. The purpose of this subsection is to verify and compare the effectiveness of BB-RIA (IBB) on classical test instances. Because the goal is to minimize the makespan, the LPO operator is not used.

As observed from Table 4, the BB and IBB algorithms can obtain better-optimized lengths L_{opt} .

Table 4. Performance comparison of classical test instances.

Test	BL		NFDH		BB		IBB	
	L_{opt}	Time	L_{opt}	Time	L_{opt}	Time	L_{opt}	Time
J1	29	362	21	419	18	733	18	615
J2	24	46	18	13	22	1680	22	1640
D1	60	60	52	680	52	1701	52	1680
D2	54	40	47	15	45	855	45	792
D3	138	820	131	17	143	2380	130	2330
D4	233	46	245	15	236	1970	223	1920
Kendell	140	66	210	30	140	956	140	934

The J2 test intention may be caused by the BB and IBB algorithms pruning the best possible combination in the dominance and equivalence operator, resulting in the optimized length NFDH being superior to BL, BB, and IBB. In terms of calculating the time, BB and IBB take longer than BL and NFDH. In actual operation, BB and IBB are the same. The numbers of search nodes in J1, J2, D1, D2, D3, D4, and Kendell are 52, 102, 64, 44, 76, 76, and 28, respectively. This experiment shows that, although IBB takes a long time to calculate, it can obtain a better solution for classical test instances.

4.2. Performance of Improved Grouping Constraint Test Instances

In this subsection, we use improved grouping constraint test instances. The first six test instances are proposed by Berkey and Wang [24] for reference, as shown in Table 5.

The improvements are as follows: GHC_k is set as a variable, and LSL_k is set as a fixed value in the same group considering the fairness principle. It is used to evaluate whether the algorithm has good performance after the grouping constraints.

Type $k(k \in \{1,2,3,4\})$ generates rectangles of Type k with a probability of 0.7, and the remaining three groups generate other rectangles with a probability of 0.1. This method was proposed by Martello and Vigo [25], and it was found that $H = 100$. L_p is the maximum range of length and width of the generated rectangle. For example, L_p of T1 is $\max(H-2/3 H, 1/2 H-1) \approx 1/2 H$. S_{max} , S_{avg} , and S_{min} are the scores of BB and IBB algorithms in terms of maximum, average, and minimum values, which are calculated as follows:

$$DL_{UB} = \max(h_{ki}) \times \max(l_{ki}) \times \sum_{k=1}^g m_k / GHC_k \tag{25}$$

$$S_{max} = \max(DL_j) / DL_{UB}, j = 1, 2, \dots, N \tag{26}$$

$$S_{avg} = \sum_{j=1}^N DL_j / (DL_{UB} \times N) \tag{27}$$

$$S_{\min} = \min(DL_j) / DL_{UB}, j = 1, 2, \dots, N \quad (28)$$

Table 5. Grouping constraint test settings.

Test	Denote	Description
Class 1	C1	$H = 10, h_j$ and l_j at random in $[1,10]$
Class 2	C2	$H = 30, h_j$ and l_j at random in $[1,10]$
Class 3	C3	$H = 40, h_j$ and l_j at random in $[1,35]$
Class 4	C4	$H = 100, h_j$ and l_j at random in $[1,35]$
Class 5	C5	$H = 100, h_j$ and l_j at random in $[1,100]$
Class 6	C6	$H = 300, h_j$ and l_j at random in $[1,100]$
Type 1	T1	h_j at random in $[2/3 H, H]$; l_j at random in $[1, 1/2 H]$
Type 2	T2	h_j at random in $[1, 1/2 H]$; l_j at random in $[2/3 H, H]$
Type 3	T3	h_j at random in $[1/2 H, H]$; l_j at random in $[1/2 H, H]$
Type 4	T4	h_j at random in $[1, 1/2 H]$; l_j at random in $[1, 1/2 H]$

In Equations (25)–(28), N is the number of rectangles, $N = \{20,40,60,80,100\}$, and DL_j is the total delay in the j -th rectangle. According to Equation (14), we can determine the relaxation upper bound DL_{UB} of the problem, which is the possible maximum area sum of all rectangles, divided by the grouping constraint GHC_k . Then, DL_{UB} can be taken as the relative value to calculate the scores S_{max} , S_{avg} , and S_{min} proportional to the delay. A smaller score indicates a smaller delay (DL) and better effectiveness of the algorithm. The success rate R_s is defined as the number of experiments (out of 100 experiments) in which the algorithm reaches the optimal value of all algorithms. If two or more algorithms reach the optimal value, they will be included. In addition, based on Equation (11) in the previous section, the maximum upper boundary of delay is approximately the area of the rectangle. Figure 8 shows the comparison of the BL/BL0/NFDH/BB/IBB algorithms.

The experimental conditions are the same as those described in this subsection. In the comparison of delays in Figure 8a, it is observed that BB and IBB have the lowest delay and the best effect, while NFDH has the worst effect. In all our other experiments, the effect of BFDH/FFDH was not as good as that of NFDH, so the data for these experiments are not shown. Figure 8b shows a comparison of the calculation times. It is observed that IBB has the longest calculation time, which is even longer by several orders of magnitude. Therefore, we adopt the logarithmic Y coordinate system, and the order of calculation time is $T_{IBB} > T_{BB} > T_{BL} \approx T_{BL0} > T_{NFDH}$. Figure 8c shows the comparison of search nodes between BB and IBB. It is observed that the number of search nodes of IBB exceeds 60 when the total number of rectangles is greater than 60, that is, the number of groups $g > 6$ will exceed BB and grow rapidly. Figure 8d shows the 95% delay spacing diagram. The NFDH algorithm is the most divergent in solving, followed by BL and BL0. BB and IBB are the most concentrated, and their average values are the smallest.

Table 6 compares the performance of the algorithms for the improved grouping constraint test instances. From the point of view of the optimal number of R_s , BB and IBB are better than BL0. The branch-and-bound method can search more space so that the solving effectiveness is better for BB and IBB than for BL0. Moreover, since NFDH, BFDH, and FFDH are not as effective as BL0, they are not listed in the results. BB outperforms IBB when the number of groups g is less than 6, but IBB performs better when the number of groups g is greater than 6 because IBB uses delay as the judgement basis for controlling the equivalence pruning. When the number of groups g is large, the advantage is clear. In addition, BB is inferior to IBB when $g > 6$, except at T3. The R_s of BB rises and the R_s of IBB declines in T4.

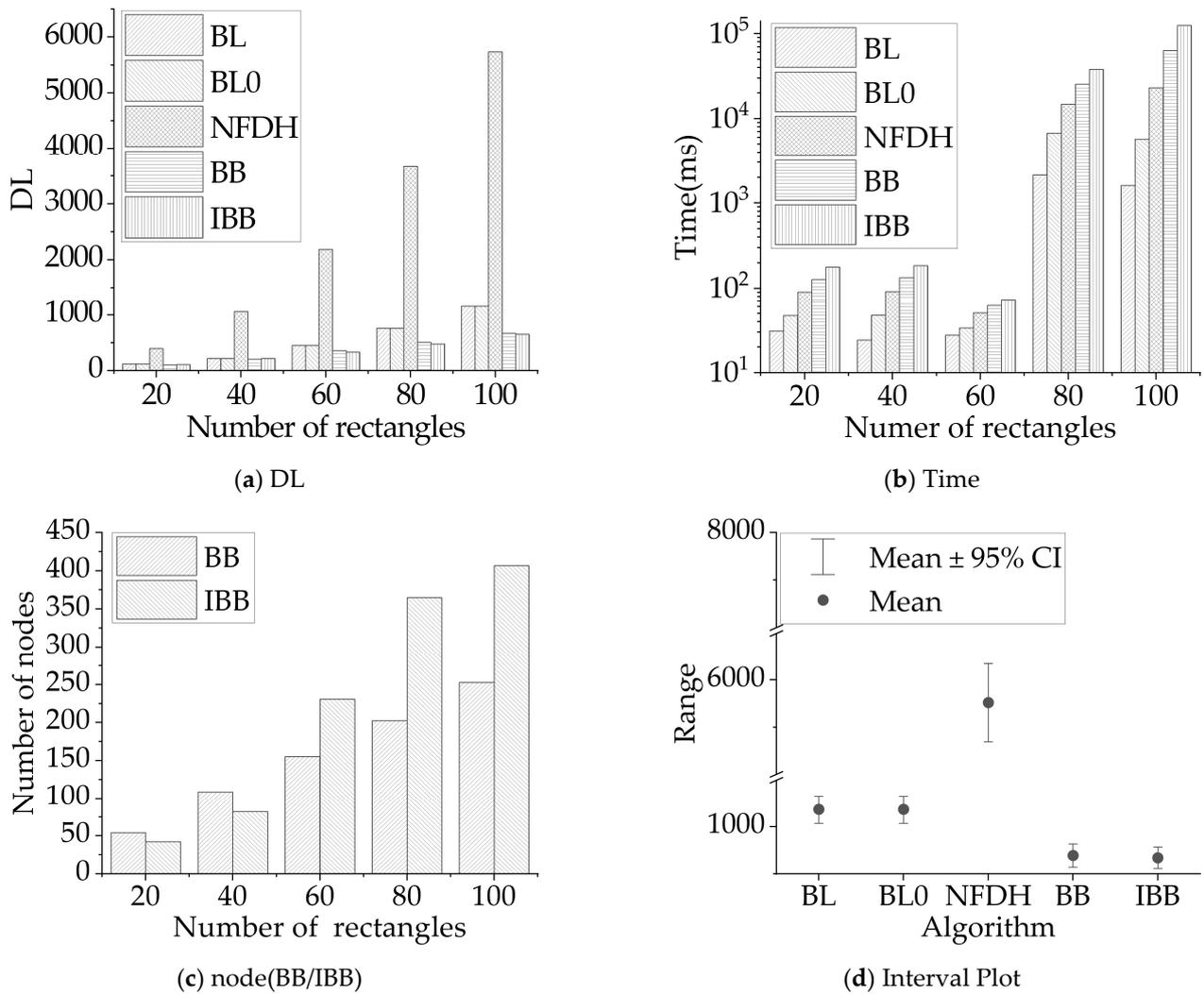


Figure 8. Comparison of BL/BL0/NFDH/BB/IBB algorithms (Test C1).

It remains to be studied whether increasing the number of groups g can make the optimal times of BB equal to or even surpass those of IBB. In R_s , the number of groups $g > 6$, and the IBB advantage is more pronounced, achieving a maximum improvement of 25% relative to the BB algorithm (Test T5, 80 rectangles).

In $S_{min/avg}/S_{max}$, the values of C2, C4, and C6 are larger than C1, C3, and C5 because a larger H/L_p of the problem corresponds to a smaller delay. For larger H , more space in the problem can be scheduled, and a smaller $S_{min}/S_{avg}/S_{max}$ is obtained. At the same time, it is also observed that in C1, C3, and C5, the S_{min} of IBB is slightly better than that of BB, but the IBB on S_{max} is partially worse than that of BB, indicating that under the condition of $L_p \approx H$, the solution of IBB is more divergent than that of BB. In T1~T4, S_{min} can be optimized for T1/T2/T4, while T3 cannot be optimized. In $S_{min}/S_{avg}/S_{max}$, BB and IBB have little difference in solving ability, IBB is relatively divergent, and IBB's optimal value is approximately 6% higher than that of BB (Test T4 80).

Table 6. Performance comparison of algorithms for improved grouping constraint test instances.

Test		R_S			S_{min}		S_{avg}		S_{max}	
		BL0	BB	IBB	BB	IBB	BB	IBB	BB	IBB
C1	20	1	100	88	7	7	15	16	26	33
	40	0	100	82	12	12	31	32	56	56
	60	0	61	85	26	25	51	51	76	76
	80	0	71	75	39	39	66	67	96	103
	100	0	71	82	49	49	84	84	126	126
C2	20	26	77	66	1	1	5	5	13	13
	40	58	42	36	1	1	4	4	12	12
	60	18	51	69	1	1	4	4	12	12
	80	16	58	71	1	1	4	4	12	12
	100	9	58	78	1	1	5	4	14	13
C3	20	3	99	87	3	3	11	11	27	27
	40	0	100	76	8	8	24	25	46	46
	60	0	64	81	16	15	36	35	59	59
	80	0	71	80	24	24	48	48	76	76
	100	0	76	75	35	35	60	60	90	91
C4	20	37	63	55	0	0	4	4	13	13
	40	46	54	43	0	0	4	4	9	10
	60	16	54	70	0	0	5	5	13	12
	80	15	61	69	1	1	5	5	16	19
	100	3	72	80	1	1	6	6	16	14
C5	20	10	97	86	5	5	25	25	41	42
	40	0	100	84	11	11	25	25	48	49
	60	0	55	88	20	19	41	41	66	64
	80	0	63	79	28	27	52	52	79	82
	100	0	70	78	35	35	67	67	98	106
C6	20	39	64	58	0	0	4	4	12	12
	40	68	32	27	0	0	4	4	12	12
	60	29	49	60	0	0	4	4	11	9
	80	29	52	52	1	1	4	4	15	17
	100	14	63	67	0	0	5	5	13	22
T1	20	10	97	86	5	5	25	25	41	42
	40	0	100	90	25	25	58	59	103	103
	60	0	58	84	50	50	93	92	138	147
	80	0	59	81	77	77	122	122	165	163
	100	0	58	83	109	108	159	159	206	238
T2	20	5	94	71	3	3	21	22	39	39
	40	0	99	85	25	25	51	51	75	75
	60	0	64	78	38	33	78	77	112	112
	80	0	69	84	60	55	107	106	158	158
	100	0	70	88	77	77	136	135	195	195
T3	20	4	99	81	13	13	34	35	55	55
	40	1	99	81	45	45	73	74	105	105
	60	0	67	62	85	82	116	117	157	163
	80	0	68	68	118	118	154	154	187	187
	100	0	64	75	157	157	192	192	229	230
T4	20	3	98	83	0	0	11	11	39	39
	40	0	100	72	2	2	27	28	99	99
	60	0	66	87	6	6	43	42	115	115
	80	0	76	86	16	10	58	58	118	118
	100	0	76	80	18	18	79	78	158	158

4.3. Performance with Lag Factor

This subsection mainly studies the influence of the change in the lag limit (LL) on the algorithm. $LSL_k = kL_r$ is set in the experiment. L_r is the upper limit of the generation length of test Class 1. The lag limit $LL \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$ is set. To save experimental time, the number of groups $g = 8$, the number of rectangles $m_k = 5$ in each group, and the other conditions are the same as in this section. Figure 9 shows the influence of LL on the delay/time/number of search nodes, and Figure 9a shows the relationship between the delay and LL .

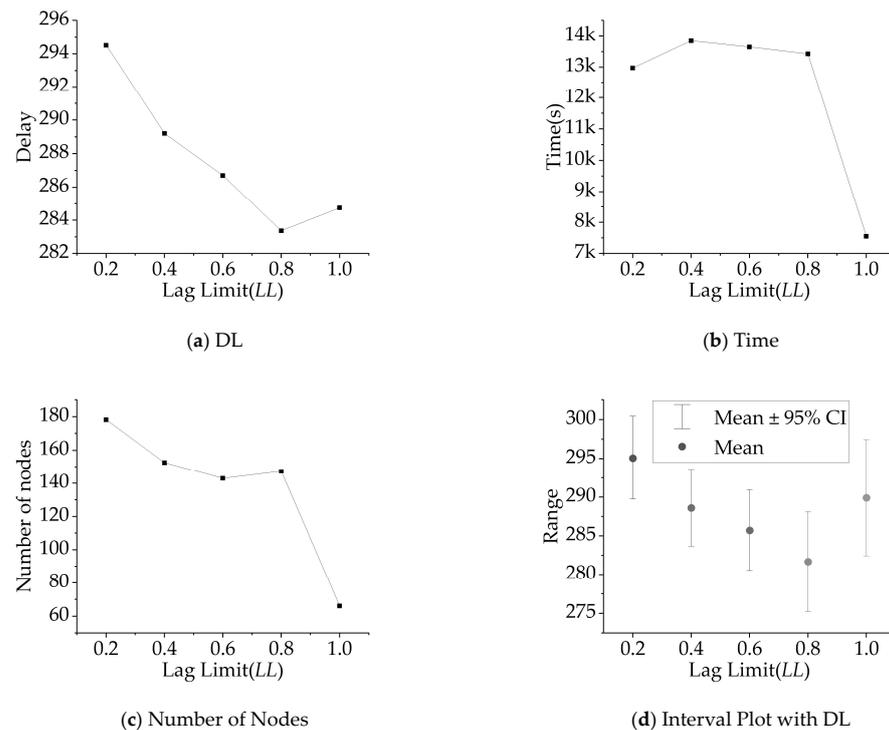


Figure 9. Influence of delay factors on the delay/time/number of the search nodes.

The delay decreases with increasing LL , reaching a minimum when $LL = 0.8$, and then rising slightly. Figure 9b shows the relationship between the calculation time and LL . The calculation time is relatively stable from $LL = 0.2$ to 0.8 and decreases rapidly when $LL = 1.0$. Figure 9c shows the relationship between the number of search nodes and LL . The number of search nodes decreases slowly with increasing LL and decreases rapidly after 0.8 . Figure 9d shows the 95% confidence interval diagram. It is observed that the overall confidence interval has roughly the same length, but the average value decreases with the increase in LL , reaching the lowest value at 0.8 and then rising at 1.0 . We also studied grouping constraints and delay, but the results were random, and there was no specific experimental conclusion.

4.4. Global Sensitivity Analysis

In this section, we perform a global sensitivity analysis (GSA) on the BB-RIA-LPO algorithm [26]. The analysis program used is provided by Flavio, which generates a Sobol quasi random set and Fourier amplitude sensitivity test (FAST) analysis. The test parameters are the lag limit (LL), number of groups (g), number of each group (m_k), and the maximum range of width and height (L_p). Their parameter ranges are $[0.6,1]$, $[2,10]$, $[10,20]$ and $[2,8]$, respectively.

Table 7 shows the first-order global sensitivity coefficients of the FAST algorithm. It can be seen from the table that the number of groups (g) and the number of each group (m_k) are relatively large for both the uniform random distribution and the Sobol quasi random

set, which indicates that they have a great influence on the BB-RIA-LPO algorithm and confirms our previous discussion about group $g > 6$.

Table 7. Performance comparison of algorithms for improved grouping constraint test instances.

	LL	g	m_k	L_p
Uniform	0.0039	0.2232	0.0240	0.4028
Sobol	0.0059	0.2219	0.0214	0.3987

5. Rural Portfolio Credit Case

This section is a case simulated according to the actual “whole village credit” project: Under the background of implementing the “rural revitalization” policy, Bank H conducts the whole village credit business for Village L, with a total amount of 600 million yuan; credit is divided into organizations and villagers’ individuals. Organization loans are usually used for production investment or project construction, so there is a large loan amount, a long loan term, and a low requirement for the latest loan obtaining time. However, villagers’ individual loans are usually used for personal consumption and personal business. The use of funds is characterized by small loan amounts and short loan terms, so they hope to obtain loans in a short time. Because of the large number, in this example, for the convenience of demonstration, multiple personal consumption credits and personal business loans are combined, their amounts are summed, their terms are set as their average values, and the total amount is calculated as a whole for scheduling. Table 8 provides detailed information on the case.

Table 8. Performance comparison of algorithms for improved grouping constraint test instances.

G_k	i	Project	Amount	Total Loan Amount	Loan Term	Latest Obtaining Time	Village Credit
1	1	Automatic irrigation equipment for planting	1	30	3	0.25	300
1	2	Treatment of pig breeding wastes	1	50	5	0.25	
1	3	Personal business loans	33	65	2.5	0.1	
1	4	Personal consumption loans	142	92	1.5	0.1	
2	1	Meat intensive processing production line	1	45	3	0.25	300
2	2	Egg incubator equipment	1	32	2	0.25	
2	3	Personal business loans	26	53	2.2	0.1	
2	4	Personal consumption loans	123	59	1.8	0.1	
3	1	Supporting facilities of “Hot Spring Village” featured health care center	1	100	8	0.25	300
3	2	Greenhouse planting facilities	1	88	5	0.25	
3	3	Personal business loans	18	44	2.8	0.1	
3	4	Personal consumption loans	102	71	1.6	0.1	
4	1	Attached aquaculture equipment for abalone aquaculture	1	46	3	0.25	300
4	2	Agricultural Products Wholesale Center Project	1	120	6	0.25	
4	3	Personal business loans	16	39	2	0.1	
4	4	Personal consumption loans	99	99	1.1	0.1	
5	1	Land fertility improvement project of fruit and vegetable base	1	69	2	0.25	300
5	2	Low temperature finished grain storage construction project	1	110	8	0.25	
5	3	Personal business loans	18	45	2.4	0.1	
5	4	Personal consumption loans	102	77	1.3	0.1	
6	1	Irrigation and drainage pump station renovation project	1	50	4	0.25	300
6	2	Photovoltaic power generation energy construction project	1	100	6	0.25	
6	3	Personal business loans	16	45	2.6	0.1	
6	4	Personal consumption loans	99	1020	1.9	0.1	

Figure 10 shows the Gantt chart of case scheduling. The mark R_{ki} indicates the loan of the k -th village and the i -th item. All villagers' loans have been dispatched, among which R_{12} , R_{41} , and R_{51} , the villagers who received loans later, received loans for 1.2, 1.2, and 1.4 years. This is because the total amount of loan demand of the fourth and fifth villagers exceeds the credit amount of the village, so among them, the loan amount is large, and R_{41} and R_{51} with a long loan term will be delayed. For this case, the BL, BL0, and BB-RIA-LPO algorithms generate total delays of 10.7, 4.4, and 3.8 years, respectively. The new algorithm reduces the total delay by 15.79%.

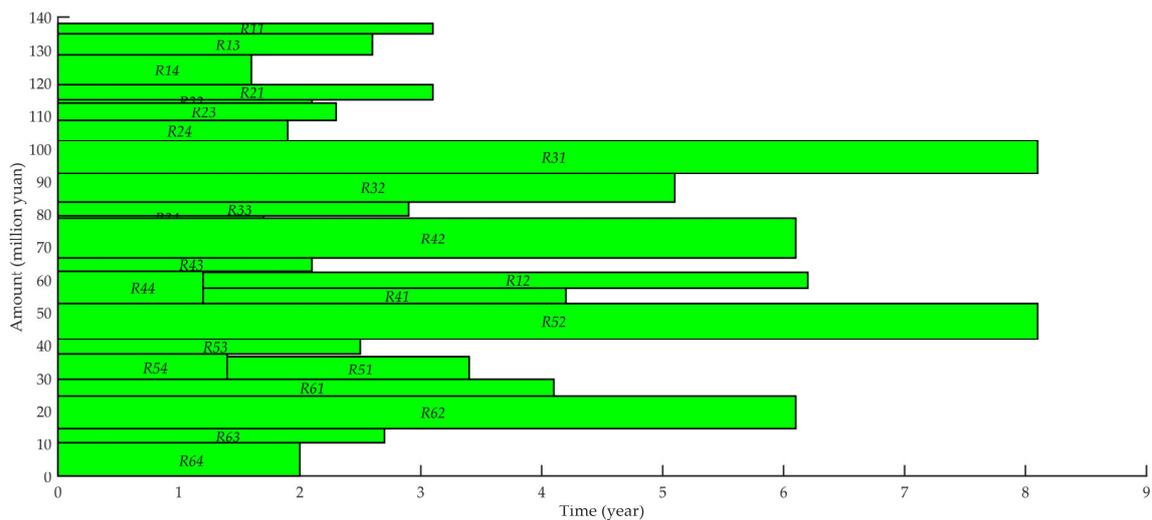


Figure 10. Gantt chart of case scheduling.

6. Conclusions

In this paper, we study the solving algorithm of the portfolio credit-granting problem in the field of management. First, the portfolio credit granting problem in the field of system management is abstracted into the combinatorial (grouping) constraint two-dimensional packing delay optimization problem. Second, a mathematical model of two-dimensional packing delay optimization based on combinatorial constraints, that is, 2SPGDP, is established, and its upper bounds, lower bounds, and delays are studied. Third, BB-RIA, ISS, and LPO were used to solve the two-dimensional packing delay optimization problem.

Numerical experiments show that the method can solve the classical test instances correctly. In the two-dimensional packing problem, the improved method has better solution quality and success rate than the bottom-left algorithm (BL) when the number of groups is greater than 6. Compared with the classical branch and bound algorithm, the maximum success rate can be improved by 6–25%, and the total delay can be reduced by 6%. The lag limit of the improved algorithm can obtain the best solution quality at $LL = 0.8$. In the case of rural portfolio credit, the algorithm reduces the delay by 15.79%.

This study still has some limitations and needs further improvement. In terms of the algorithm, the relationship between the lag factor and grouping latest start length (LSL) setting has not been clearly studied. The algorithm with group number $g < 6$ and the solution time needs to be shortened. Second, from the actual problem of rural portfolio credit, a more rigorous risk mechanism needs to be added, and larger examples need to be verified.

In brief, the 2SPGDP model proposed in this paper can be used as a reference for the problem of combination constraint and delay optimization. The BB-RIA-LPO algorithm can reduce the delay and obtain a better solution. The algorithm is applied to the rural portfolio credit-scheduling problem to improve efficiency and better schedule preferential loans to villagers through grouping constraints. As rural loans are unsecured preferential loans with the attribute of public service, it is one way to strengthen the risk mitigation

of rural commercial banks to adopt whole village credit and decentralized management, which is conducive to rural revitalization.

Author Contributions: Conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, writing—original draft preparation, writing—review and editing, H.H.; writing—review and editing, visualization, supervision, project administration, Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China under Grant 62176102.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data were obtained from the European Operations Research Institute.

Conflicts of Interest: The author declares no conflict of interest.

References

- Chen, X.X.; Zheng, H.R. Research on Rural Inclusive Financial Credit Service from the Perspective of Satisfaction-Take Fujian sample of inclusive financial reform pilot area as an example. *Financ. Theory Pract.* **2022**, *9*, 69–80.
- Wang, Q. Research on the Optimization of the Whole Village Credit Business of CC Rural Commercial Bank. Master's Thesis, Jilin University, Changchun, China, 2022. [[CrossRef](#)]
- Li, N. Study on the Farmer Loan Management Strategy of "Whole Village Credit" in W Branch. Master's Thesis, Dalian University of Technology, Dalian, China, 2021. [[CrossRef](#)]
- Gao, L. Design of Credit Scheme for Rural Whole Village of Huanghe Bank. Master's Thesis, Xi'an University of Technology, Xi'an, China, 2021. [[CrossRef](#)]
- Chen, W. Research on the Whole Village Credit Granting of Small Loans to Farmers in WH Rural Credit Union. Master's Thesis, Zhengzhou University, Zhengzhou, China, 2021. [[CrossRef](#)]
- Liu, Y. Research on Problems and Countermeasures of JD Rural Commercial Bank's "Whole Village Credit". Master's Thesis, Yangzhou University, Yangzhou, China, 2019. [[CrossRef](#)]
- Liu, M.; Tong, X.; Dai, Y. Recent developments in algorithms for packing problem. *Math. Num. Sinica* **2016**, *38*, 257–280.
- Cid-Garcia, N.M.; Rios-Solis, Y.A. Exact solutions for the 2d-strip packing problem using the positions and covering methodology. *PLoS ONE* **2021**, *16*, e0245267. [[CrossRef](#)] [[PubMed](#)]
- Qi, X.Y.; Zheng, Q.X.; Li, M.; Guo, T.J. The Model with Left Bottom Corner Coordinates for 2D Rectangular Strip Packing Problem. In Proceedings of the 2017 International Conference on Wireless Communications, Networking and Applications, Shenzhen, China, 20–22 October 2017; pp. 172–176.
- Souza Queiroz, L.R.D.; Andretta, M. A branch-and-cut algorithm for the irregular strip packing problem with uncertain demands. *Int. Trans. Oper. Res.* **2022**, *29*, 3486–3513. [[CrossRef](#)]
- Oviedo-Salas, E.; Terán-Villanueva, J.D.; Ibarra-Martínez, S.; Santiago-Pineda, A.; Ponce-Flores, M.P.; Laria-Menchaca, J.; Castán-Rocha, J.A.; Treviño-Berrones, M.G. GRASP Optimization for the Strip Packing Problem with Flags, Waste Functions, and an Improved Restricted Candidate List. *Appl. Sci.* **2022**, *12*, 1965. [[CrossRef](#)]
- Wei, L.; Hu, Q.; Leung, S.C.; Zhang, N. An improved skyline based heuristic for the 2D strip packing problem and its efficient implementation. *Comput. Oper. Res.* **2017**, *80*, 113–127. [[CrossRef](#)]
- Chen, Z.; Chen, J.L. An Effective Corner Increment-Based Algorithm for the Two-Dimensional Strip Packing Problem. *IEEE Access* **2018**, *6*, 72906–72924. [[CrossRef](#)]
- Chen, M.; Li, K.; Zhang, D.; Zheng, L.; Fu, X. Hierarchical Search-Embedded Hybrid Heuristic Algorithm for Two-Dimensional Strip Packing Problem. *IEEE Access* **2019**, *7*, 179086–179103. [[CrossRef](#)]
- Li, Y.-B.; Sang, H.-B.; Xiong, X.; Li, Y.-R. An Improved Adaptive Genetic Algorithm for Two-Dimensional Rectangular Packing Problem. *Appl. Sci.* **2021**, *11*, 413. [[CrossRef](#)]
- Mondal, S.; Tsourdos, A. Two-Dimensional Quantum Genetic Algorithm: Application to Task Allocation Problem. *Sensors* **2021**, *21*, 1251. [[CrossRef](#)] [[PubMed](#)]
- Neuenfeldt Júnior, A.; Francescato, M.; Stieler, G.; Disconzi, D. A Multi-label Transformation Framework for the Rectangular 2D Strip-Packing Problem. *Manag. Prod. Eng. Rev.* **2021**, *14*, 27–37.
- Zheng, S.; Yang, Z.; He, Z.; Wang, N.; Chu, C.; Yu, H. Hybrid simulated annealing and reduced variable neighbourhood search for an aircraft scheduling and parking problem. *Int. J. Prod. Res.* **2019**, *58*, 2626–2646. [[CrossRef](#)]
- Vega-Mejia, C.A.; Montoya-Torres, J.R.; Islam, S.M.N. Consideration of triple bottom line objectives for sustainability in the optimization of vehicle routing and loading operations: A systematic literature review. *Ann. Oper. Res.* **2019**, *273*, 311–375. [[CrossRef](#)]

20. Júnior, A.N.; Silva, E.; Francescatto, M.; Rosa, C.B.; Siluk, J. The rectangular two-dimensional strip packing problem real-life practical constraints: A bibliometric overview. *Comput. Oper. Res.* **2022**, *137*, 105521. [[CrossRef](#)]
21. Parker, R.G. *Deterministic Scheduling Theory*, 1st ed.; Chapman & Hall: Boca Raton, FL, USA, 1996; pp. 25–35.
22. Scheithauer, G. *Introduction to Cutting and Packing Optimization- Problems, Modeling Approaches, Solution Methods*, 1st ed.; Springer: Basel, Switzerland, 2018; pp. 183–225.
23. Test Problems for 2D Rectangular Strip Packing: Benchmark Problems in Literature, the Association of European Operational Research Societies. Available online: <https://www.euro-online.org/websites/esicup/data-sets/#1535972088188-55fb7640-4228> (accessed on 1 October 2021).
24. Berkey, J.O.; Wang, P.Y. Two dimensional finite bin packing algorithms. *J. Oper. Res. Soc.* **1987**, *38*, 423–429. [[CrossRef](#)]
25. Martello, S.; Vigo, D. Exact solution of the two-dimensional finite bin packing problem. *Manag. Sci.* **1998**, *44*, 388–399. [[CrossRef](#)]
26. Cannavo, F. Sensitivity analysis for volcanic source modeling quality assessment and model selection. *Comput. Geosci.* **2012**, *44*, 52–59. [[CrossRef](#)]