Hindawi Publishing Corporation
Discrete Dynamics in Nature and Society
Volume 2015, Article ID 841637, 6 pages
http://dx.doi.org/10.1155/2015/841637

Research Article

A Hybrid Heuristic Algorithm for Ship Block Construction Space Scheduling Problem

Shicheng Hu,[1] Tengjiao Liu,[1] Song Wang,[1] Yonggui Kao,[2] and Xuedong Sun[3]

[1] *School of Economics and Management, Harbin Institute of Technology, Weihai 264209, China*
[2] *Department of Mathematics, Harbin Institute of Technology, Weihai 264209, China*
[3] *School of Software, Sun Yat-sen University, Guangzhou 510275, China*

Correspondence should be addressed to Xuedong Sun; sunxdys@163.com

Received 27 December 2014; Accepted 17 January 2015

Academic Editor: Shuenn-Ren Cheng

Ship block construction space is an important bottleneck resource in the process of shipbuilding, so the production scheduling optimization is a key technology to improve the efficiency of shipbuilding. With respect to ship block construction space scheduling problem, a hybrid heuristic algorithm is proposed in this paper. Firstly, Bottom-Left-Fill (BLF) process is introduced. Next, an initial solution is obtained by guiding the sorting process with corners. Then on the basis of the initial solution, the simulated annealing arithmetic (SA) is used to improve the solution by offering a possibility to accept worse neighbor solutions in order to escape from local optimum. Finally, the simulation experiments are conducted to verify the effectiveness of the algorithm.

1. Introduction

Space is the key resource in ship block construction process. How to minimize the makespan of the project under space resource and precedence constraints is a complicated scheduling problem. As for this problem, two related problems are involved: resource constrained project scheduling problem (RCPSP) and bin packing problem.

RCPSP can be described as a problem which should be scheduled under the limits of technology and other constraints to meet the objective of a project [1]. Generally, the goal is to get the shortest project makespan under the available resources and precedence constraints. The methods can be classified into two categories: exact methods and heuristic methods. Hartmann [2] puts forward that RCPSP belongs to NP-hard problem because it is always used to extend machine scheduling problem [3, 4]. So with the augment of problem scale, the computational complexity will increase rapidly. Many researchers have used exact algorithm to solve RCPSP [5, 6]; however, most of them proposed that exact algorithm is not feasible in reality. Priority rules proposed by Kelley [7] indicated that RCPSP can be solved by heuristic algorithms. Liu and Wang [8] tried to reduce the project makespan by heuristic algorithms and achieved good results. Bhaskar et al. [9] utilized parallel methods and priority rules to solve RCPSP with fuzzy activity times. Lee et al. [10] proposed a ship block construction space scheduling problem and described this problem theoretically. Koh et al. [11] solved the scheduling problem in shipbuilding company by heuristic algorithm.

The bin packing problem is putting more boxes into a limited bin in order to minimum the height. This problem can be classified into two categories, two-dimensional and three-dimensional problems. For the former one, researchers tend to solve bin packing problems by heuristic methods. They are Bottom-Left (BL) algorithm [12], Bottom-Left-Fill (BLF) algorithm [13]. In addition, Belov et al. [14] considered adapting one-dimensional problem for solving two-dimensional problems. Chan et al. [15] tried to solve two-dimensional problems by heuristics with stochastic neighborhood structures. For three-dimensional problems, the most popular 3BF [16], proposed by Silvano Martello in 2000, figure out the problem of how to choose the most suitable cubes. Alvarez-Valdes et al. [17] used a GRASP/Path relinking algorithm to solve multiple bin-size bin packing problems. Liao and Hsu [18] found new lower bounds to improve the efficiency of three-dimensional problems.

This paper is organized as follows. After introduction, Section 2 presents a mathematical model. Section 3 gives the hybrid heuristic algorithm for this problem. In Section 4, we conduct simulation experiments to verify the effectiveness of the algorithm. Section 5 proposes general conclusions.

2. Ship Block Construction Space Scheduling Problem

Ship block construction space scheduling problem can be described as a project which includes n activities $N = \{1, 2, \ldots, n\}$; place is required to process activities. A activity can be defined as $a_{ik} = e_{ik} \times c_{ik}$, where e_{ik} and c_{ik} represent the length and width of place k which activity i needs. The duration of activity i is t_i. I_i and H_i are the start and finish time of activity i. During the project, there are m places $M = \{1, 2, \ldots, m\}$ which can be defined as $B_k = E_k \times C_k$.

During the project, every activity is under precedence constraints, and we propose that P_j is the predecessors of activity j. So j can not be started if any one of its predecessors in P_j has not been finished. We assume that the start time of the whole project is 0. For the convenience of modeling, we also introduce two dummy nodes: activity 0 and $n+1$. They do not need time and space. 0 is the predecessor of all the activities in the project; meanwhile, $n+1$ is the successor of all the activities. So I_{n+1} is the makespan of the project. In addition, we regard an activity as a cube which is represented as $z_{ik} = e_{ik} \times c_{ik} \times t_i$. All the activities can be rotated in horizontal with 90 degree, and $r_i = 1$ and 0, indicating that activity i is rotated and not, respectively. The model of this problem is shown in what follows:

$$\text{Minimize: } I_{n+1}, \tag{1}$$

$$I_i + t_i \le I_j, \quad \forall i \in P_j, \tag{2}$$

$$z_{ik} \cap z_{jk} = \varphi, \quad \forall i, j \in N,\ i < j,\ \forall k \in M, \tag{3}$$

$$\left[(1 - r_i)\, w_{ik} + r_i e_{ik}\right] \le \left[(1 - r_i)\, c_{ik} + r_i e_{ik}\right] + x_i, \tag{4}$$

$$\left[(1 - r_i)\, c_{ik} + r_i e_{ik}\right] + x_i \le E_k, \tag{5}$$

$$\left[(1 - r_i)\, e_{ik} + r_i c_{ik}\right] \le \left[(1 - r_i)\, e_{ik} + r_i c_{ik}\right] + y_i, \tag{6}$$

$$\left[(1 - r_i)\, e_{ik} + r_i c_{ik}\right] + y_i \le C_k, \tag{7}$$

$$r_i \in \{0, 1\}. \tag{8}$$

In the model, formula (1) is the objective of the problem, (2) proposes the precedence constraints, and (3) means that two cubes cannot overlap. Formulae (4)–(7) denote that each cube should be completed within available place. The cube can be rotated horizontally and 0 and 1 represent if the cube is rotated, as formula (8) has shown.

3. A Hybrid Heuristic Algorithm

3.1. Initial Solution Method. In this paper, we apply BLF to get the initial solution. BLF, presented by Chazelle [13] in 1983, belongs to heuristic algorithm. In this algorithm, the method of placing cubes is determined by corners [19].

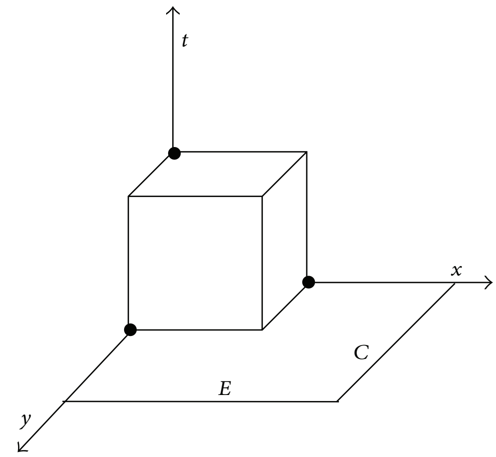


Figure 1: Corners.

3.1.1. Corner. In BLF, it is important to find corners to place the activity. Firstly, we will try the lowest and leftmost point (lowest point first); if this placement can match the activity, then place it in the position and update corners; otherwise, try next point until a corner is found. The corners can be represented as $\{(x, y, t), (e, c, h)\}$, where (x, y, t) indicates the location of the corners and (e, c, h) represents the available space of the point in x-y-z dimensions. See Figure 1; E and C denote the length and width of the place and t represents the duration of the project.

3.1.2. Bottom-Left-Fill Algorithm. In this paper, we represent the activities as cubes whose length, width, and height are limited. For these cubes, the length and width denote the length and width of the place they need, and the height represents the time they need. Meanwhile, we regard the unrestricted height as timeline t. When all activities have been placed, the total height equals the duration of the project. In the process, we find activities that can be placed at each time point first and then consider the sequence of these activities. We place the activity with larger bottom area first; if equal, choose the higher height; if still equal, select the one with the longer length. We describe the algorithm as follows.

(1) We embed the place (the platform in ship block construction space) in a three-dimensional coordinate system, put the bottom-left-rear point on the origin (0, 0, 0), and put length, width, and height on x, y, and t axis, respectively. E and C represent length and width of the place.

(2) Set 5 sets: A_n, C_n, D_n, I, and B. A_n is a set of activities which have been scheduled but not yet completed. C_n is a set of activities which have been scheduled and already completed. D_n is a set of activities which can be scheduled but have not been scheduled yet. I is a set of corners, while B is a set of all activities in input order.

(3) Move the activities which can be scheduled from B to D_n and then sort these activities: firstly, interchange the length and width of the activities so that $e_i \geq c_i$ can be met for any activity b_i; then sort the activities in D_n by placement policy mentioned above, and the first activity should be scheduled in D_n after being sorted. Detect the corners in I until the first eligible corner is found by t-y-x rule, which means that the smaller t among the corners is better. Then, consider the smaller y. Finally, select the smaller x. Put the bottom-left-rear point of the activity on the selected corner, and the new eligible corners will be new elements of I; meanwhile delete the corner which is used this time from I. Record the activities and their corners sequentially.

(4) If an activity in A_n has been completed, move it from A_n to C_n and then move its nearest successor from B to D_n. Sort activities in D_n by rules mentioned in (3). Repeat (3) and (4) until all activities are completed.

(5) Select the higher t of all corners as the duration of the project.

3.2. Optimization Solution Method. Proposed by Steinbrunn et al. [20], simulated annealing (SA) is a metaheuristic algorithm which offers a possibility to accept worse neighbor solutions to escape from local optimum. Until 1983, Kirkpatrick et al. [21] transformed this idea to SA arithmetic and applied it to traveling salesman problem successfully.

Chan et al. [22] develop a hybrid algorithm which gleans the ideas both from tabu search and sample sort simulated annealing to solve distributed scheduling problem. Bouleimen and Lecocq [23] implemented SA to RCPSP and MRCPSP. Yannibelli and Amandi [24] combined SA and other algorithms to solve a multiobjective project scheduling problem. In SA arithmetic, we describe the neighborhood as the modification of the order of two activities which can be scheduled at the same time. d_t represents temperature decreasing rate, L denotes the initial value of neighborhood, d_L represents the neighbors that are generated once the temperature drops, f denotes current duration, f_{best} denotes the shortest duration till now. For our ship block construction space scheduling problem, the SA can be described as follows.

(1) Initialization: the initial and final temperature is S_t and E_t, respectively. The algorithm iterates from initial solution B. The value of initial evaluation function is $f = 3D - PS(B)$, while the evaluation function value of initial optimal solution $B_{\text{best}} = B$ is $f_{\text{best}} = f$.

(2) t denotes the current temperature and L_t denotes the length of current neighborhood. Initially $t = S_t$, $L_t = L$.

(3) $j = 1$.

(4) Select a new solution B' from the neighborhood $N(B)$ of B.

(5) Calculate the incremental $d_f = f - f'$.

(6) If $d_f > 0$, accept B' as a new solution, $f = f'$, $B = B'$; if $f' < f_{\text{best}}$, then $f_{\text{best}} = f'$, $B_{\text{best}} = B'$. Else accept B' as a new solution with a probability $P = \exp(10 * d_f/T)$.

(7) $j = j + 1$, if $j \leq L_t$, go to step (4); else go to step (8).

(8) $L_t = d_L$, $t^* = d_t$, and compare t and E_t. If $t \geq E_t$, go to step (3); else the termination condition is met and output current f_{best} and B_{best}; end the program.

Table 1: The parameters of activities.

Activity	Duration	Length, width	Place
1	0	0, 0	1
2	8	2, 1	1
3	4	2, 2	1
4	6	2, 1	2
5	3	2, 2	2
6	8	2, 1	2
7	5	2, 1	2
8	9	3, 1	1
9	2	3, 2	1
10	3	1, 1	1
11	7	2, 2	1
12	2	2, 2	2
13	7	3, 1	1
14	9	2, 1	2
15	4	3, 2	1
16	6	2, 1	2
17	3	2, 1	1
18	0	0, 0	2

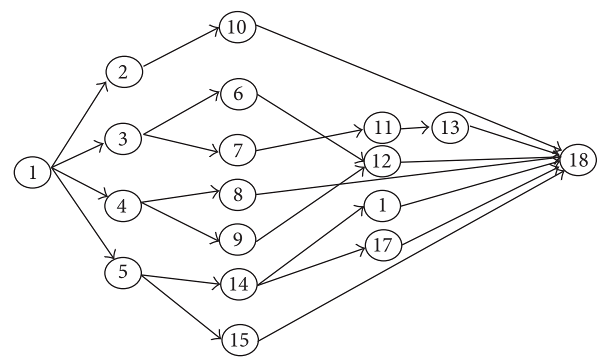


Figure 2: The network diagram of activities.

4. Simulation Experiment

We use a set of activities to verify our algorithm. The parameters of these activities are shown in Table 1, and the network diagram of activities is presented in Figure 2. There are 2 places in the experiment, x, y, and t represent length, width, and time, respectively. The length and width of place 1 are 3 and 3, while 3 and 2 are for place 2.

4.1. Generation of Initial Solution. We get the initial solution as follows.

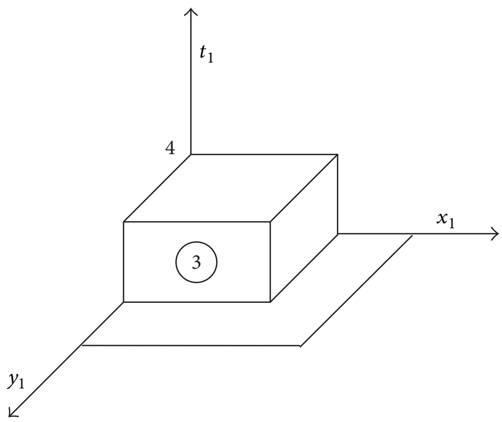


FIGURE 3: The status of place 1 while activity 3 is put.

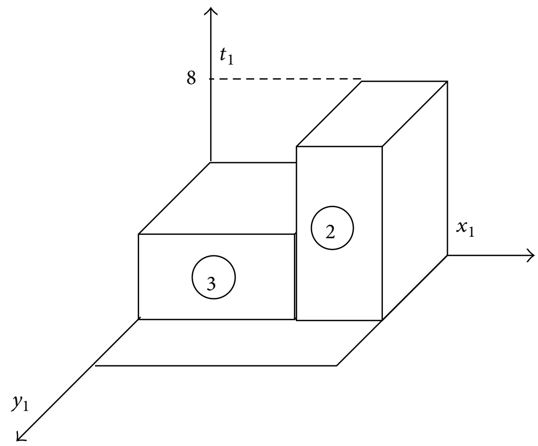


FIGURE 5: The status of place 1 while activity 2 is put.

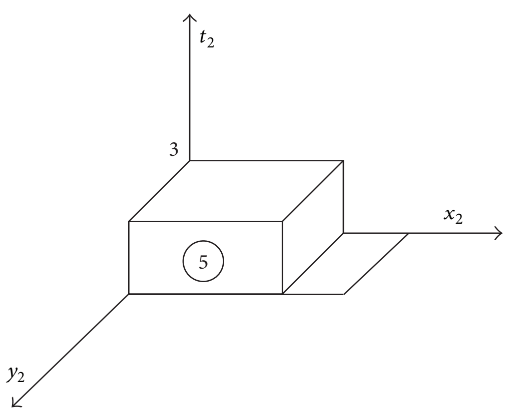


FIGURE 4: The status of place 2 while activity 5 is put.

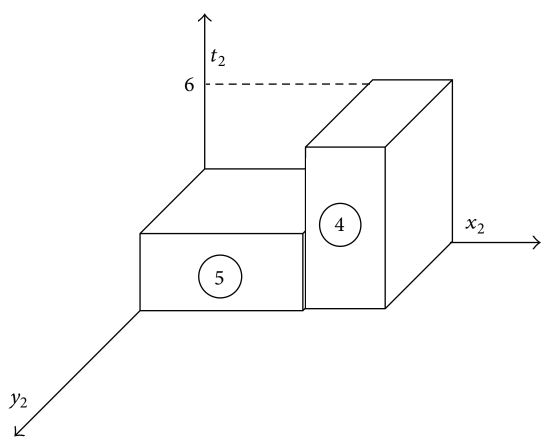


FIGURE 6: The status of place 2 while activity 4 is put.

(1) At first, $t = 0$. $\{2, 3, 4, 5\}$ can be arranged at this time. According to the rule above, we adjust the order to $\{3, 5, 2, 4\}$. Activity 3 will be completed in place 1, so we put the bottom-left-rear point of it at the origin of place 1. So three new corners $\{(0, 0, 4), (3, 3, \infty)\}$, $\{(0, 2, 0), (3, 1, \infty)\}$, $\{(2, 0, 0), (1, 3, \infty)\}$ are produced. Update A_n, D_n, I, $C_n = \{1\}$, $A_n = \{3\}$, $D_n = \{5, 2, 4\}$, $I_1 = \{\{(2, 0, 0), (1, 3, \infty)\}, \{(0, 2, 0), (3, 1, \infty)\}, \{(0, 0, 4), (3, 3, \infty)\}\}$, $I_2 = \{\{(0, 0, 0), (3, 2, \infty)\}\}$. See Figure 3.

(2) Next, activity 5 will be scheduled on place 2. We put the bottom-left-rear point of activity 5 at the origin of place 2. Then new corners $\{(0, 0, 3), (3, 2, \infty)\}$, $\{(0, 2, 0), (3, 0, \infty)\}$, $\{(2, 0, 0), (1, 2, \infty)\}$ are produced. According to the rule, we delete $\{(0, 2, 0), (3, 0, \infty)\}$. Update A_n, D_n, I. $C_n = \{1\}$, $A_n = \{3, 5\}$, $D_n = \{2, 4\}$, $I_1 = \{\{(2, 0, 0), (1, 3, \infty)\}, \{(0, 2, 0), (3, 1, \infty)\}, \{(0, 0, 4), (3, 3, \infty)\}\}$, $I_2 = \{\{(2, 0, 0), (1, 2, \infty)\}, \{(0, 0, 3), (3, 2, \infty)\}\}$. See Figure 4.

(3) Detect the corners of I_1 and $\{(2, 0, 0), (1, 3, \infty)\}$ is selected for activity 2. After activity 2 is scheduled, three new corners are produced and only $\{(2, 0, 8), (1, 3, \infty)\}$ is eligible to be a corner. Considering that part of the space has been occupied by activity 2, we should retest the available space of previous corners when we update I_1. $\{(0, 0, 4), (3, 3, \infty)\}$ should be modified into $\{(0, 0, 4), (2, 3, \infty)\}$. Then $C_n = \{1\}$, $A_n = \{3, 5, 2\}$, $D_n = \{4\}$, $I_1 = \{\{(0, 2, 0), (3, 1, \infty)\}, \{(0, 0, 4), (2, 3, \infty)\}, \{(2, 0, 8), (1, 3, \infty)\}\}$, $I_2 = \{\{(2, 0, 0), (1, 2, \infty)\}, \{(0, 0, 3), (3, 2, \infty)\}\}$. See Figure 5.

(4) At time 0, activity 4 will be put in order. Detect I_2 and put it on $\{(2, 0, 0), (1, 2, \infty)\}$. After testing, only $\{(2, 0, 6), (1, 2, \infty)\}$ can be a new corner. Similarly, the available space of previous corners in I_2 should also be retested and $\{(0, 0, 3), (3, 2, \infty)\}$ should be changed into $\{(0, 0, 3), (2, 2, \infty)\}$. $C_n = \{1\}$, $A_n = \{3, 5, 2, 4\}$, $D = \emptyset$, $I_1 = \{\{(0, 2, 0), (3, 1, \infty)\}, \{(0, 0, 4), (2, 3, \infty)\}, \{(2, 0, 8), (1, 3, \infty)\}\}$, $I_2 = \{\{(0, 0, 3), (2, 2, \infty)\}, \{(2, 0, 6), (1, 2, \infty)\}\}$. See Figure 6.

(5) At time 0, space is still available, but it is not enough for any remaining activity. So next time point is $t = 3$, and activity 5 is completed at this time. $C_n = \{1, 5\}$, $A_n = \{3, 2, 4\}$, $D_n = \{14, 15\}$. After ordering, D_n is changed into $\{15, 14\}$. In order to arrange activity 15, we detect I_1. If there is any corner whose value of t is less than 3, we should change it into 3 and modify available space of this

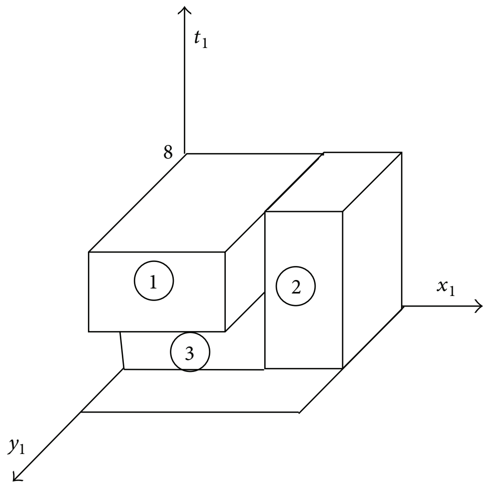
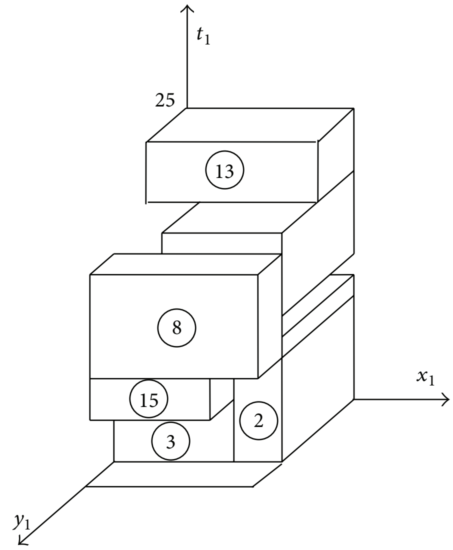


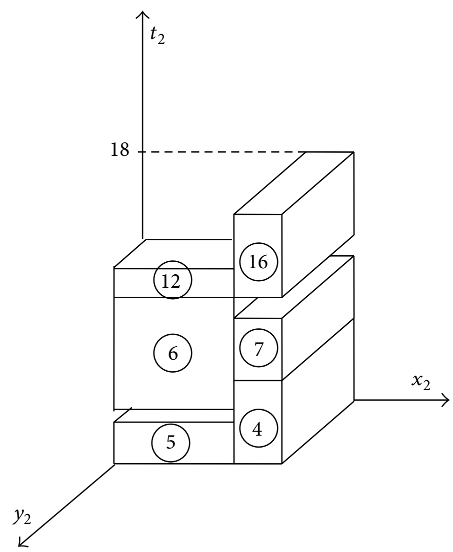
Figure 7: The status of place 1 while activity 15 is put.

corner. For example, $\{(0, 2, 0), (3, 1, \infty)\}$ should be changed into $\{(0, 2, 3), (3, 1, \infty)\}$ before activity 15 is put. Meanwhile, the space of this corner should be changed into $\{(0, 2, 3), (3, 1, 1)\}$ after activity 15 is put. We lay activity 15 on $\{(0, 0, 4), (2, 3, \infty)\}$. $C_n = \{1, 5\}$, $A_n = \{3, 2, 4, 15\}$, $D_n = \{14\}$, $I_1 = \{\{(0, 2, 3), (3, 1, 1)\}, \{(0, 0, 8), (3, 3, \infty)\}\}$, $I_2 = \{\{(0, 0, 3), (2, 2, \infty)\}, \{(2, 0, 6), (1, 2, \infty)\}\}$. See Figure 7.

(6) According to this train of thought, we get an initial solution $\{3(0, 0, 0)\}$, $\{5(0, 0, 0)\}$, $\{2(2, 0, 0)\}$, $\{4(2, 0, 0)\}$, $\{15(0, 0, 4)\}$, $\{14(0, 0, 3)\}$, $\{6(0, 1, 4)\}$, $\{7(2, 0, 6)\}$, $\{9(0, 0, 8)\}$, $\{8(0, 2, 8)\}$, $\{10(0, 0, 10)\}$, $\{11(1, 0, 11)\}$, $\{12(0, 0, 12)\}$, $\{16(2, 0, 12)\}$, $\{17(0, 0, 13)\}$, $\{13(0, 0, 18)\}$, in which $\{B(x, y, t)\}$ means activity B should be started at time t, and the space it needs can be represented as (x, y). According to the initial solution, the whole duration of the project is 25. The status of place 1 and place 2 is shown in Figure 8.



(a) Place 1



(b) Place 2

Figure 8: The status while all activities are put.

4.2. Optimization of Initial Solution. In SA, the order of activities has great influence on the duration of the project. So by searching the neighborhood and increasing the size of neighborhood dynamically, we manage to use SA and improve the initial solution.

We show a step of improvement as follows.

(1) $f = f_{\text{best}} = 25$, $B = B_{\text{best}} = \{3, 5, 2, 4, 15, 14, 6, 7, 9, 8, 10, 11, 12, 16, 17, 13\}$.

(2) $t = S_t$, $L_t = L$.

(3) If $t \geq E_t$, go to step (4); else output current f_{best} and B_{best}; finish the program.

(4) $j = 1$.

(5) If $j > L_t$, $L_t += d_L$, and go to step (4); else go to next step.

(6) Select B' from the neighborhood $N(B)$ of B. In our example, we swap the order of activity 6 and activity 7. Then, we get another solution $\{3(0, 0, 0)\}$, $\{5(0, 0, 0)\}$, $\{2(2, 0, 0)\}$, $\{4(2, 0, 0)\}$, $\{15(0, 0, 4)\}$, $\{14(0, 0, 3)\}$, $\{7(0, 1, 4)\}$, $\{6(2, 0, 6)\}$, $\{9(0, 0, 8)\}$, $\{8(0, 2, 8)\}$, $\{10(0, 0, 10)\}$, $\{11(1, 0, 10)\}$, $\{16(0, 0, 12)\}$, $\{17(0, 0, 13)\}$, $\{12(0, 0, 18)\}$, $\{13(0, 2, 17)\}$. $f' = 24$, $B' = \{3, 5, 2, 4, 15, 14, 7, 6, 9, 8, 10, 11, 16, 17, 12, 13\}$. $d_f = f - f' = 25 - 24 = 1$. If $d_f > 0$, go to step (7); else, go to step (8). The current d_f is 1, so go to step (7).

(7) Compare f' and f_{best}, if $f' < f_{\text{best}}$, $f_{\text{best}} = f'$, $B_{\text{best}} = B'$; else $j = j + 1$ and go to step (5). In this example, $f' < f_{\text{best}}$, so $f_{\text{best}} = 24$, $B_{\text{best}} = \{3, 5, 2, 4, 15, 14, 7, 6, 9, 8, 10, 11, 16, 17, 12, 13\}$.

(8) Generate a number x randomly, if $x < \exp(10 * d_f / t)$, $f = f'$, $B = B'$, $j = j + 1$, and go to step (5).

After this improvement, the optimal solution reduces from 25 to 24. So it is proved that the method is effective.

5. Conclusion

In this paper, we combine BLF and SA to solve ship block construction space scheduling problem. During the procedure of scheduling activities, we guide the sorting process with corners. Then, the sorting of initial solution can be changed by SA. However, how to improve the searching efficiency will be the future research, especially when the number of blocks is very large.

Conflict of Interests

The authors declare that they have no financial and personal relationships with other people or organizations that can inappropriately influence their work, and they also declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported in part by NSFC under Grant no. 61202345, NSFS under Grant no. ZR2012FM006, and SDPW under Grant no. IMZQWH010016. The authors thank the 2 reviewers and handling editor for their meticulous reading of the paper and constructive comments which greatly improved the paper.

References

[1] R. Kolisch, "Efficient priority rules for the resource-constrained project scheduling problem," *Journal of Operations Management*, vol. 14, no. 3, pp. 179–192, 1996.

[2] S. Hartmann, "A competitive genetic algorithm for resource-constrained project scheduling," *Naval Research Logistics*, vol. 45, no. 7, pp. 733–750, 1998.

[3] Y. Q. Yin, T. C. E. Cheng, X. Q. Yang, and C.-C. Wu, "Two-agent single-machine scheduling with unrestricted due date assignment," *Computers & Industrial Engineering*, vol. 79, pp. 148–155, 2015.

[4] Y. Yin, S. R. Cheng, and C. C. Wu, "Parallel-machine scheduling to minimize flowtime, holding, and batch delivery costs," *Asia-Pacific Journal of Operational Research*, vol. 31, no. 6, Article ID 1450044, 17 pages, 2014.

[5] E. L. Demeulemeester and W. S. Herroelen, "New benchmark results for the resource-constrained project scheduling problem," *Management Science*, vol. 43, no. 11, pp. 1485–1492, 1997.

[6] P. Raj, "Solving spatial scheduling problem: an analytical approach," in *Proceedings of the 37th International Conference on Computers and Industrial Engineering*, pp. 2002–2011, 2007.

[7] J. E. Kelley, "The critical-path method: resources planning and scheduling," *Industrial Scheduling*, vol. 13, pp. 347–365, 1963.

[8] Z. Liu and H. Wang, "Heuristic algorithm for RCPSP with the objective of minimizing activities' cost," *Journal of Systems Engineering and Electronics*, vol. 17, no. 1, pp. 96–102, 2006.

[9] T. Bhaskar, M. N. Pal, and A. K. Pal, "A heuristic method for RCPSP with fuzzy activity times," *European Journal of Operational Research*, vol. 208, no. 1, pp. 57–66, 2011.

[10] K. J. Lee, J. K. Lee, and S. Y. Choi, "A spatial scheduling system and its application to shipbuilding: DAS-CURVE," *Expert Systems with Applications*, vol. 10, no. 3-4, pp. 311–324, 1996.

[11] S. Koh, R. Logendran, D. Choi, and S. Woo, "Spatial scheduling for shape-changing mega-blocks in a shipbuilding company," *International Journal of Production Research*, vol. 49, no. 23, pp. 7135–7149, 2011.

[12] B. S. Baker, E. G. Coffman Jr., and R. L. Rivest, "Orthogonal packings in two dimensions," *SIAM Journal on Computing*, vol. 9, no. 4, pp. 846–855, 1980.

[13] B. Chazelle, "bottom-left bin packing heuristic: an efficient implementation," *IEEE Transactions on Computers*, vol. 32, no. 8, pp. 697–707, 1983.

[14] G. Belov, G. Scheithauer, and E. A. Mukhacheva, "One-dimensional heuristics adapted for two-dimensional rectangular strip packing," *Journal of the Operational Research Society*, vol. 59, no. 6, pp. 823–832, 2008.

[15] T. M. Chan, F. Alvelos, E. Silva, and J. M. V. de Carvalho, "Heuristics with stochastic neighborhood structures for two-dimensional bin packing and cutting stock problems," *Asia-Pacific Journal of Operational Research*, vol. 28, no. 2, pp. 255–278, 2011.

[16] S. Martello, D. Pisinger, and D. Vigo, "The three-dimensional bin packing problem," *Operations Research*, vol. 48, no. 2, pp. 256–267, 2000.

[17] R. Alvarez-Valdes, F. Parreño, and J. M. Tamarit, "A GRASP/Path relinking algorithm for two- and three-dimensional multiple bin-size bin packing problems," *Computers and Operations Research*, vol. 40, no. 12, pp. 3081–3090, 2013.

[18] C.-S. Liao and C.-H. Hsu, "New lower bounds for the three-dimensional orthogonal bin packing problem," *European Journal of Operational Research*, vol. 225, no. 2, pp. 244–252, 2013.

[19] M. Kucuk and M. Ermis, "A new hybrid evolutionary algorithm for three-dimensional packing problems," in *Proceedings of the IEEE International Conference on Systems Man and Cybernetics (SMC '10)*, pp. 4029–4034, IEEE, Istanbul, Turkey, October 2010.

[20] M. Steinbrunn, G. Moerkotte, and A. Kemper, "Heuristic and randomized optimization for the join ordering problem," *VLDB Journal*, vol. 6, no. 3, pp. 191–208, 1997.

[21] S. Kirkpatrick, J. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[22] F. T. S. Chan, A. Prakash, H. L. Ma, and C. S. Wong, "A hybrid Tabu sample-sort simulated annealing approach for solving distributed scheduling problem," *International Journal of Production Research*, vol. 51, no. 9, pp. 2602–2619, 2013.

[23] K. Bouleimen and H. Lecocq, "A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version," *European Journal of Operational Research*, vol. 149, no. 2, pp. 268–281, 2003.

[24] V. Yannibelli and A. Amandi, "Hybridizing a multi-objective simulated annealing algorithm with a multi-objective evolutionary algorithm to solve a multi-objective project scheduling problem," *Expert Systems with Applications*, vol. 40, no. 7, pp. 2421–2434, 2013.