Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 628259, 9 pages
http://dx.doi.org/10.1155/2015/628259

Research Article

A Branch and Bound Algorithm for Project Scheduling Problem with Spatial Resource Constraints

Shicheng Hu,[1] Song Wang,[1] Yonggui Kao,[2] Takao Ito,[1] 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 1 September 2014; Accepted 28 September 2014

Academic Editor: Yunqiang Yin

With respect to the block assembly schedule in a shipbuilding enterprise, a spatial resource constrained project scheduling problem (SRCPSP) is proposed, which aims to minimize the makespan of a project under the constraints of the availability of a two-dimensional spatial resource and the precedence relationship between tasks. In order to solve SRCPSP to the optimum, a branch and bound algorithm (BB) is developed. For the BB-SRCPSP, first, an implicitly enumerative branch scheme is presented. Secondly, a precedence based lower bound, as well as an effective dominance rule, is employed for pruning. Next, a heuristic based algorithm is used to decide the order of a node to be selected for expansion such that the efficiency of the algorithm is further improved. In addition, a maximal space based arrangement is applied to the configuration of the areas required each day in an available area. Finally, the simulation experiment is conducted to illustrate the effectiveness of the BB-SRCPSP.

1. Introduction

RCPSP [1] (resource constrained project scheduling problem) is originated from the techniques of CPM (critical path method) and PERT (project evaluation and review technique). Soon after the integer programming model of the RCPSP is proposed, it was proved to be a NP-hard problem. Due to the wide existence of the two conditions in practice, that is, precedence relationship between tasks and resource availability, the RCPSP, which employs these two conditions as hard constraints, has been the hot research topic so far. As the core classical problem in project scheduling, sufficient achievements have been acquired on its algorithms, models, and applications.

The algorithms on RCPSP are classified into two categories, optimal algorithm and approximate algorithm, which produce optimal solution and nearly optimal solution, respectively. Branch and bound algorithm [2] is the most used optimal algorithm, which gets the optimal solution with the cost of time-consuming. For the improvement of efficiency, more efforts are paid to the theory of lower bound, which can avoid large amounts of unnecessary searches. The approximate algorithms are divided into heuristic, metaheuristic, and hybrid heuristic [3]. The heuristic is the mostly employed algorithm to RCPSP, which starts from a partial feasible solution to construct a complete feasible solution. A heuristic contains two parts, priority rule and schedule generation scheme. The metaheuristic often constructs a feasible solution by the use of a heuristic and then improves it to a nearly optimal solution. The classical metaheuristics include simulated annealing [4], tabu search [5], genetic algorithm [6], neural network [7], immune algorithm [8], ant colony [9], and particle swarm [10]. Recently, more and more efforts are involved in the research of hybrid heuristic [11] and satisfactory results have been achieved. The hybrid heuristic is one of the most potential researches in the future.

To satisfy the needs of real applications, different variations are derived from the classical RCPSP: the strict precedence relationship between tasks was extended to a more general precedence relationship [12]; a new type of nonrenewable resource was introduced [13]; the resource usage can have more than one mode [14]; in addition to

project makespan, the objective included project cost [15], net present value [16], resource leveling [17], resource investment [18], time-dependent cost [19–21], and position-dependent cost [22, 23]. The combinations of different constraints and objectives can produce different resource scheduling model with different background. Meanwhile, the different resource scheduling models were developed from deterministic ones to probabilistic ones so as to cope with different risks and uncertainties in applications.

Lee was the first to study the spatial resource scheduling problem [24]. He employed Lozano-Perez's two-dimensional arrangement algorithm to solve the problem of how to arrange the nonregular polygons in a rectangle. Lee applied this method to the curved-bottom block assembly scheduling and further to the flat-bottom block assembly scheduling. Also, several other different solution schemes were also proposed to this problem. Afterwards, the ship painting spatial resource scheduling problem was proposed [25]. Next, the ship block erecting spatial resource scheduling problem was studied [26]. Furthermore, the ship megablock assembly spatial resource scheduling problem was tackled [27]. In other application fields, an operation system for reconfigurable embedded platforms was developed [28] and a bin packing problem with the constraint of precedence relationship was presented [29].

In this paper, a spatial resource constrained project scheduling problem is modeled and a corresponding branch and bound algorithm is developed. This paper is organized as follows: in Section 2, a conceptual model is proposed; in Section 3 the branch and bound method is described; a demonstrative simulation example is presented in Section 4; and the summary conclusion is provided in the final section.

2. Spatial Resource Constrained Project Scheduling Problem (SRCPSP)

For ship manufacturing enterprises, spatial resource scheduling is a representative problem in the shipbuilding project management. Here, the ship block assembly project is used as an example to illustrate it. A block is a cubic module with different shape and volume that is divided from a ship in the design stage (a large ship is generally divided into hundreds of blocks), which is the most important intermediate product in a ship building. All blocks will be assembled together to form a ship according to the precedence relationship between them. The assembly of blocks is operated on a platform, which is a two-dimensional area equipped with jigs. Since for a shipbuilding enterprise, the platform is an expensive and scare resource, whose utility efficiency has a direct impact on the makespan of a ship. The optimization of the platform configuration is the key problem to be solved in a ship block assembly project scheduling. The platform scheduling is how to arrange the start time of each task such that the makespan of a ship block assembly project is minimized under the constraints of the precedence relationship between block assembly tasks and the availability of platform resource. In a ship building enterprise, except for the assembly of blocks, the painting of blocks, the general assembly of blocks, and the erecting of blocks into a final ship are also operated on different types of platforms, which are also expensive and scare resources in an enterprise. Therefore, for the different production project plans, the spatial resource scheduling is a key problem for a manufacturing enterprise to solve. The spatial resource constrained project scheduling problem (SRCPSP) can be expressed as follows:

$$\text{minimize } S_{n+1} \tag{1}$$

$$S_i + d_i \le S_j, \quad \forall i \in P_j \tag{2}$$

$$a_{ik} \cap a_{jk} = \phi, \quad \forall i, j \in N, \ \forall k \in M \tag{3}$$

$$0 \le w_{ik} \le W_k, \quad \forall i \in N, \ \forall k \in M \tag{4}$$

$$0 \le l_{ik} \le L_k, \quad \forall i \in N, \ \forall k \in M \tag{5}$$

$$\delta_i \in \{0, 1\}, \quad \forall i \in N. \tag{6}$$

A SRCPSP has a set of tasks, $N = \{0, 1, \ldots, n, n+1\}$. Each task i has a requirement $a_{ik} = w_{ik} \times l_{ik}$ for spatial resource k, where w_{ik} and l_{ik} denote the width and length of a two-dimensional area a_{ik}, respectively. And each task i will be completed in a period of d_i. The resource requirements and durations for the dummy start and end tasks 0 and $n + 1$ are 0. There are a set of spatial resources, $M = \{1, \ldots, m\}$, each of which has a availability $A_k = W_k \times L_k$, where W_k and L_k denote the width and length of a two-dimensional area A_k, respectively. Formula (2) represents the precedence relationship between tasks, which means that a task j cannot start before any of its predecessors P_j. Formulae (2)–(5) present the spatial resource constraint; they define that for each day, the spatial resource requirements a_{ik} and a_{jk} for any two tasks i and j which have the same type of resource k cannot overlap (see (3)) and the width w_{ik} and length l_{ik} of spatial resource requirements a_{ik} for any tasks i should be confined in the corresponding dimensions W_k and L_k of the available spatial resource A_k (see (4) and (5)). In order to produce a better optimal solution, the 90-degree-based rotation of the required area is permitted. Formula (6) denotes two orientations for each required area a_{ik}; $\delta_i = 0$ indicating that the width of a_{ik} is parallel to the width of A_k and $\delta_i = 1$ indicating that the length of a_{ik} is parallel to the width of A_k. The objective of SRCPSP is to minimize the makespan S_{n+1} (see (1)).

3. A Branch and Bound Algorithm for SRCPSP (BB-SRCPSP)

A branch and bound algorithm includes branch scheme and bound computation. For the branch and bound algorithm to SRCPSP, the arrangement of a spatial area a_{ik} required by a task i in the available area A_k should also be specified. This section will tackle these three problems involved in the branch and bound algorithm for SRCPSP (BB-SRCPSP).

3.1. Branch Scheme. The branching scheme involves how to develop the nodes of a tree from one root node. Each node in a tree represents a partial schedule, and the root node

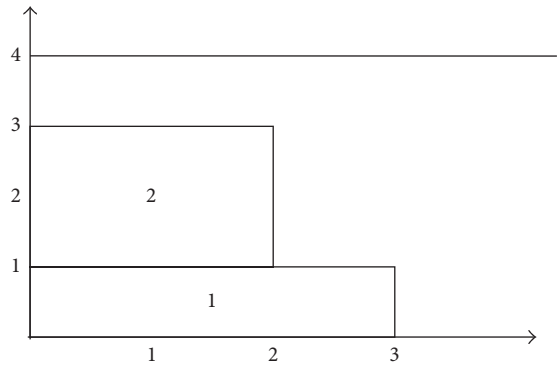

Figure 1: The Gantt chart of a partial schedule at time p.

represents one in which only the start task 0 is schedule at time zero. Related to each decision time t, there are four sets of tasks, the partial schedule Q_t in which each task is assigned a start time, the active schedule S_t in which each task is being processed but not completed, the complete schedule C_t in which each task is finished, and a decision set D_t in which the predecessors of each task are in C_t such that it could be selected as the next task to be scheduled.

For the nodes that are expanded from an ancestor, they have the same decisions made previously, which means that except for the current tasks that are scheduled, the tasks that are scheduled previously have the same start time. Given a node, the production of its braches can be described as follows: at a decision time p, there are four sets of tasks, Q_t, S_t, C_t, and D_t, related to a node b, which will produce several descendants at next time q, each corresponding to a branch. We will select the complete time of the first task in S_t which has the earliest finish time as the next decision time q. Then we will decide the alternatives from D_t, each of which should satisfy the resource constraints. An alternative may be any task in D_t or a combination of the tasks in D_t that could be scheduled at time q.

A demonstrative example is shown on how to branch a node. Figure 1 is the Gantt chart of a partial schedule: $p = 0$, $Q_p = \{0, 1, 2\}$, $S_p = \{1, 2\}$, $C_p = \{0\}$, and $D_p = \{3, 4, 5\}$. Assume that the availability of the only one type of resource $R = 4$ and the resource requirements r_3, r_4, and r_5 of tasks 3, 4, and 5 are 1, 2, and 4, respectively. According to the tree creation process described above, the next decision time is the completion time of task 2; $q = 2$. At time q, the alternative set $A_q = \{\{3\}, \{4\}, \{3, 4\}\}$ since the requirement of each alternative in A_q is no more than the resource availability at time q. So from the node corresponding to partial schedule $Q_p = \{0, 1, 2\}$ at time p, three branches will be produced, which correspond to three partial schedules at time q, $Q_{q1} = \{0, 1, 2, 3\}$, $Q_{q2} = \{0, 1, 2, 4\}$, and $Q_{q3} = \{0, 1, 2, 3, 4\}$. As is indicated above, the three branches have a common partial schedule shown in Figure 1, $Q_p = \{0, 1, 2\}$. The current scheduled tasks at time q of the three descendant partial schedules are the only difference between them. The produced branches are shown in Figure 2.

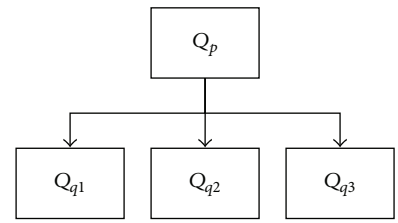

Figure 2: The produced branches from a partial schedule at time q.

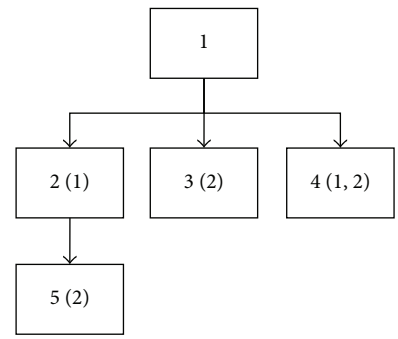

Figure 3: The three branches emanated from the root node.

3.2. Bound Computation. The tree will have an exhaustive enumeration if no pruning is introduced and the branch and bound algorithm will be very time-consuming. In order to improve the efficiency of BB-SRCPSP, we design some pruning strategies.

The lower bound based pruning is the most effective method that is employed in the literatures. A lower bound is a predicted value on the makespan of a partial schedule, $\mathrm{LB}(Q_t)$. An upper bound is always set as the makespan of the incumbent optimal solution, UB. If $\mathrm{LB}(Q_t) \geq \mathrm{UB}$, it can be deduced that from the partial schedule Q_t, a better optimal solution than the incumbent will never be produced and then the corresponding node should be pruned; that is, it is unnecessary to create branches from this node. The lower bound value will be computed once a new partial schedule is produced and the incumbent upper bound will be updated if a better optimal solution is found.

For BB-SRCPSP, we compute the precedence based lower bound $\mathrm{PLB}(Q_t)$. Given a partial schedule Q_t, assuming the set of tasks that are scheduled at time t to be H, the $\mathrm{PLB}(Q_t)$ can be computed as follows:

$$\mathrm{PLB} = \max\left\{t + d_i + \mathrm{CPL}\left(j\right) \mid i \in H, j \in S\left(i\right)\right\}, \quad (7)$$

where $\mathrm{CPL}(j)$ is the critical path length from node j to the end node $n + 1$ of a project and $S(i)$ is the set of immediate successors of node i.

Given a partial schedule Q, if some tasks can be left-shifted without violating either the resource or precedence constraints, this partial schedule will be dominated. This is a very effective dominance rule that is used for pruning. A demonstrative example is given on how this dominance rule works. Emanating from a root node is three branches nodes 2, 3, and 4, shown in Figure 3. One of the descendants of node 1 is node 5, which corresponds to the partial schedule shown in

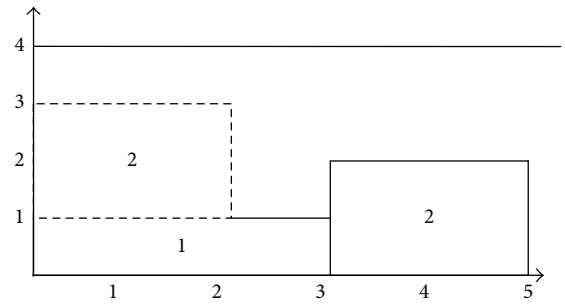

Figure 4: The Gantt chart of a partial schedule corresponding to node 5.

Figure 4. Note that the task 2 in Figure 4 could be left-shifted by 3 time units without violating any constraint, so this partial schedule would be dominated and the corresponding node 5 in the tree should be pruned.

It is known that the order of the same node in the tree chosen to be expanded has a great impact on whether or when it will be pruned and thus on the efficiency of a BB algorithm. The node in the tree which has the largest lower bound will be selected for expansion. Then an arbitrary node will be selected to break the ties if more than one candidate tasks exist.

3.3. Arrangement of a Spatial Area. For a traditional RCPSP, the resource constraint requires that the sum of the quantities of the resource demands of the tasks on each day is no larger than the quantity of the same type resource availability. But for a SRCPSP, the satisfaction of the resource constraint means that in addition to that for a RCPSP, the precise position of an area required by a task should also be specified in an available area. The configuration of an area is a very complicated problem. In order to get a better optimal solution, the area available on each day should be filled more efficiently; alternatively speaking, the configuration of an area will impact the quality of a solution.

The configuration of an area is a two-dimensional packing problem. So far, many methods have been presented to tackle it, among which the maximal space-based algorithm is proved to be very effective and is employed in our BB-SRCPSP. Each time if there would be more than one maximal space produced, the one with the smallest distance to a corner of the available area will be chosen as the one to be filled and the corner of the selected maximal space with the least distance to a corner of the available area will be chosen as the position that a task will be put on.

To illustrate how the maximal space is produced, the reader is referred to Figure 5. Initially, the left-bottom corner of area 1 is used to put an area and then two maximal spaces are produced, areas 2 and 3. Afterwards, the area 2 is selected and three maximal spaces are generated, areas 4, 5, and 6.

4. Simulation Experiment

In this section, a set of simulated data is used to illustrate the proposed branch and bound algorithm for SRCPSP, which

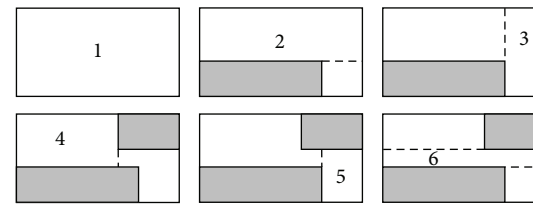

Figure 5: The production of maximal spaces.

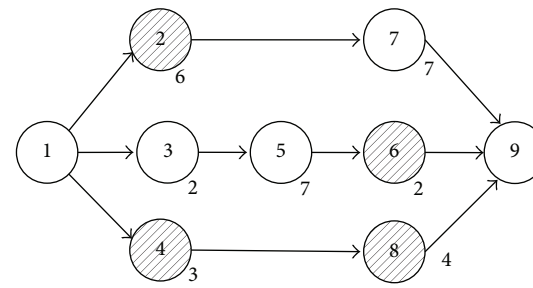

Figure 6: The precedence relationships between tasks.

Table 1: The parameters of tasks.

Task number	Duration	Length, width	Area type
1	0	0, 0	
2	6	3, 1	1
3	2	3, 2	2
4	3	2, 1	1
5	7	2, 2	2
6	2	2, 2	1
7	7	3, 1	2
8	4	3, 2	1
9	0	0, 0	

is adapted from the well-known PSPLIB [30]. The instance includes 9 tasks; the precedence relationship between tasks is shown in Figure 6. There are two types of spatial resources, whose availability is 4×3 square units. The parameters for each task are listed in Table 1.

At time $t = 0$, $Q_t = \{1\}$, $S_t = \Phi$, $C_t = \{1\}$, and $D_t = \{2, 3, 4\}$. 7 nodes numbered 2, 3, 4, 5, 6, 7, and 8 are created as shown in Figure 7 and the corresponding partial schedules are illustrated in Figures 8, 9, 10, 11, 12, 13, and 14.

The lower bounds, $\mathrm{LB}(Q_t)$, for nodes 2, 5, 6, and 8, are equal to 13, respectively. So node 6 is selected arbitrarily for expansion.

From node 6, $t = 3$, $Q_t = \{1, 2, 4\}$, $S_t = \{2\}$, $C_t = \{1, 4\}$, and $D_t = \{3, 8\}$. 3 nodes numbered 9, 10, and 11 are created as shown in Figure 7 and the corresponding partial schedules are illustrated in Figures 15, 16, and 17.

The lower bounds, $\mathrm{LB}(Q_t)$, for nodes 2, 5, 8, 9, 10, and 11, are equal to 13, respectively. So node 10 is selected arbitrarily for expansion.

From node 10, $t = 6$, $Q_t = \{1, 2, 4, 8\}$, $S_t = \{6\}$, $C_t = \{1, 2, 4\}$, and $D_t = \{3, 7\}$. 3 nodes numbered 12, 13, and 14 are

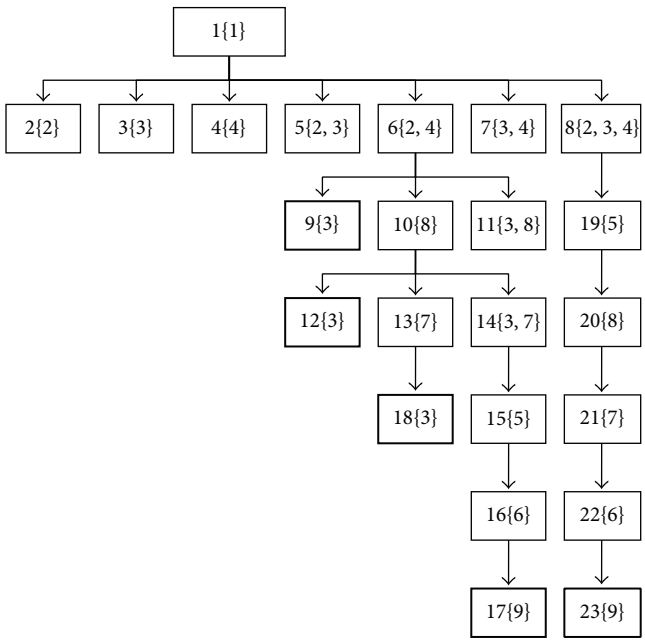

Figure 7: The search tree.

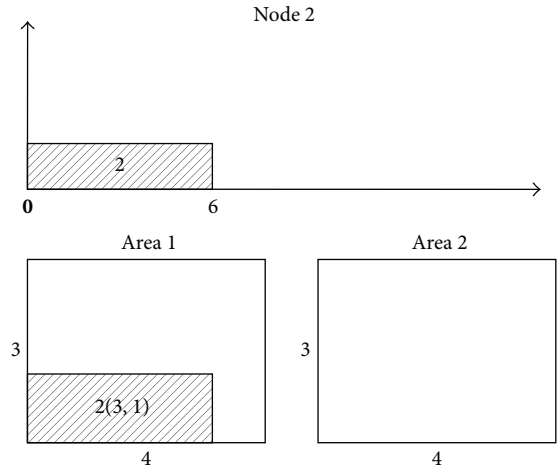

Figure 8: The partial schedule corresponding to node 2.

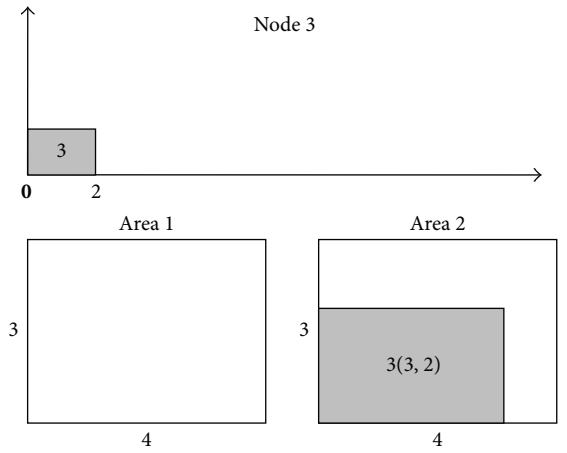

Figure 9: The partial schedule corresponding to node 3.

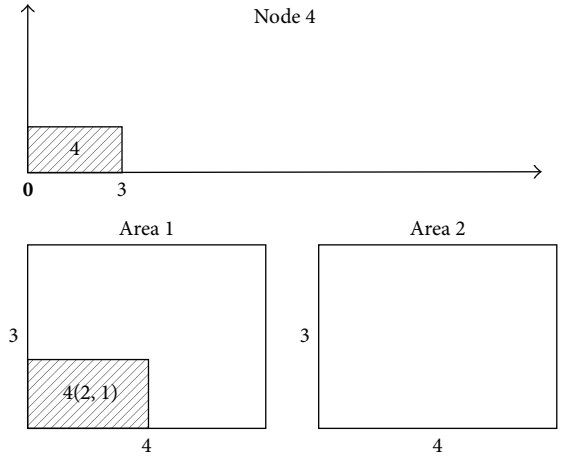

Figure 10: The partial schedule corresponding to node 4.

created as shown in Figure 7 and the corresponding partial schedules are illustrated in Figures 18, 19, and 20.

The lower bounds, LB(Q_t), for nodes 2, 5, 8, 9, 11, 12, 13, and 14, are equal to 13, respectively. So node 14 is selected arbitrarily for expansion.

From node 14, $t = 7$, $Q_t = \{1, 2, 4, 8, 3, 7\}$, $S_t = \{3, 7\}$, $C_t = \{1, 2, 4, 8\}$, and $D_t = \Phi$. Then $t = 8$, $Q_t = \{1, 2, 4, 8, 3, 7\}$, $S_t = \{7\}$, $C_t = \{1, 2, 4, 8, 3\}$, and $D_t = \{5\}$. 1 node numbered 15 is created as shown in Figure 7 and the corresponding partial schedule is illustrated in Figure 21.

The lower bounds, LB(Q_t), for nodes 2, 5, 8, 9, 11, 12, 13, and 15, are equal to 13, respectively. So node 15 is selected arbitrarily for expansion.

From node 15, $t = 13$, $Q_t = \{1, 2, 4, 8, 3, 7, 5\}$, $S_t = \{5\}$, $C_t = \{1, 2, 4, 8, 3, 7\}$, and $D_t = \Phi$. Then $t = 15$, $Q_t = \{1, 2, 4, 8, 3, 7, 5\}$, $S_t = \Phi$, $C_t = \{1, 2, 4, 8, 3, 7, 5\}$, and $D_t = \{6\}$. 1 node numbered 16 is created as shown in Figure 7 and the corresponding partial schedule is illustrated in Figure 22.

The lower bounds, LB(Q_t), for nodes 2, 5, 8, 9, 11, 12, 13, and 16, are equal to 13, respectively. So node 16 is selected arbitrarily for expansion.

From node 16, $t = 17$, $Q_t = \{1, 2, 4, 8, 3, 7, 5, 6\}$, $S_t = \Phi$, $C_t = \{1, 2, 4, 8, 3, 7, 5, 6\}$, and $D_t = \{9\}$. 1 node numbered 17 is created as shown in Figure 7 and the corresponding partial schedule is illustrated in Figure 23. As $|Q_t| = 9$, then an optimal schedule is produced, UB = 17.

The lower bounds, LB(Q_t), for nodes 2, 9, 12, and 13, are equal to 13, respectively. So node 13 is selected arbitrarily for expansion.

From node 13, $t = 7$, $Q_t = \{1, 2, 4, 8, 7\}$, $S_t = \{7\}$, $C_t = \{1, 2, 4, 8\}$, and $D_t = \{3\}$. 1 node numbered 18 is created as shown in Figure 7 and the corresponding partial schedule is illustrated in Figure 24.

For the partial schedule corresponding to node 18, the task 3 can left-shift to 0 without violating the constraints, so this partial schedule is dominated and the node 18 is pruned.

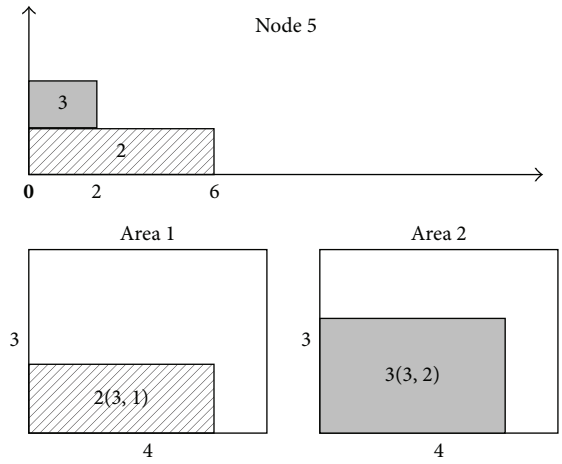

Figure 11: The partial schedule corresponding to node 5.

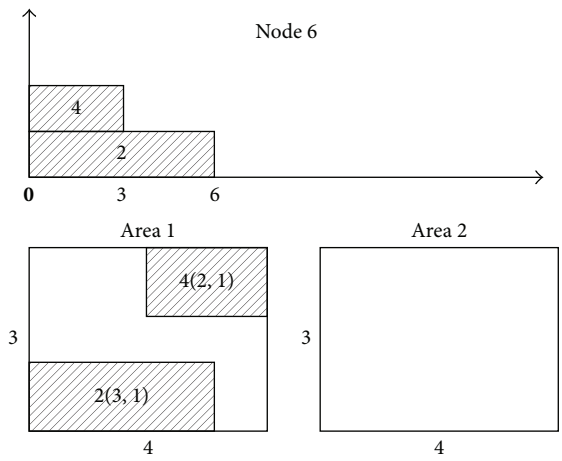

Figure 12: The partial schedule corresponding to node 6.

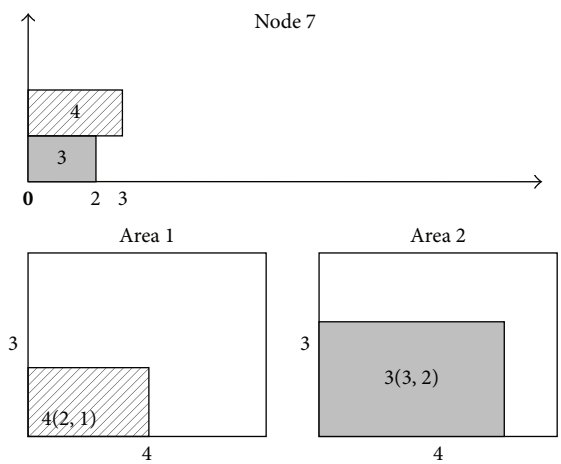

Figure 13: The partial schedule corresponding to node 7.

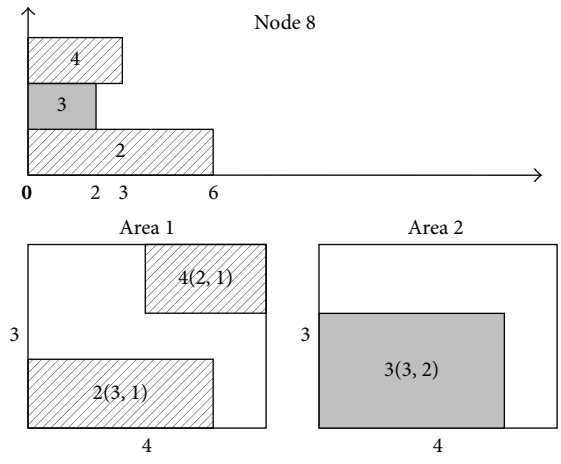

Figure 14: The partial schedule corresponding to node 8.

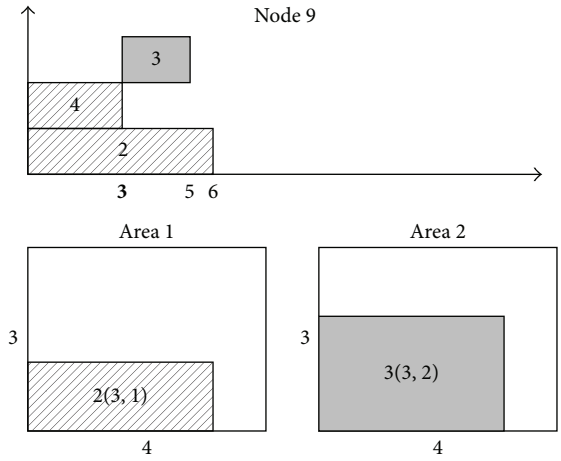

Figure 15: The partial schedule corresponding to node 9.

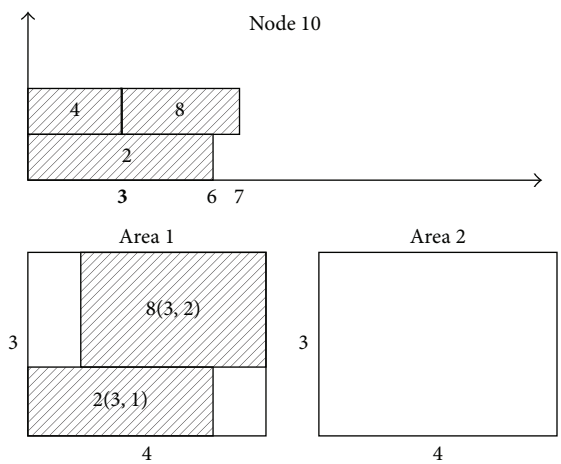

Figure 16: The partial schedule corresponding to node 10.

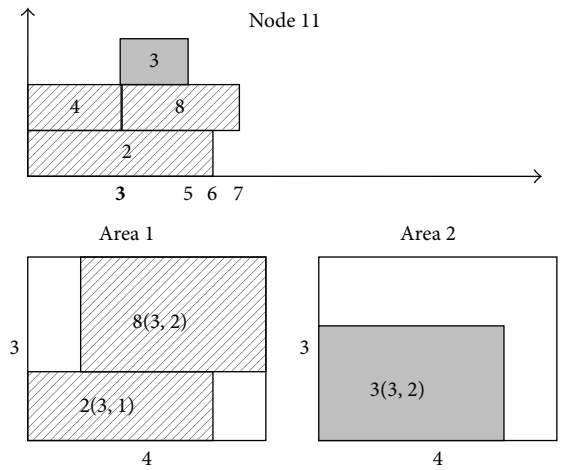

Figure 17: The partial schedule corresponding to node 11.

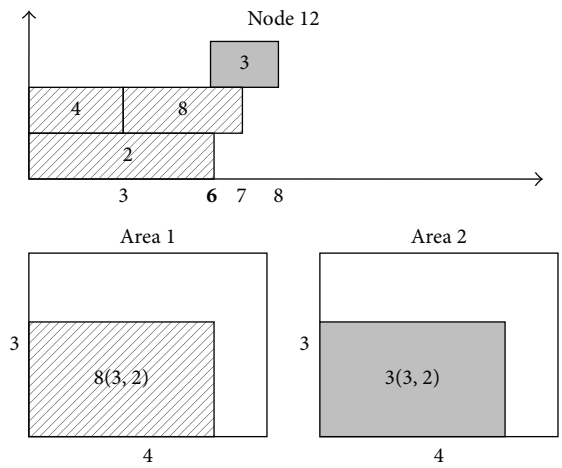

Figure 18: The partial schedule corresponding to node 12.

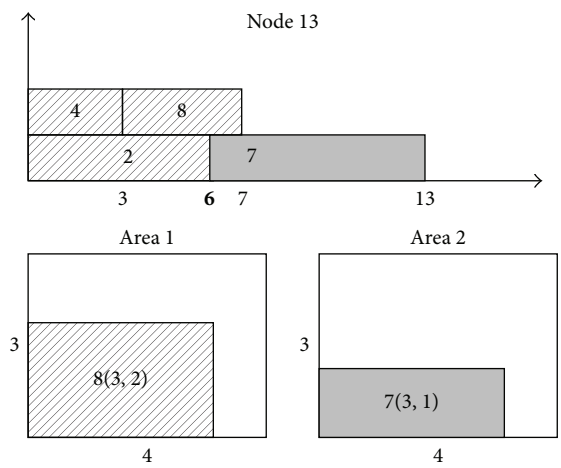

Figure 19: The partial schedule corresponding to node 13.

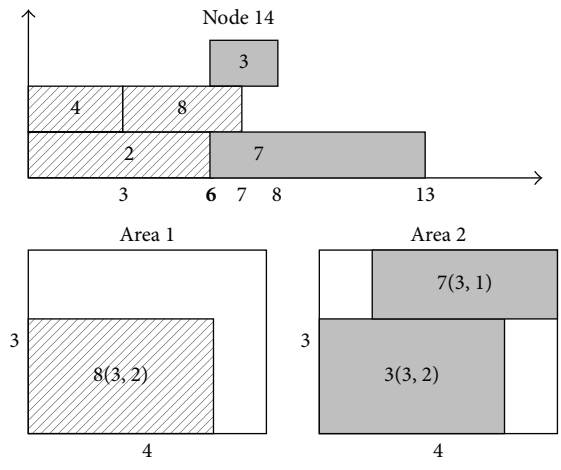

Figure 20: The partial schedule corresponding to node 14.

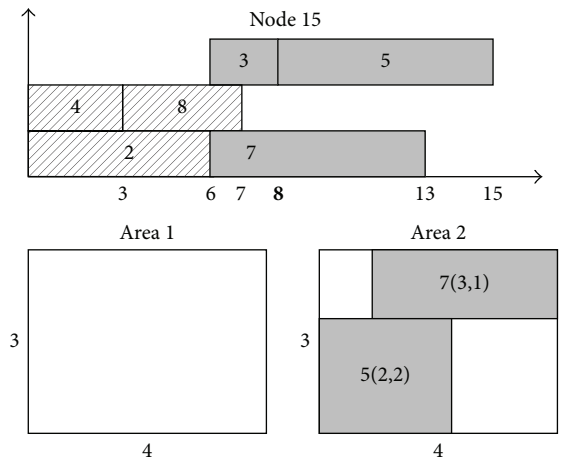

Figure 21: The partial schedule corresponding to node 15.

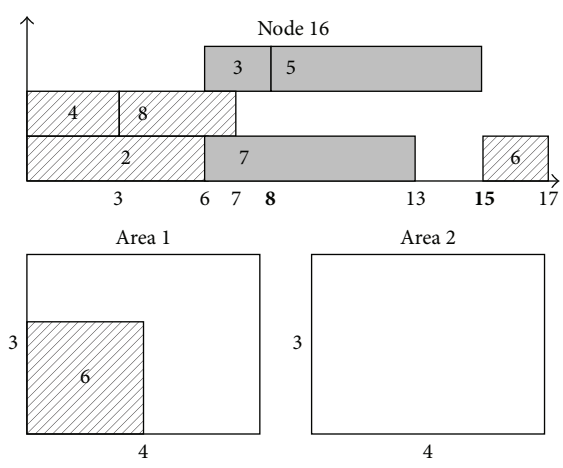

Figure 22: The partial schedule corresponding to node 16.

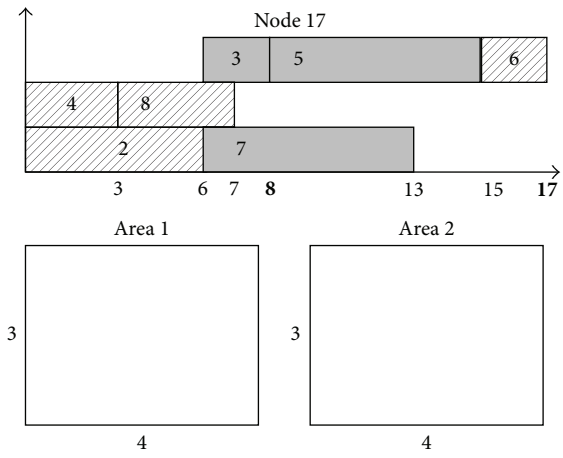

Figure 23: The partial schedule corresponding to node 17.

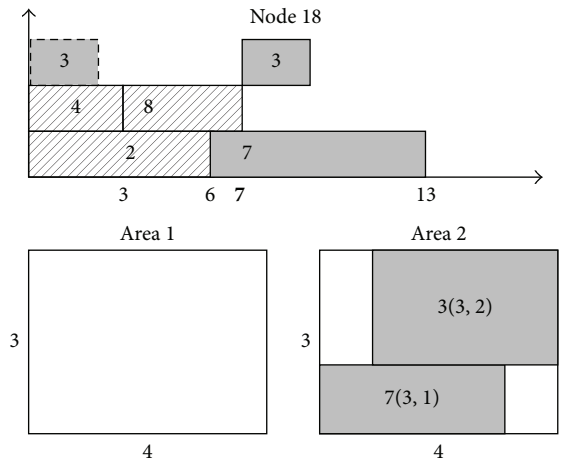

Figure 24: The partial schedule corresponding to node 18.

The dominance rule is also applied to nodes 9 and 12, and the descendants of nodes 2, 3, 4, 5, and 7, as shown in Figure 7.

The same algorithm process is implemented to node 8 and another optimal schedule is produced as shown in Figure 25. The descendants expanded from node 8 are illustrated in Figure 7. As it is better than the previous one shown in Figure 23, it is the best solution for the instance.

5. Conclusion

For the spatial resource scheduling problem, a sophisticated branch and bound algorithm is developed to achieve the optimal solution. The branching is an implicit enumeration based scheme and the lower bound is computed based on the critical path of a network. In order to improve the efficiency of BB-SRCPSP, an effective dominance rule is employed and a heuristic based node selection is adopted. For the optimization of the area arrangement, the maximal space based algorithm is applied. The proposed BB-SRCPSP is the first to solve a spatial resource scheduling problem to its optimal solution and thus is a guideline for the design of a more efficient branch and bound algorithm in this field.

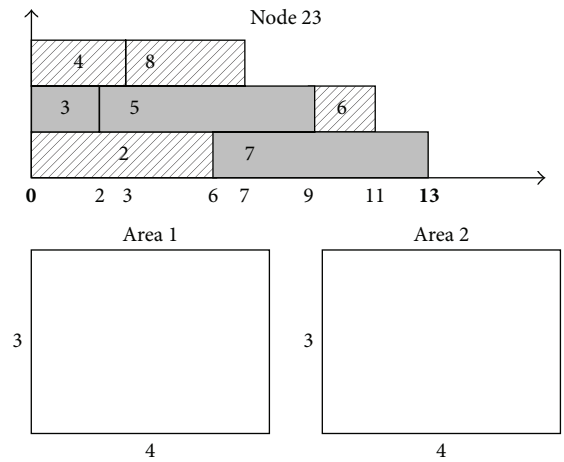

Figure 25: The partial schedule corresponding to node 23.

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 3 reviewers and handling editor for their meticulous reading of the paper and constructive comments which greatly improved the paper.

References

[1] S. Hartmann and D. Briskorn, "A survey of variants and extensions of the resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 207, no. 1, pp. 1–14, 2010.

[2] P. Brucker and S. Knust, "Linear programming and constraint propagation-based lower bound for the RCPSP," *European Journal of Operational Research*, vol. 127, no. 2, pp. 355–362, 2000.

[3] R. Kolisch and R. Padman, "An integrated survey of deterministic project scheduling," *Omega*, vol. 29, no. 3, pp. 249–272, 2001.

[4] Z. He, N. Wang, T. Jia, and Y. Xu, "Simulated annealing and tabu search for multi-mode project payment scheduling," *European Journal of Operational Research*, vol. 198, no. 3, pp. 688–696, 2009.

[5] M. Mika, G. Waligóra, and J. Weglarz, "Tabu search for multi-mode resource-constrained project scheduling with schedule-dependent setup times," *European Journal of Operational Research*, vol. 187, no. 3, pp. 1238–1250, 2008.

[6] S. Hartmann, "A self-adapting genetic algorithm for project scheduling under resource constraints," *Naval Research Logistics*, vol. 49, no. 5, pp. 433–448, 2002.

[7] A. Agarwal, S. Colak, and S. Erenguc, "A Neurogenetic approach for the resource-constrained project scheduling problem," *Computers and Operations Research*, vol. 38, no. 1, pp. 44–50, 2011.

[8] R. Agarwal, M. K. Tiwari, and S. K. Mukherjee, "Artificial immune system based approach for solving resource constraint project scheduling problem," *The International Journal of Advanced Manufacturing Technology*, vol. 34, no. 5-6, pp. 584–593, 2007.

[9] W. Chen, J. Zhang, H. S. Chung, R. Z. Huang, and O. Liu, "Optimizing discounted cash flows in project scheduling-an ant colony optimization approach," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 40, no. 1, pp. 64–77, 2010.

[10] H. Zhang, X. Li, H. Li, and F. Huang, "Particle swarm optimization-based schemes for resource-constrained project scheduling," *Automation in Construction*, vol. 14, no. 3, pp. 393–404, 2005.

[11] M. Ranjbar, B. De Reyck, and F. Kianfar, "A hybrid scatter search for the discrete time/resource trade-off problem in project scheduling," *European Journal of Operational Research*, vol. 193, no. 1, pp. 35–48, 2009.

[12] R. Klein and A. Scholl, "PROGRESS: optimally solving the generalized resource-constrained project scheduling problem," *Mathematical Methods of Operations Research*, vol. 52, no. 3, pp. 467–488, 2000.

[13] J. Coelho and M. Vanhoucke, "Multi-mode resource-constrained project scheduling using RCPSP and SAT solvers," *European Journal of Operational Research*, vol. 213, no. 1, pp. 73–82, 2011.

[14] A. Lova, P. Tormos, M. Cervantes, and F. Barber, "An efficient hybrid genetic algorithm for scheduling projects with resource constraints and multiple execution modes," *International Journal of Production Economics*, vol. 117, no. 2, pp. 302–316, 2009.

[15] S. C. Hu and X. F. Xu, "A production scheduling algorithm for cost optimization," *Computer Integrated Manufacturing Systems*, vol. 9, no. 9, pp. 735–739, 2003.

[16] M. J. Sobel, J. G. Szmerekovsky, and V. Tilson, "Scheduling projects with stochastic activity duration to maximize expected net present value," *European Journal of Operational Research*, vol. 198, no. 3, pp. 697–705, 2009.

[17] K. Neumann and J. Zimmermann, "Procedures for resource leveling and net present value problems in project scheduling with general temporal and resource constraints," *European Journal of Operational Research*, vol. 127, no. 2, pp. 425–443, 2000.

[18] G. Waligóra, "Discrete-continuous project scheduling with discounted cash flows—a tabu search approach," *Computers & Operations Research*, vol. 35, no. 7, pp. 2141–2153, 2008.

[19] Y. Yin, T. C. E. Cheng, and C.-C. Wu, "Scheduling with time-dependent processing times," *Mathematical Problems in Engineering*, vol. 2014, Article ID 201421, 2 pages, 2014.

[20] Y. Q. Yin, W. H. Wu, T. C. E. Cheng, and C. C. Wu, "Single-machine scheduling with time-dependent and position-dependent deteriorating jobs," *International Journal of Computer Integrated Manufacturing*, 2014.

[21] Y. Yin, S.-R. Cheng, and C.-C. Wu, "Scheduling problems with two agents and a linear non-increasing deterioration to minimize earliness penalties," *Information Sciences*, vol. 189, pp. 282–292, 2012.

[22] Y. Yin, D. Xu, K. Sun, and H. Li, "Some scheduling problems with general position-dependent and time-dependent learning effects," *Information Sciences*, vol. 179, no. 14, pp. 2416–2425, 2009.

[23] Y. Q. Yin, M. Liu, J. H. Hao, and M. C. Zhou, "Single-machine scheduling with job-position-dependent learning and time-dependent deterioration," *IEEE Transactions on Systems, Man, and Cybernetics: Systems and Humans*, vol. 42, no. 1, pp. 192–200, 2012.

[24] 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.

[25] C. Park, K.-H. Chung, J.-C. Park, K.-K. Cho, T.-H. Baek, and E.-I. Son, "A spatial scheduling application at the block paint shop in shipbuilding: the HYPOS project," *Production Planning and Control*, vol. 13, no. 4, pp. 342–354, 2002.

[26] R. Varghese and D. Y. Yoon, "Dynamic spatial block arrangement scheduling in shipbuilding industry using genetic algorithm," in *Proceedings of the IEEE 3rd International Conference on Industrial Informatics (INDIN '05)*, pp. 444–449, IEEE, August 2005.

[27] 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.

[28] C. Steiger, H. Walder, and M. Platzner, "Operating systems for reconfigurable embedded platforms: online scheduling of real-time tasks," *IEEE Transactions on Computers*, vol. 53, no. 11, pp. 1393–1407, 2004.

[29] M. Dell'Amico, J. C. D. Díaz, and M. Iori, "The bin packing problem with precedence constraints," *Operations Research*, vol. 60, no. 6, pp. 1491–1504, 2012.

[30] R. Kolisch and A. Sprecher, "PSPLIB—a project scheduling problem library," *European Journal of Operational Research*, vol. 96, no. 1, pp. 205–216, 1997.