Science & Technology Development Journal – Engineering and Technology, 3(SI1):SI50-SI58

Open Access Full Text Article

Research Article

A mathematical model for teamwork scheduling problem in available time windows

Trang Hong Son[1,2], Tran Van Lang[3], Nguyen Huynh-Tuong[1,*]

Use your smartphone to scan this QR code and download this article

ABSTRACT

This paper deals with teamwork scheduling problem in available time windows. This problem has been posed by combining the three constraints are the jobs can split into some sub-jobs which should not be less than a threshold called $split_{min}$, the jobs are only assigned into available time windows and the jobs can be assigned into many people in the organization. Since then the four properties of this problem considered are everyone handles any jobs; a job can be handled by some person at the same time; jobs can be broken down into some sub-jobs; the size of the job/sub-job should not be less than $split_{min}$. The goal aims to determine a feasible schedule that minimizes makespan. And a numerical example is presented to demonstrate the essential constraint with given input data to well define this scheduling problem. Besides the authors proposed a mathematical model to determine the optimal solution by using solvers to solve it and some simple heuristics with computing time less than one second to find the good solutions such as Assignment approach, SPT/LPT rules. All experiments were evaluated on two criteria are the maximum completion time for all jobs and runtime in seconds to determine the solution. These experiments were conducted by the comparison of the lower bound, the exact method based on using CPLEX solver to solve the MILP model, and proposed heuristics. The experimental results show it is very time consuming to determine the optimal solution by CPLEX solver, while the solution found by heuristic algorithms is only good enough.

Key words: parallel machine, splitting job, available time window, MILP model, assignment approach, SPT/LPT rules

[1]*Ho Chi Minh City University of Technology, VNU-HCM, Vietnam*

[2]*Hoa Sen University, Vietnam*

[3]*Institute of Applied Mechanics and Informatics, VAST, Vietnam*

Correspondence

Nguyen Huynh-Tuong, Ho Chi Minh City University of Technology, VNU-HCM, Vietnam

Email: htnguyen@hcmut.edu.vn

History

- Received: 01-8-2019
- Accepted: 23-8-2019
- Published: 13-11-2020

DOI :10.32508/stdjet.v3iSI1.528

Check for updates

Copyright

© VNU-HCM Press. This is an open-access article distributed under the terms of the Creative Commons Attribution 4.0 International license.

VNU-HCM Press

INTRODUCTION

Context

People nowadays face a lot of pressure from life such as family, work, affection, etc., a lot of problems have to be solved in a proper and reasonable way. Many support tools have been created to help us solve problems more easily and conveniently like smartphones, robots, or utilities. And scheduling applications such as Microsoft To-Do, Google Tasks, Apple Reminders, etc. have also been created to help us be able to organize our jobs in the most efficient way. The characteristics of the jobs when scheduling by these tools are the work must be continuous, uninterrupted or broken down. These applications are very powerful in scheduling individual jobs, but it is very difficult to schedule jobs for a group of people or an organization, because each person will have time-windows differently.

In the past, there have been many studies of job scheduling with constraints that jobs can be splittable into many sub-jobs such as resumable studies[1–3], lot sizing studies[4–7], capacitated machine studies[8–11], etc. and there have been also many studies on scheduling in available time windows[12–14]. As a pioneering result among[1–3], Min and Cheng[2] considered a cheduling resumable simple linear deteriorating jobs on a single machine with an availability constraint to minimize makespan. The authors showed this problem is equivalent to a binary integer programming problem and proved it is NP-hard in the ordinary sense, and then show there exists an FPTAS for it by applied the technique of Woeginger. In the lot-sizing scheduling problem, scheduling is focused on integrated production planning and scheduling problem. Wolosewicz *et al.*[5] presented a novel approach for solving an integrated production planning and scheduling problem. The authors proposed a new model integrating lot-sizing decisions and scheduling constraints and a Lagrangian heuristic to solve this model. In 2008, Raut *et al.* [8] addressed the NP-hard cheduling a capacitated single machine with time deteriorating job values. The authors proposed new heuristics based on a multiplicative piecewise metric as an approximation of the slope of job value deterioration. Combining both of these constraints together creates an interesting *NP-Hard* problem that was published at Nguyen *et al.*[15]. This paper proposes to in-

Cite this article : Son T H, Lang T V, Huynh-Tuong N. **A mathematical model for teamwork scheduling problem in available time windows**. *Sci. Tech. Dev. J. – Engineering and Technology;* 3(SI1):50-58.

corporate a further constraint that jobs can be performed by many people in the organization. And the teamwork scheduling problem in available time windows has been posed by combining the three constraints are the jobs can split into some sub-jobs which should not be less than $split_{min}$, the jobs are only assigned into available time windows and the jobs can be assigned into many person in the organization. These properties of this problem are everyone handles any jobs (each person is treated as a working machine in the context of this problem); a job can be handled by some person at the same time; jobs can be broken down into some sub-jobs; the size of job/sub-job should not be less than $split_{min}$.

Notations

The teamwork scheduling problem is denoted by according to Graham et al. [16] as follows:

$$P|splittable,\ split_{min},\ time-window|C_{max}$$

The other notations used in the problem are:

- $J = \{J_1, \dots, J_n\}$: the set of n jobs.
- J_i: the i^{th} job.
- $M = \{M_1, \dots, M_k\}$: the set of k machines.
- M_j: the j^{th} machine.
- $W^j = \{W^j{}_1, \dots, W^j{}_m\}$: the set of m windows for machine M_j.
- $W^j{}_t$ the t^{th} window for machine M_j.
- p_i: the processing time for job J_i.
- C_i: the completion time for job J_i.
- $C_{max} = \max(C_i)$ the maximum completion time for all jobs, also called the "makespan".
- $w^j{}_t$: size of window $W^j{}_t$.
- $b^j{}_t$: the t^{th} break time for machine M_j.

Example

A numerical example demonstrates the essential constraint with the following input data.

- There are $J = \{J_1, J_2, J_3, J_4, J_5, J_6\}$ and processing time for each job is respectively defined $p_1 = 6$, $p_2 = 7$, $p_3 = 8$, $p_4 = 5$, $p_5 = 10$, $p_6 = 4$ (details in Table 1).
- There are 2 machines M_1 and M_2 with available time windows on M_1 are [0,7], [7,12], [12,20], [20, +∞) and available time windows on M_2 are [0,5], [5,11], [11,18], [18, +∞) (details in Table 2).
- Based on available time windows of machines, a time window axis is created corresponding to break time at t = 7, t = 12, t = 20 on M_1 and t = 5, t = 11, t = 18 on M_2 as Figure 1.

Table 1: Jobs

Jobs	Processing time
J_1	6
J_2	7
J_3	8
J_4	5
J_5	10
J_6	4

Table 2: Machines

	Windows	Available time
Machine 1	$W^1{}_1$	[0,7]
	$W^1{}_2$	[7,12]
	$W^1{}_3$	[12,20]
	$W^1{}_4$	[20, +∞)
Machine 2	$W^2{}_1$	[0,5]
	$W^2{}_2$	[5,11]
	$W^2{}_3$	[11,18]
	$W^2{}_4$	[18, +∞)

Let $split_{min} = 3$, the possible solutions for this problem are as follows.

- A feasible solution with $C_{max} = 24$ as described in Table 3 and Figure 2.

Table 3: A feasible solution

Job	Window	Processing time	Time window
J_1	$W^1{}_1$	6	[0-6]
J_2	$W^2{}_1$	4	[0-4]
J_2	$W^2{}_2$	3	[5-8]
J_3	$W^1{}_2$	5	[7-12]
J_3	$W^2{}_2$	3	[8-11]
J_4	$W^2{}_3$	5	[11-16]
J_5	$W^1{}_3$	7	[12-19]
J_5	$W^2{}_4$	3	[18-21]
J_6	$W^1{}_4$	4	[20-24]

- An optimal solution with Cmax = 21 as described in Table 4 and Figure 3.

In this paper, the authors propose a MILP model which can be solved by the solver to determine the optimal solution and some simple heuristics with computing time less than one second to find the good solutions. The experimental results illustrate the performance of the proposed heuristics in comparing with the exact method implemented by MILP solvers. This paper is organized as follows the resolution methods are presented in Section 2, Section 3 gives computational results, discussion and conclusion are shown in Section 4 and Section 5.

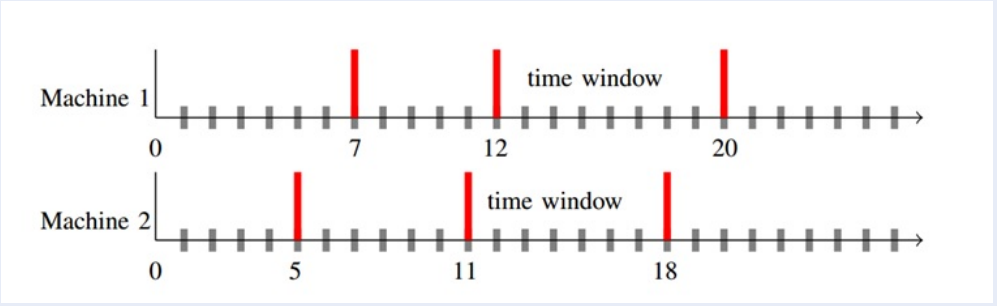

Figure 1: Demonstration of available time windows on time axis

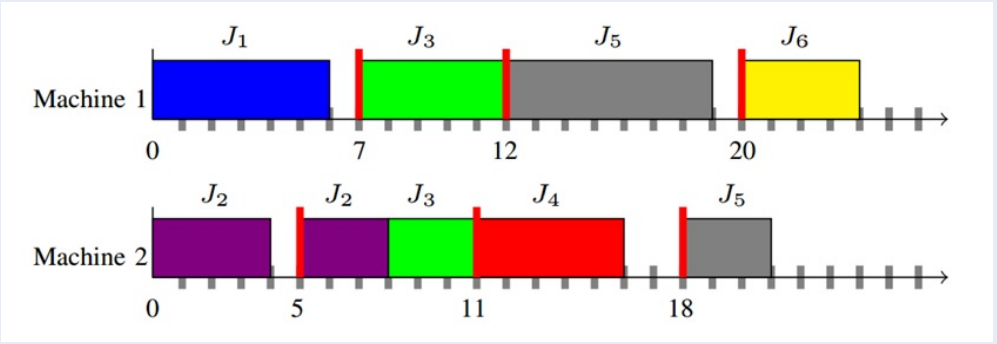

Figure 2: A feasible solution

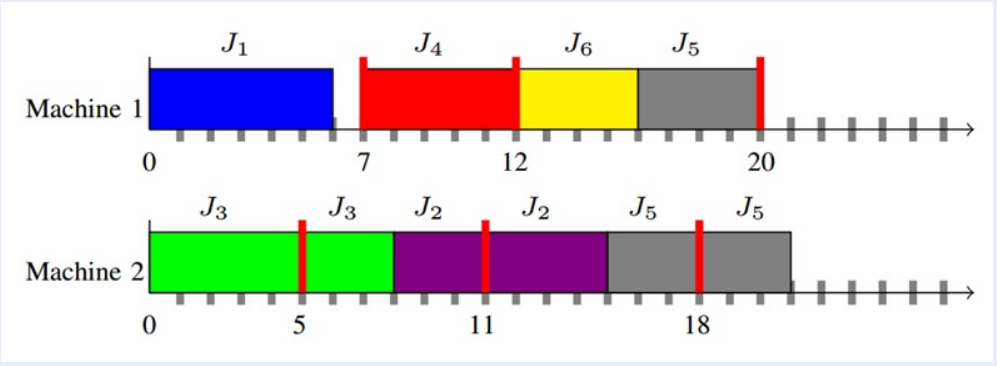

Figure 3: An optimal solution

RESOLUTION METHODS

Mathematical model

The proposed MILP model is presented as:

a) Decision variables:

- $x_{i,j,t} \in \{0, 1\}$: is 0 if J_i is not assigned into M_j at W_t, otherwise is 1.
- $y_{i,j,t}$: the processing time for J_i in M_j at W_t corresponding to $x_{i,j,t}$.
- $s_{i,j,t}$: the starting-time for J_i in M_j at W_t corresponding to $x_{i,j,t}$.

b) Intermediate variables are:

- $c_{i,j,t} = s_{i,j,t} + y_{i,j,t}$: the completion time for J_i in M_j at W_t.
- $C_i = \max_{t=1,\ldots,m} (c_{i,j,t})$: the completion time for J_i.
- $C_{max} = \max_{i=1,\ldots,n} (C_i)$: the maximum completion time for all jobs.

Table 4: An optimal solution

Job	Window	Processing time	Time window
J_1	$W^1{}_1$	6	[0-6]
J_2	$W^2{}_2$	3	[8-11]
J_2	$W^2{}_3$	4	[11-15]
J_3	$W^2{}_1$	5	[0-5]
J_3	$W^2{}_2$	3	[5-8]
J_4	$W^1{}_2$	5	[7-12]
J_5	$W^2{}_3$	3	[15-18]
J_5	$W^1{}_3$	4	[16-20]
J_5	$W^2{}_4$	3	[18-21]
J_6	$W^1{}_3$	4	[12-16]

- $bv_{i\_1,i\_2,j,t} \in \{0, 1\}$: used to convert from OR constraint to AND constraint.

c) Objective function: $min(C_{max})$

d) Constraints:

$$\sum_{j=1}^{k}\sum_{t=1}^{m} y_{i,j,t} = p_i,\ \forall i = 1...n \tag{1}$$

$$\sum_{i=1}^{n} y_{i,j,t} \le w_{j,t}, \forall j = 1...k, \forall t = 1...m \tag{2}$$

$$split_{min} \times x_{i,j,t} \le y_{i,j,t} \le p_i \times x_{i,j,t},\ \forall i = 1...n, \forall j = 1...k, \forall t = 1...m \tag{3}$$

$$b_{j,t} \times x_{i,j,t} \le s_{i,j,t} \le INF \times x_{i,j,t},\ \forall i = 1...n, \forall j = 1...k, \forall t = 1...m \tag{4}$$

$$b_{j,t} \le s_{i,j,t} \le b_{j,t+1} - y_{i,j,t},\ \forall i = 1...n, \forall j = 1...k, \forall t = 1...m \tag{5}$$

$$\begin{cases} c_{i_1,j,t} - s_{i_2,j,t} \le INF \times bv_{i_1,i_2,j,t}, \\ \forall i_1 \ne i_2 = 1...n, \forall j = 1...k, \forall t = 1...m \\ c_{i_2,j,t} - s_{i_1,j,t} \le INF \times (1 - bv_{i_1,i_2,j,t}), \\ \forall i_1 \ne i_2 = 1...n, \forall j = 1...k, \forall t = 1...m \end{cases} \tag{6}$$

Heuristics

Based on the given constraints, the answering two following questions will determine a feasible solution. There are which job/sub-job be assigned into an available time window and how length of this job/sub-job? To answer above questions, three proposed algorithms are: Assignment, Shortest Processing Time and Longest Processing Time. And some notations are used in this section, there are rj_i is remaining time for job J_i and rw_t is the remaining size of window $W^j{}_t$.

Assignment - ASGN

The pseudocode of this heuristic is traverse each job in list jobs:

- firstly determine machine which completion time is minimum (Algorithm 2);
- then traverse each window of that machine from left to right: the job is considered to assign into the current window (see more detail in Algorithm 1).

Based on the input data in Tables 1 and 2, the schedule from ASGN algorithm shows as Figure 6.

Shortest Processing Time - SPT

The only difference between SPT algorithm and ASGN algorithm is the list jobs input of the ASGN algorithm have no order, while the list jobs input of SPT algorithm will have the order of processing time of jobs gradually increasing. This means that jobs with smaller processing time will be prioritized to assign into available time windows. And based on the input data in Tables 1 and 2, the schedule from the SPT algorithm shows as Figure 7.

Longest Processing Time - LPT

The LPT algorithm is the opposite of the SPT algorithm, the jobs which have larger processing time will be prioritized to handle first. And based on the input data in Tables 1 and 2, the schedule from the LPT algorithm shows as Figure 8.

EXPERIMENTAL RESULTS

Dataset

We created 9 tuples (n, k, $split_{min}$) by combine of n = {10, 20, 30}, k = {2, 3, 4} and $split_{min}$ = 3. And for each a tuple, according to the way of Hariri & Potts [17] and Baptiste [18], we generated 10 sample instances. In that:

- p_i is an integer which randomly generated from uniform distribution in [$split_{min}$, 24].
- w_t is an integer which randomly generated from uniform distribution in [$split_{min}$, 12].

An example of 10 sample instances created in a tuple set (10,2,3) is shown in Table 5.

Lower bound

Note that a feasible solution without any idle-time is the optimal solution. So we proposed lower bound calculated by the formula:

$$LB = \lceil \frac{\Sigma p_i}{k} \rceil$$

Algorithm 1: ASGN, with $O(n \times m)$

input: $Jobs$: list jobs sorted in non-order
$Macs$: list machines

```
1  begin
2      foreach job J_i ∈ Jobs do
3          machine ← GetMachineHasReady(Macs);
4          Put all windows of machine into list Wins;
5          foreach window W_t ∈ Wins do
6              if rj_i ≥ split_min and rw_t ≥ split_min then
7                  if rj_i ≤ rw_t then
8                      Assign job J_i with the size of rj_i into
                        window W_t;
9                      break;
10                 else
11                     if rj_i − rw_t ≥ split_min then
12                         Assign job J_i with the size of rw_t
                            into window W_t;
13                         break;
14                     else if rj_i − split_min ≥ split_min then
15                         Assign job J_i with the size of
                            rj_i − split_min into window W_t;
16                         break;
17                 end
18             end
19         end
20     end
21 end
```

Figure 4: Algorithm 1

Benchmarking

The CPLEX v12.7.1 solver was selected to solve the proposed MILP model while heuristic algorithms were implemented on .NET framework v4.5 which evaluated on two criteria:

- C_{max} value is found by CPLEX solver or heuristic algorithms.
- Runtime (t) in seconds.

Table 6 shows the summary of experimental results as follows.

DISCUSSION

Our experimental results in Table 6 indicate the percentage gap between C*max (optimal value) and LB (lower bound value) is tiny (about 0.29%). This gap has also another mean that the number of idle times should be used in an optimal solution. It may be inferred that with these inputs it is not easy to determine a good approximating solution by trivial heuristics. Finding the optimal solution by CPLEX solver takes much time, and more exponential time increases when larger the number of jobs. And within the time

Algorithm 2: GetMachineHasReady, with $O(k)$

input: $Macs$: list machines

1 **begin**

2 | Return a machine in $Macs$, with completion time is minimum;

3 **end**

Figure 5: Algorithm 2

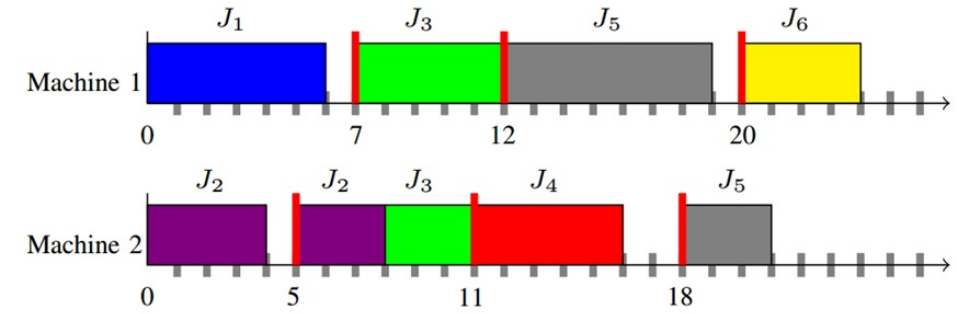

Figure 6: ASGN with C_{max} = 24

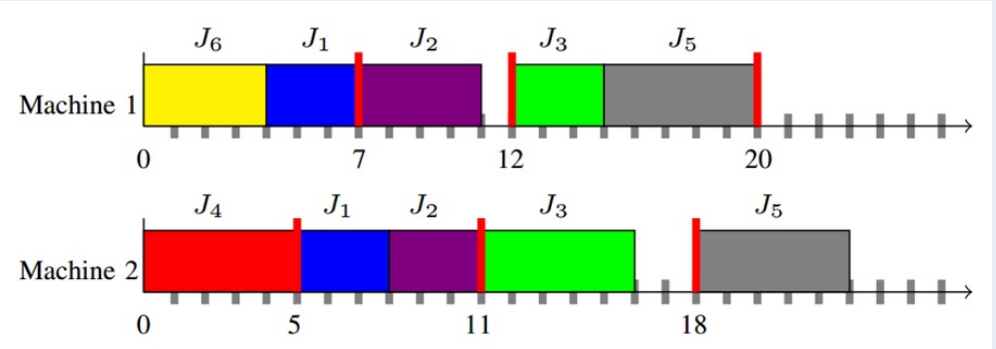

Figure 7: SPT-order with C_{max} = 23

limit (less than 10 minutes) for finding an acceptable solution, the tuple set (n=30, k=4) is the maximum threshold that the CPLEX solver can determine an optimal solution.

For heuristic algorithms, the time to find solutions is very fast (about 0 second approximately), but no heuristic can determine any optimal value for a tuple set. In comparing between them by counting the best one for each tuple set, ASGN achieved 0% the best, SPT achieved about 22% the best, and LPT achieved about 78% the best. By calculating the total Cmax values - 9314, 9489, 9497 correspondingly in Table 6 - which related to the total idle times used in overall for each heuristic, the LPT algorithm is the best one among the proposed heuristic algorithms. Although these differences are negligible (2% at most) while all

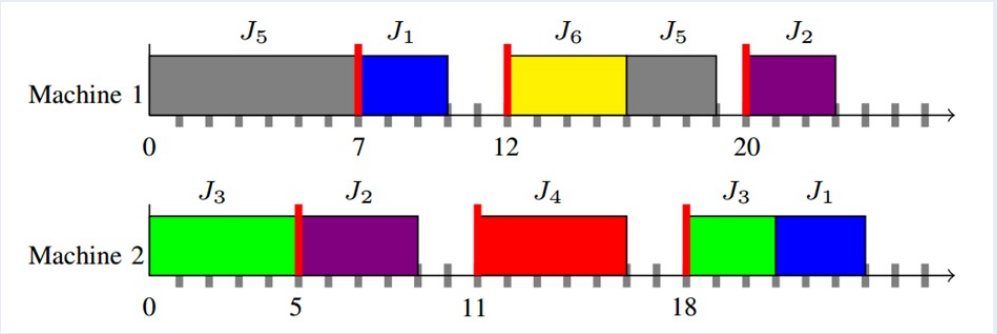

Figure 8: LPT-order with C_{max}= 24

Table 5: Input data in tuple set (10,2,3)

inst.	Jobs	Machine 1	Machine 2
1	p_i = 3 3 16 6 9 19 4 14 3 23	w^1_t = 8 4 3 8 8 6 4 8 4	w^2_t = 8 5 5 4 11 3 10 9
2	p_i = 22 18 9 17 18 10 13 16 16 9	w^1_t = 10 10 11 5 6 12 8 10 5	w^2_t = 9 4 5 6 5 11 7 10 6 10 12
3	p_i = 18 4 15 6 7 20 19 9 22 20	w^1_t = 9 12 12 3 6 11 5 5 12	w^2_t = 5 8 4 11 3 7 9 6 3 11 5
4	p_i = 16 5 22 18 15 18 6 18 4 15	w^1_t = 5 5 11 7 3 5 5 5 11 12	w^2_t = 4 7 7 7 6 8 5 8 9 8
5	p_i = 11 17 12 16 6 13 3 9 13 17	w^1t = 10 8 4 6 3 6 5 7 12	w^2_t = 5 6 4 4 11 4 4 9 7 6
6	p_i = 3 13 16 3 14 7 15 21 8 8	w^1_t = 8 5 12 10 5 5 8 5	w^2_t = 4 6 3 5 10 8 7 4 5 7
7	p_i = 14 14 3 21 16 23 13 7 5 14	w^1_t = 9 4 7 7 3 5 7 3 7 6 12	w^2_t = 9 8 12 8 3 6 10 9
8	p_i = 3 15 8 19 24 17 23 14 5 10	w^1_t = 7 4 9 9 11 12 12 10	w^2_t = 3 7 3 4 12 9 8 9 8 4 6
9	p_i = 9 4 3 19 23 22 20 23 14 17	w^1_t = 11 11 10 12 9 10 11 4	w^2_t = 3 6 7 7 12 11 7 11 12 8
10	p_i = 6 12 3 12 16 14 5 5 10 20	w^1_t = 7 6 12 9 6 12	w^2_t = 6 11 8 5 8 10 10

Table 6: Summary of experimental results

id	n	k	LB	CPLEX		ASGN		SPT		LPT	
				C^*_{max}	t	C_{max}	t	C_{max}	t	C_{max}	t
1	10	2	639	644	3.41	688	0	677	0	687	0
2	10	3	464	467	2.78	513	0	502	0	504	0
3	10	4	355	357	4.14	395	0	402	0	387	0
4	20	2	1375	1380	28.66	1461	0	1469	0	1457	0
5	20	3	845	848	60.23	918	0	917	0	905	0
6	20	4	679	680	78.74	746	0	753	0	728	0
7	30	2	2062	2064	409.61	2207	0	2248	0	2162	0
8	30	3	1357	1360	443.1	1472	0	1452	0	1438	0
9	30	4	979	980	598.1	1089	0	1077	0	1046	0
SUM			8755	8780	-	9489	-	9497	-	9314	-

Notes:
LB is lower bound value and C^*_{max} is optimal value
each tuple is the sum of 10 different instances

of these heuristics can determine a good solution in about 13% gap maximum from optimal solution.

CONCLUSION

In this paper we considered the teamwork scheduling problem with availability time windows, splittable jobs and min-split constraint so as to minimize the makespan. The mathematical MILP model is given to achieve the optimal goal of this problem. Three proposed heuristic algorithms to determine a good solution in about 13% gap maximum from the optimal solution are Assignment approach, Shortest Processing Time and Longest Processing Time rules. The experimental results show that it is very time consuming to find the optimal solution by CPLEX solver, while the solution found by heuristic algorithms is only good enough. Future works may address applying some evolutionary algorithms such as metaheuristic on the larger instances to improve the quality of solutions. Besides adding more constraints to this teamwork scheduling problem fits more with reality.

ABBREVIATIONS

MILP - mixed integer linear programming
NP - nondeterministic polynomial time
FPTAS - fully polynomial time approximation scheme
ASGN - assignment
SPT - shortest processing time
LPT - longest processing time
LB - lower bound
CPLEX - IBM ILOG CPLEX Optimization Studio

COMPETING INTERESTS

The authors declare that there is no conflict of interests regarding the publication of this paper.

AUTHORS' CONTRIBUTION

Trang Hong Son considered the problem, proposed methods and conducted experiments on this problem. Tran Van Lang reviewed the presentation of this paper. And Nguyen Huynh-Tuong gave some suggestions on the MILP model.

REFERENCES

1. Gawiejnowicz S, Kononov A. Complexity and approximability of scheduling resumable proportionally deteriorating jobs. European Journal of Operational Research. 2010;200:305–308. Available from: https://doi.org/10.1016/j.ejor.2008.12.014.
2. Ji M, Cheng TCE. Scheduling resumable simple linear deteriorating jobs on a single machine with an availability constraint to minimize makespan. Computers and Industrial Engineering. 2010;59:794–798. Available from: https://doi.org/10.1016/j.cie.2010.08.005.
3. Fan B, Li S, Zhou L, Zhang L. Scheduling resumable deteriorating jobs on a single machine with non-availability constraints. Theoretical Computer Science. 2011;412:275–280. Available from: https://doi.org/10.1016/j.tcs.2010.09.017.
4. Park YW, Klabjan D. Lot sizing with minimum order quantity. Discrete Applied Mathematics. 2015;181:235–254. Available from: https://doi.org/10.1016/j.dam.2014.09.015.
5. Wolosewicz C, Dauzre-Prs S, Aggoune R. A lagrangian heuristic for an integrated lot-sizing and fixed scheduling problem. European Journal of Operational Research. 2015;244:3–12. Available from: https://doi.org/10.1016/j.ejor.2015.01.034.
6. Olekw-Szlapka J, Pawowski G. Scheduling and lot sizing problems for variable range of products using ga-based method. IFAC-PapersOnLine. 2016;49:662–667. Available from: https://doi.org/10.1016/j.ifacol.2016.07.786.
7. Alem D, Curcio E, Amorim P, Lobo BA. A computational study of the general lot-sizing and scheduling model under demand uncertainty via robust and stochastic approaches. Computers and Operations Research. 2018;90:125–141. Available from: https://doi.org/10.1016/j.cor.2017.09.005.
8. Raut S, Swami S, Gupta JND. Scheduling a capacitated single machine with time deteriorating job values. International Journal of Production Economics. 2008;114:769–780. Available from: https://doi.org/10.1016/j.ijpe.2007.12.014.
9. James RJW, Almada-Lobo B. Single and parallel machine capacitated lotsizing and scheduling: New iterative mip-based neighborhood search heuristics. Computers and Operations Research. 2011;38:1816–1825. Available from: https://doi.org/10.1016/j.cor.2011.02.005.
10. Shim IS, Kim HC, v HH, Lee DH. A two-stage heuristic for single machine capacitated lot-sizing and scheduling with sequence-dependent setup costs. Computers and Industrial Engineering. 2011;61:920–929. Available from: https://doi.org/10.1016/j.cie.2011.06.002.
11. Boonmee A, Sethanan K. A glimpse for multi-level capacitated lot-sizing and scheduling problem in the poultry industry. European Journal of Operational Research. 2016;250:652–665. Available from: https://doi.org/10.1016/j.ejor.2015.09.020.
12. Zhong X, Ou J, Wang G. Order acceptance and scheduling with machine availability constraints. European Journal of Operational Research. 2014;232:435–441. Available from: https://doi.org/10.1016/j.ejor.2013.07.032.
13. Fan J, Lu X, Liu P. Integrated scheduling of production and delivery on a single machine with availability constraint. Theoretical Computer Science. 2015;562:581–589. Available from: https://doi.org/10.1016/j.tcs.2014.10.047.
14. Liu P, Lu X. Integrated production and job delivery scheduling with an availability constraint. International Journal of Production Economics. 2016;176:1–6. Available from: https://doi.org/10.1016/j.ijpe.2016.03.006.
15. Nguyen VH, Tuong NH, Nguyen HP, Nguyen TH. Single machine scheduling with splitable jobs and availability constraints. REV Journal on Electronics and Communications. 2013;3(1-2):21–27. Available from: https://doi.org/10.21553/rev-jec.51.
16. Graham RL, Lawler EL, Lenstra JK, Kan AHGR. Optimization and approximation in deterministic sequencing and scheduling: a survey. Annals of Discrete Mathematics. 1979;5:287–326. Available from: https://doi.org/10.1016/S0167-5060(08)70356-X.
17. Hariri AMA, Potts CN. Single machine scheduling with deadlines to minimize the weighted number of tardy jobs. Management Science. 1994;40(12):1712–1719. Available from: https://doi.org/10.1287/mnsc.40.12.1712.
18. Baptiste P, Croce FD, Grosso A, T'Kindt V. Sequencing a single machine with due dates and deadlines: An ilp-based approach to solve very large instances. Journal of Scheduling. 2010;13(1):39–47. Available from: https://doi.org/10.1007/s10951-008-0092-6.

Tạp chí Phát triển Khoa học và Công nghệ – Kĩ thuật và Công nghệ, 3(SI 1):50-58

Open Access Full Text Article

Bài nghiên cứu

Mô hình toán học cho bài toán lập lịch làm việc nhóm trong những khung cửa sổ thời gian

Trang Hồng Sơn[1,2], Trần Văn Lăng[3], Nguyên Huỳnh-Tường[1,*]

Use your smartphone to scan this QR code and download this article

TÓM TẮT

Bài báo đề cập đến bài toán lập lịch làm việc nhóm trong những khung cửa sổ thời gian. Bài toán này được đặt ra bằng cách kết hợp ba ràng buộc đó là các công việc có thể được chia nhỏ nhưng không thể nhỏ hơn một ngưỡng gọi là $split_{min}$, các công việc chỉ được sắp xếp vào những khung cửa sổ thời gian khả dụng, và các công việc có thể được phân công cho nhiều người trong nhóm. Do đó bốn tính chất của bài toán này được xem xét là mọi người đều xử lý bất kỳ công việc nào; một công việc có thể được xử lý bởi một số người cùng một lúc; công việc có thể được chia thành các công việc nhỏ hơn; kích thước của các công việc không thể nhỏ hơn $split_{min}$. Mục tiêu chính của bài toán là tìm ra một lịch làm việc khả thi sao cho tất cả các công việc được hoàn thành sớm nhất có thể. Để xác định rõ bài toán lập lịch đang xem xét, một ví dụ bằng số được trình bày để mô phỏng các ràng buộc thiết yếu với dữ liệu đầu vào đã cho. Bên cạnh đó, các tác giả đã đề xuất một mô hình toán học được giải bởi các solvers để tìm ra lời giải tối ưu và một số phương pháp heuristic đơn giản để tìm ra các lời giải tốt như Assignment approach, Shortest Processing Time, và Longest Processing Time rules. Tất cả các thực nghiệm được đánh giá theo hai tiêu chí là thời gian hoàn thành tối đa cho tất cả các công việc và thời gian để xác định lời giải cho bài toán. Những thực nghiệm này được thực hiện bằng cách so sánh giá trị lower bound, phương pháp chính xác dựa trên mô hình MILP và các phương pháp heuristic được đề xuất. Các kết quả thử nghiệm cho thấy rất tốn thời gian để tìm ra lời giải tối ưu bằng CPLEX solver, trong khi lời giải tìm thấy bằng thuật toán heuristic chỉ đủ tốt.

Từ khoá: máy song song, chia nhỏ công việc, cửa sổ thời gian khả dụng, mô hình MILP, phương pháp phân bổ, quy tắc SPT/LPT

[1]*Trường Đại học Bách khoa, Đại học Quốc gia TP.HCM, Việt Nam*

[2]*Trường Đại học Hoa Sen, Việt Nam*

[3]*Viện Cơ học và Tin học ứng dụng, VAST, Việt Nam*

Liên hệ

Nguyên Huỳnh-Tường, Trường Đại học Bách khoa, Đại học Quốc gia TP.HCM, Việt Nam

Email: htnguyen@hcmut.edu.vn

Lịch sử

- Ngày nhận: 01-8-2019
- Ngày chấp nhận: 23-8-2019
- Ngày đăng: 13-11-2020

DOI :10.32508/stdjet.v3iSI1.528

Check for updates

Bản quyền

© ĐHQG Tp.HCM. Đây là bài báo công bố mở được phát hành theo các điều khoản của the Creative Commons Attribution 4.0 International license.

VNU-HCM Press

Trích dẫn bài báo này: Sơn T H, Lăng T V, Huỳnh-Tường N. **Mô hình toán học cho bài toán lập lịch làm việc nhóm trong những khung cửa sổ thời gian**. *Sci. Tech. Dev. J. - Eng. Tech.;* 3(SI1):50-58.