A Near-Optimal Solution Method of Multi-Item Multi-Process Dynamic Lot Size Scheduling Problem*

Kenji MURAMATSU**, Aditya WARMAN**
and Minoru KOBAYASHI**

This paper addresses a multi-item multi-process dynamic lot size scheduling problem with general product structure and setup time. In this problem, there exist various heterogeneous decision features such as lot sizing, lot sequencing, dispatching, and so on. We present a near-optimal solution method, which we call a narrow sense Lagrangian decomposition coordination method of solving all decision features involved in this problem simultaneously without specifying or awaking to them one by one. First, splitting the planning horizon into very small time-slots, for any item on any machine at any timeslot we denote a state of processing by using a binary decision variable which takes a value of unity if it is processed, and else then zero. Second, dealing with the transition of the inventory state of each item and time transition of each setup explicitly, we formulate the problem into a multi-dimensional dynamic optimization problem with constraints. Third, paying attention to the existence of the interaction constraints, we decompose the whole problem into item-based sub problems that can be reformulated into dynamic programming of one dimension to dissolve the curse of dimensionality. At the aim of guaranteeing the decomposability, we formulate the problem by echelon inventory. The computational procedure consists of solving sub problems for given Lagrange multiplier values and of coordinating those values. Finally, we verify the presented method by using a numerical model.

Key Words: Dynamic Lot Size Scheduling, Optimization, Multi-Item Multi-Process Production System, Lagrangian Decomposition Coordination Method, Setup Time

1. Introduction

1.1 Problem

A production system with multiple processes is processing various items (involving parts, semi-products, and final products). Each process has different machines. Any order (outer demand) with due date is given as a shipment requirement datum at its corresponding timeslot. Exploding these data to items based on the bill of materials generates the requirements of all items.

The shop type that we treat is flow, however, the repetitive processing by the same machine (e.g., item 10 and 11 on machine 6 in Fig. 1) and reverse flow

(e.g., from item 9 to item 7 in Fig. 1) are allowed. Multiple different machines are available for each item. Product structure to be treated is general. Therefore, in a system there exist many items, which are produced of multiple items, and then are built in multiple items on the next process (that have immediate predecessors and immediate successors simultaneously in terminology of graph theory). For any processing, setup time and setup cost incur, which depends on the item to be processed and the machine (This paper does not treat sequence dependent setup). Processing time also depends on item and machine and is proportional to quantity. For any work-in-process item, holding cost incurs and is proportional to quantity and interval.

We would like to attain the schedule for all of the machines and items in the system so as to minimize the whole cost consisting of setup and holding cost over the planning horizon without allowing the short-

* Received 4th October, 2002 (No. 02-4247)

** Department of Management System Engineering, Tokai University, Hiratsuka, Kanagawa 259-1292, Japan. E-mail: kenji@keyaki.cc.u-tokai.ac.jp

Fig. 1 System diagram

age of item or delay. This problem belongs to dynamic lot size scheduling problem in the sense that decision features of lot sizing and others is not time-invariant. In this paper, specifying this problem as a dynamic lot size scheduling problem of multi-item, multi-process, and different machines with sequence independent set up time and general product structure, we treat this dynamic optimization problem.

1.2 Objective of this study

In this problem, there exist various heterogeneous decision features such as lot sizing, lot sequencing, dispatching, and so on. The objective of this paper is to present a method for solving all of those decision features simultaneously without specifying them one by one nor even awaking to them. For this purpose, we propose a near-optimal solution method that we call a narrow sense Lagrangian decomposition coordination (LDC) method.

LDC method is, as well known, one of the Lagrangian relaxation methods (Lagrange multiplier methods), which attains both of optimization or near-optimization and satisfaction of constraints simultaneously by means of iterative computation consisting of optimization of sub-problems and of coordination of the Lagrange multipliers after relaxing the constraint and decomposing the problem into sub-optimization problems to dissolve the curse of dimensionality.

Unfortunately, in our problem, the lower bound of the objective function value obtained by exploiting the largest Lagrangian value is quite low. Instead, we present the second means for a lower bound in section 6.3.

1.3 Characteristics of presented solution method

This solution method consists of the following three; formulation of the problem into a binary programming model by very small grid of timeslot, utilization of the echelon inventory, and the LDC method in the narrow sense stated later. First, we formulate the problem into binary programming model with high resolution by using only binary decision variables δ_{it}^k (c.f. definition in section 2.1) after ticking the planning horizon. The advantage of this formulation is that we can formulate the problem without specifying various heterogeneous decision features involved in the problem one by one. Second, by formulating the work-in-process inventory transition equation by use of echelon inventory to erase the decision variables related to the items of the immediate successors, we guarantee the independency of decision variables among items. Our solution method is assumed to be of practical use, so, in the case of large-scale we cannot solve the problem within the reasonable limits of computer memory and computation time unless it is decomposed into item based sub-problems to dissolve the curse of dimensionality. The utilization of echelon inventory is indispensable for decomposition. As a result, however, it is inevitable of the shortage of the real work-in-process inventory of each item. Therefore, in order to dissolve this, we impose the constraints to prohibit its shortage on interaction constraints of the model.

Third, as to actual computing, after the item-based decomposition of the problem into sub-problems, we attain a nearoptimal solution by iterative computation cycle not involving any supplementary calculation other than conditional optimization of sub-problems by dynamic programming and coordination of the Lagrange multipliers.

In the conventional methods on scheduling, it is usual to solve the lot sequence under the condition put on the lot size to be time invariant and fixed in advance, or on the other hand, to treat the other remaining decision features putting some assumptions on the lot sequence. According to the method we present, however, we can treat the whole problem without putting any of these artificial constraints. Moreover, we can attain the solution as to all of the decision features involved in the problem such as lot sizing, lot sequencing, dispatching, and so on simultaneously just only when the computation has been finished. This is the main characteristic of this solution method.

Here, we mention the background of presenting this method. Putting any artificial constraint on the solution space in advance just for the sake of solving the original problem, in some occasions it will become

only the obstacle of solution method against the intention. Moreover, even if it were able to obtain the optimal solution under those artificial constraints, it would not be clear of the optimality for the original problem. Furthermore, on corresponding to the change of the situation such as dynamic demand, urgent order, and machine break down in quick, we could not deny that these would be the obstacles. Therefore, our solution method in which we impose none of these artificial constraints would be expected to have the advantages on scheduling in changing business situation.

1.4 Brief review of the conventional studies

The problem treated in this paper belongs to multi-item multi-process scheduling problems with general product structure. So, first, mentioning them, these are classified into lot sizing problem and dynamic lot size scheduling problem, and between the former and the latter there are big difference. It is because that the decision feature of the former is fundamentally confined to that of lot sizing, and the latter has lot sequence decision feature other than that of the former, and yet both of lot sizing and lot sequencing should be treated under the situation of dynamic demand and work-in-process inventory transition over multi-process. As to the former a lot of work is known such as Refs.(1)–(3). On the other hand, as to the latter, its study has been just at the beginning Refs.(4), (5) and there has not been the paper except for Ref.(4). In addition, extending the method having been developed to solve the former to the latter, its computational procedure would become too complicated to expect the solution.

Indeed, along the line of this procedure, Kimms⁽⁶⁾ has presented an approximation method based on the combination of backward oriented heuristic and genetic algorithm after formulating a problem without setup time into multi-level PLSP stated below. In his paper, he has reported its corresponding numerical results for the quite small examples of processing one part, two semi-products and one final product on two different machines the percentage of the problem unsolved increased as the rate of work load increased and as it reached to 70 percents of the capacity, 42 percents of all the problems could not be solved. As a feature of problem decomposition is not treated in this solution procedure, it is not to be denied that the computation quantity for solution increases abruptly as problem scale increases. On the other hand, in the method we present, the computation quantity is settled in linear order as to the number of items.

Next, we make clear the relation between this study and the conventional studies with three above mentioned concepts constituting our method as key-

words. The literatures related to single process lot sizing and scheduling problem are treated in detail in Refs.(5)–(8) and hence we will omit them from this review.

As to problem formulation by use of binary decision variables and very small timeslots, there exist the stream of Discrete Lot sizing and Scheduling Problem (DLSP)⁽⁹⁾, Proportional Lot sizing and Scheduling Problem (PLSP)⁽¹⁰⁾, and General Lot sizing and Scheduling Problem (GLSP)⁽¹¹⁾. Any of these generates the decision feature of lot sizing and that of sequencing one after another. All of these studies are confined to the formulation of single process problem.

Muramatsu⁽¹⁸⁾ developed a LDC method of solving both decision features simultaneously without awakening to them. The method presented in this study is the extension of Ref.(18) to a multi-item multi-process problem with general product structure.

Independent treatment of inventory transition equation of each item by echelon inventory goes back to Ref.(12). Thereafter, a lot of study uses it as in Refs.(2), (3) and others.

Application of LDC method on scheduling problem has been already seen in Ref.(13). Thereafter there has been a lot of study as in Refs.(3), (9) and (14).

After all, summarizing the relation between conventional studies and our study leads to the following. As to any of single-process dynamic lot sizing problem and dynamic lot size scheduling problem by binary programming model and DLSP, formulation of inventory transition equation by echelon inventory, and LDC method on scheduling problem, there have been the studies respectively. However, the study succeeded in solving of a multi-item multi-process dynamic lot size scheduling problem with general product structure has not been known except for the paper of Kimms that treated small ones. Therefore, this paper is the first one that presents a near-optimal solution method for this problem.

2. Formulation

2.1 Symbols

Indices and their sets

i : item number, $i \in I = \{1, 2, \dots, n_I\}$, however $i = 0$ denotes the market (out of system)

k : machine number, $k \in K = \{1, 2, \dots, n_K\}$

t : timeslot number, $t \in T = \{1, 2, \dots, n_T\}$

A : set of work-in-process items

$P(i)$: set of items preceding to item i

$S(i)$: set of items succeeding to item i

$K(i)$: set of machines which can process item i

$M(k)$: set of items which can be processed by machine k

Input data

- x_{i0} : actual inventory of item i at timeslot $t=0$
 δ_{i0}^k : initial state of item i on machine k at timeslot $t=0$
 s_{i0}^k : setup time for item i on machine k
 s_{i0}^k : remaining time of setup for item i on machine k at timeslot $t=0$
 r_{i0t} : external demand for item i at timeslot t
 ρ_{ij} : requirement of item i per 1 unit of item j
 p_i^k : operation quantity of item i on machine k per one timeslot
 c_i^k : setup cost of item i on machine k
 h_i : inventory holding cost per timeslot for value newly added by operating of item i , that is echelon inventory holding cost
 r_{ijt} : requirements of item i being used for operating item j at timeslot t
 $r_{i \cdot t}$: requirements of item i at timeslot t
 τ_{ij} : lead time or moving time between processes when item i is inserted in item j

Decision and state variables

- δ_{it}^k : decision variable that denotes operation of item i on machine k at timeslot t , if item i is processed at timeslot t on machine k , $\delta_{it}^k=1$, else then 0, where δ_{it} is a vector whose k -th element is δ_{it}^k , $k \in K(i)$, and $\delta_i = (\delta_{i1}, \dots, \delta_{it}, \dots, \delta_{inT})$, $\delta = (\delta_1, \dots, \delta_i, \dots, \delta_{nI})$.
 s_{it}^k : remaining time of setup of machine k at timeslot t for item i , $0 \leq s_{it}^k \leq s_{i0}^k$, where, s_{it} is a vector whose k -th element is s_{it}^k , $k \in K(i)$, and $s_i = (s_{i1}, \dots, s_{it}, \dots, s_{inT})$, $s = (s_1, \dots, s_i, \dots, s_{nI})$.
 x_{it} : echelon inventory at timeslot t of item i , which is defined as

$$x_{it} = x_{i0} + \sum_{t'=1}^t \sum_{k \in K(i)} p_i^k \delta_{it'}^k \bar{I}(s_{it'}^k) - \sum_{t'=0}^t r_{i \cdot t'} \quad (1)$$

where, if $s_{it}^k=0$ then $\bar{I}(s_{it}^k)=1$, else then 0. In case item i is a final product, echelon inventory is equivalent to actual inventory, where, $x_i = (x_{i1}, \dots, x_{it}, \dots, x_{inT})$, $x = (x_1, \dots, x_i, \dots, x_{nI})$

Lagrange multipliers

- μ_{it}^k : Lagrange multiplier for the interference constraint (6) of machine k at timeslot t
 μ_{it} : Lagrange multiplier for the shortage of item i at timeslot t

2.2 ConstraintsRelation among requirements

Requirements of item j is exploded into item i with lead time τ_{ij} , where item i is work-in-process item. Therefore, the following relation holds.

$$\rho_{ij} \begin{cases} \geq 1, & i \in A, \quad j \in S(i) \\ = 0, & \text{otherwise} \end{cases} \quad (2)$$

$$r_{ijt} = \rho_{ij} r_{j \cdot (t - \tau_{ij})}$$

$$r_{i \cdot t} = \sum_{j \in S(i)} r_{ijt} + r_{i0t} \quad (3)$$

Transition equation of remaining time of setup on machine k

$$s_{it}^k = s_{i0}^k (1 - \delta_{i(t-1)}^k) \delta_{it}^k + (s_{i(t-1)}^k - 1) + \delta_{i(t-1)}^k \delta_{it}^k \quad (4)$$

where, $(y)_+ = \max(0, y)$.

Echelon inventory transition equation

$$x_{it} = x_{i(t-1)} + \sum_{k \in K(i)} p_i^k \delta_{it}^k \bar{I}(s_{it}^k) - r_{i \cdot t} \quad (5)$$

Machine interference constraint

$$\sum_{i \in M(k)} \delta_{it}^k - 1 \leq 0 \quad (6)$$

Constraint to avoid the shortage of work-in-process item

$$\sum_{j \in S(i)} \rho_{ij} (x_{jt} - x_{j0}) - \sum_{t'=0}^t \left(r_{i \cdot t'} - r_{i0t'} - \sum_{j \in S(i)} \rho_{ij} r_{j \cdot t'} \right) - x_{it} \leq 0 \quad (7)$$

where in case of lead time $\tau_{ij}=0$, the second term of left hand side in (7) takes the value zero and disappears because of Eqs.(2) and (3).

2.3 Costs

Related costs are setup cost c_i^k and echelon inventory holding cost $h_i x_{it}$, and then cost function is as follows.

$$f_i(\delta_i, s_i, x_i) = \sum_{t \in T} \left(\sum_{k \in K(i)} c_i^k (1 - \delta_{i(t-1)}^k) \delta_{it}^k + h_i x_{it} \right) \quad (8)$$

2.4 Problem

After all, the problem is formulated as follows.

Problem P0

$$\min_{\delta, s, x} \sum_{i \in I} f_i(\delta_i, s_i, x_i) \quad (9)$$

subject to (4), (5), (6), (7)

3. Narrow Sense Lagrangian Decomposition Coordination Method**3.1 Lagrangian**

Problem P0 is one of constrained optimization problems and hence, as well known, we will transform it into an unconstrained optimization problem by Lagrangian relaxation. Denoting the Lagrangian as

$$\begin{aligned} L(\delta, s, x; \mu) = & \sum_{i \in I} f_i(\delta_i, s_i, x_i) \\ & + \sum_{k \in K(i)} \sum_{t \in T} \mu_{it}^k \left(\sum_{i \in M(k)} \delta_{it}^k - 1 \right) \\ & + \sum_{i \in P(i)} \sum_{t \in T} \mu_{it} \left(\sum_{j \in S(i)} \rho_{ij} (x_{jt} - x_{j0}) \right. \\ & \left. - \sum_{t'=0}^t \left(r_{i \cdot t'} - r_{i0t'} - \sum_{j \in S(i)} \rho_{ij} r_{j \cdot t'} \right) - x_{it} \right) \end{aligned} \quad (10)$$

the problem to be solved is given as

Problem P1

$$\max_{\mu} \min_{\delta, s, x} L(\delta, s, x; \mu) \quad (11)$$

subject to (4), (5)

3.2 Item-based decomposition

Note that for given μ , the Lagrangian is separable. Namely, putting $L_i(\delta_i, s_i, x_i; \mu)$ as

$$L_i(\delta_i, s_i, x_i; \mu) = f_i(\delta_i, s_i, x_i) + \sum_{t \in T} \sum_{k \in K(i)} \mu_{it}^k \delta_{it}^k - \sum_{t \in T} \left(\mu_{it} a(i) - \sum_{j \in P(i)} \mu_{jt} \rho_{ji} \right) x_{it} \quad (12)$$

where

$$a(i) = \begin{cases} 1, & \text{item } i \text{ has predecessor} \\ 0, & \text{otherwise,} \end{cases}$$

we have

$$L(\delta, s, x; \mu) = \sum_{i \in I} L_i(\delta_i, s_i, x_i; \mu) - \sum_{t \in T} \sum_{k \in K} \mu_{it}^k - \sum_{t \in T} \sum_{i \in A} \mu_{it} \left(\sum_{j \in S(i)} \rho_{ij} x_{jt} \right) + \sum_{t'=1}^t \left(r_{i \cdot t'} - r_{i0t'} - \sum_{j \in S(i)} \rho_{ij} r_{j \cdot t'} \right) \quad (13)$$

Therefore, for given μ , we can decompose the problem of minimizing the Lagrangian into one dimensional sub-problems.

Problem P2

$$\min_{\delta_i} L_i(\delta_i, s_i, x_i; \mu) \quad (14)$$

subject to (4), (5)

3.3 Coordination of Lagrange multipliers

We present the following non-decreasing rules of updating Lagrange multipliers.

$$\mu_{it}^{h(v+1)} = \mu_{it}^{hv} + \max \left\{ 0, \alpha_{\mu_{it}^h} \left(\sum_{i \in M(h)} \delta_{it}^{hv} - 1 \right) \right\} \quad (15)$$

$$\mu_{it}^{v+1} = \mu_{it}^v + \max \left\{ 0, \alpha_{\mu_{it}} \left(\sum_{j \in S(i)} \rho_{ij} (x_{jt}^v - x_{jt}) \right) - \sum_{t'=1}^t \left(r_{i \cdot t'} - r_{i0t'} - \sum_{j \in S(i)} \rho_{ij} r_{j \cdot t'} \right) - x_{it}^v \right\} \quad (16)$$

where, $\alpha_{\mu_{it}^h}$ and $\alpha_{\mu_{it}}$ is step length in Refs. (15) and (16) respectively, and its tentative value is given by

$$\alpha_{\mu_{it}^h} = \max_k \{ p_i^h \cdot h_i \}$$

$$\alpha_{\mu_{it}} = \max \{ p_i^h \cdot h_i / n_T \}$$

4. Sub-Optimization by Dynamic Programming

In this section, we present a method of reformulating suboptimization problem P2 into dynamic programming (DP) model and solving it.

4.1 Admissible set and transition

The state of machine k can be stated with binary decision variables δ_{it}^k and the remaining time of setup s_{it}^k . Table 1 shows the state transition of machine k from timeslot $t-1$ to t . Symbol $\bigcirc$ and blank respectively means that the transition is admissible and not.

4.2 Functional equation of optimal cost function

According to the well known DP approach, for given μ the functional equation of the cost function of problem P2 is defined as follows. For $t=0$ (boundary

Table 1 Admissible set and transition

$(\delta_{it-1}^k, s_{it-1}^k)$	$(\delta_{it}^k, s_{it}^k)$			
	(0,0)	(1,0)	$(1, 1 < s_{it}^k < s_{itmax}^k)$	$(1, s_{itmax}^k)$
(0,0)	$\bigcirc$			$\bigcirc$
(1,0)	$\bigcirc$	$\bigcirc$		
(1,1)		$\bigcirc$		
$(1, 1 < s_{it}^k < s_{itmax}^k)$			$\bigcirc$	

condition),

$$V_{i0}(\delta_{i0}, s_{i0}, x_{i0}; \mu) = \begin{cases} 0, & \text{when variables take} \\ & \text{given initial value} \\ +\infty, & \text{otherwise} \end{cases} \quad (17)$$

and for $t \geq 1$,

$$V_{it}(\delta_{it}, s_{it}, x_{it}; \mu) = \min_{\delta_{it(t-1)}, s_{it(t-1)}} \left\{ \sum_{t \in T} \left(\sum_{k \in K(i)} c_i^k (1 - \delta_{it(t-1)}^k) \delta_{it}^k + h_i x_{it} \right) + \sum_{t \in T} \sum_{k \in K(i)} \mu_{it}^k \delta_{it}^k - \sum_{t \in T} \left(\mu_{it} a(i) - \sum_{j \in P(i)} \mu_{jt} \rho_{ji} \right) x_{it} + V_{it-1}(\delta_{it-1}, s_{it-1}, x_{it-1}; \mu) \right\} \quad (18)$$

5. Solution Procedure

After all, the solution procedure of the presented method is summarized as follows.

Step 1 put all Lagrange multipliers to 0.

Step 2 for the given multipliers, solve each sub-problem by DP. Count the perturbation values in (6) and (7). If all constraints as to (6) and (7) are satisfied, then stop and else go to step 3.

Step 3 update Lagrange multipliers by Eqs. (15) and (16), and then go to step 2.

6. Numerical Model

In this section, we verify the presented method by use of a numerical model. We show the problem data in 6.1 and illustrate a series of results in 6.2. In 6.3, we present the second means for a lower bound.

6.1 Data

The problem data are given below and in Tables 2-4

Number of items: 15 items

Number of machines: 6 machines

Planning horizon: 1 weeks (294 timeslots)

Initial state: all machines are idle

Initial Lagrange multipliers: $\mu_{it}^k = 0, \forall k \in K, t \in T, \mu_{it} = 0, \forall i \in I, t \in T$

Step length: $\alpha_{\mu_k} = 0.038, \alpha_{\mu_k} = 0.040, \alpha_{\mu_k} = 0.045,$

$\alpha_{\mu_k} = 0.084, \alpha_{\mu_k} = 0.012, \alpha_{\mu_k} = 0.084, \alpha_{\mu_k} = 0.000255$

6.2 Result

The feasible solution was obtained at 678th iteration (CPU time: about 25 minutes by 1.5 GHz, Pentium 4), namely, any one of machine interference and shortage of work-in-process item dissolved from the whole timeslots, and hence, following our algorithm, the computation was stopped. At that iteration the objective function value and the Lagrangian value was, respectively, 2 233.56 and -4 548.13. The transition of each function value is shown in Fig. 2. The Lagrangian value decreases drastically, and hence a method of exploiting the largest Lagrangian value as

Table 2 Bill of materials ρ_{ij}

	j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
i	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
	2	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0
	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	4	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0
	5	0	0	0	0	0	0	0	0	0	2	0	1	0	0	0
	6	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0
	7	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
	8	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0
	9	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	11	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3 Processing rate p_i^k , setup time $s_{i\max}^k$, setup cost c_i^k

	k	1	2	3	4	5	6
i	1	25	2	3			
	2	19	1	5			
	3		6	2	5		
	4		10	1	7	4	1
	5			9	2	6	14
	6			7	2	5	11
	7						12
	8						10
	9						11
	10						7
	11						8
	12						8
	13						
	14						
	15						

Table 4 Initial inventory x_{i0} , holding cost h_i and order requirements from market r_{i0t}

	x_{i0}	h_i	r_{i0t}						
			t	42	84	126	168	210	252
i	1	47	0.0010	0	0	0	0	0	0
	2	56	0.0020	0	0	0	0	0	0
	3	16	0.0030	73	60	75	79	86	73
	4	20	0.0040	0	0	0	0	0	0
	5	18	0.0050	0	0	0	0	0	0
	6	15	0.0060	0	0	0	0	0	0
	7	13	0.0070	0	0	0	0	0	0
	8	25	0.0080	0	0	0	0	0	0
	9	26	0.0110	0	0	0	0	0	0
	10	30	0.0120	46	31	55	46	26	39
	11	23	0.0140	35	45	45	32	33	37
	12	11	0.0140	23	24	0	42	25	47
	13	13	0.0110	36	35	29	37	36	0
	14	16	0.0160	0	28	37	30	29	33
	15	17	0.0180	21	35	25	26	40	30

a lower bound of the objective function is also of no use. This comes from the characteristics proper to the problem. Moreover, it makes exact problem solution very hard. We clarify those characteristics here.

First of all, note the fact that setup cost always attracts lot production. In addition, note that in any real lot production situation machine idling time and also work-in-process inventory always exist. It is because for any machine the rate of operation must be

Fig. 2 Transition of objective function value and Lagrangian value

less than unity and without work-in-process inventory a system cannot spend time on setup. In other words, there exist always the machine interference constraint in (6) taking the value “-1” and the constraint to avoid the shortage of work-in-process item in (7) taking a negative value. This in turn means that any one of the second and third terms in the Lagrangian in (10) takes a negative value with positive Lagrange multipliers. To make matters worse, the solution is quite sensitive to the penalty cost (the value of the Lagrange multiplier) μ_i^k and μ_{it} and removing just by a very small increment the penalty cost that has been incurred once would almost always return the last solution to the previous one. Because of the third characteristic, we proposed non-decreasing rules in (15) and (16) to update the Lagrange multipliers, which differs from the usual one such as Refs. (15) - (17). After all, those characteristics lead to drastic decrease of the Lagrangian value.

Clearly, the usage of those non-decreasing rules is a compromise to obtain any feasible solution easily and yet as fast as possible. As a result, it is certain from Lagrangian dual theory that it may cause to increase the Lagrange multiplier values too much extent and hence the objective function value also. So, even if not perfect, we need to guarantee a near optimality of the presented method by any means. It is the subject in section 6.3 and, according to the means, a lower bound of this numerical example was given as 1 653.51, and we conclude that we are satisfied with this result for the time being. A sophisticated solution method would be a theme of further research.

Figures 3 and 4 show respectively the transition of amount of constraint violation as to machine interferences in (6) and as to work-in-process inventory in (7). Though it is not clear of proof on convergence to any feasible solution and its speed, all of

Fig. 3 Transition of amount of machine interferences

Fig. 4 Transition of amount of shortage of work-in-process inventory

Fig. 5 Lagrange multipliers value μ_t^k

several numerical examples have converged at any rate. Figure 5 shows the final value of the Lagrange multiplier μ_t^k . This value corresponds to the machine cost of machine k at timeslot t , which reflects on the level of machine load of that machine at that timeslot

Fig. 6 Inventory transition and Gantt chart

in that situation. Therefore, its value is a dynamic index for bottleneck in this system. The final result is shown in Fig. 6.

6.3 A new lower bound by deleting the constraint (7)

It is well known that a lower bound of a problem obtained by deleting (relaxing) some constraints is less than or equal to the original lower bound of the original problem. As existence of work-in-process inventory makes it difficult to increase the Lagrangian value, for the time being by deleting the constraint (7), we solve the relaxed Lagrangian to obtain a new lower bound.

First, for each machine k , introduce the dummy product $n_l + k$, and put $h_{nl+k} = 0$ and $c_{n_l+k} = 0$. Redefine every related index sets. Mathematically, for any k and any t Eq. (6) holds if and only if

$$\sum_{i \in M(k)} \delta_{it}^{kv} - 1 \leq 0 \quad (19)$$

$$n_T - \sum_{i \in M(k)} \sum_{t \in T} \delta_{it}^k \leq 0 \quad (20)$$

Computationally, however, this new formulation with dummy items facilitate to eject “-1” from the left hand side of (19).

Therefore, we put the relaxed Lagrangian as

$$\begin{aligned} \tilde{L}(\delta, s, x; \mu) = & \sum_{i \in I} f_i(\delta_i, s_i, x_i) + \sum_{k \in K(i)} \sum_{t \in T} \mu_{\cdot t}^k \\ & \times \left(\sum_{i \in M(k)} \delta_{it}^k - 1 \right) + \sum_{k \in K(i)} \mu_{\cdot}^k \left(n_T - \sum_{i \in M(k)} \sum_{t \in T} \delta_{it}^k \right) \end{aligned} \quad (21)$$

where, $\mu_{\cdot}^k$ denotes the Lagrange multiplier for Eq. (20).

The rule of updating the Lagrange multipliers is as follows

$$\mu_{\cdot t}^{k(v+1)} = \max \left\{ 0, \mu_{\cdot t}^{kv} + \alpha \mu_{\cdot}^k \left(\sum_{i \in M(k)} \delta_{it}^{kv} - 1 \right) \right\} \quad (22)$$

$$\mu_{\cdot}^{k(v+1)} = \max \left\{ 0, \mu_{\cdot}^{kv} + \alpha \mu_{\cdot}^k \left(n_T - \sum_{i \in M(k)} \sum_{t \in T} \delta_{it}^{kv} \right) \right\} \quad (23)$$

7. Conclusion

In this paper, we presented a near-optimal solution method that we called the narrow sense Lagrangian decomposition coordination method for a multi-item multi-process dynamic lot size scheduling problem with general product structure and sequence independent setup time. This method enabled us to solve heterogeneous decision features involved in this problem such as lot sizing, lot sequencing, dispatching simultaneously without specifying them one by one. Improvement of a lower bound and acceleration of computation are under study.

References

- (1) Afentakis, P. and Gavish, B., Optimal Lot-Sizing Algorithms for Complex Product Structures, *Oper. Res.*, Vol. 34 (1986), pp. 237-249.
- (2) Billington, P.J., McClain, J.O. and Thomas, L.J., Heuristics for Multilevel Lot-Sizing with a Bottleneck, *Manage. Sci.*, Vol. 32, No. 8 (1986), pp. 989-1006.
- (3) Tempelmeir, H. and Destroff, M., A Lagrangian-based Heuristic for Dynamic Multi-item Multi-level Constrained Lotsizing with Setup Times, *Manage. Sci.*, Vol. 42, No. 5 (1996), pp. 738-757.
- (4) Kimms, A., Multi Level Lot Sizing and Scheduling—Methods for Capacitated, Dynamic, and Deterministic Models, *Physica, Heidelberg*, (1997).
- (5) Drexl, A. and Kimms, A., Lot Sizing and Scheduling—Survey and Extensions, *Eur. J. Oper. Res.*, Vol. 99 (1997), pp. 221-235.
- (6) Kimms, A., A Genetic Algorithm for Multi-level Multimachine Lot Sizing and Scheduling, *Compu. Oper. Res.*, Vol. 26 (1999), pp. 829-848.
- (7) Meyr, H., Simultaneous Lotsizing and Scheduling by Combining Local Search with Dual Reoptimization, *Eur. J. Oper. Res.*, Vol. 120 (2000), pp. 311-326.
- (8) Meyr, H., Simultaneous Lotsizing and Scheduling on Parallel Machines, *Eur. J. Oper. Res.*, Vol. 139 (2002), pp. 277-292.
- (9) Fleischmann, B., The Discrete Lot-Sizing and Scheduling Problem, *Eur. J. Oper. Res.*, Vol. 44 (1990), pp. 337-348.
- (10) Drexl, A. and Hasse, K., Proportional Lot Sizing and Scheduling, *Int. J. Prod. Econ.*, Vol. 40 (1995), pp. 73-87.
- (11) Fleischmann, B. and Meyr, H., The General Lot-sizing and Scheduling Problem, *OR. Spektrum.*, Vol. 19, No. 1 (1997), pp. 11-21.
- (12) Clark, A.J. and Scarf, H., Optimal Policies for a Multi-Echelon Inventory Problem, *Manage. Sci.*, Vol. 6 (1960), pp. 457-490.
- (13) Lasdon, L.S., *Optimization Theory for Large Systems*, Macmillan, (1970).
- (14) Vemuganti, R.R., Arsham, H. and Shao, Jr., S.P., An Implementation of Lagrangian Decomposition in Solving A Multi-item Production Scheduling Problem with Changeover Cost and Restrictions, *Mathl. Comput. Modelling*, Vol. 12, No. 12 (1989), pp. 1437-1446.
- (15) Held, M., Wolfe, P. and Crowder, H.P., Validation of Subgradient Optimization, *Math. Prog.*, Vol. 6 (1974), pp. 62-88.
- (16) Georion, A.M., Lagrangian Relaxation for Integer Programming, *Math. Prog. Stud.*, No. 2 (1974), pp. 82-114.
- (17) Fisher, M.L., The Lagrangian Relaxation Method for Solving Integer Programming Problems, *Manage. Sci.*, Vol. 27, No. 1 (1981), pp. 1-18.
- (18) Muramatsu, K., Simultaneous Optimization for Lot Sizing and Lot Sequencing: Multi-item Lot Size Scheduling with Setup Time, *Journal of Japan Industrial Management Association*, (in Japanese), Vol. 52, No. 5 (2001), pp. 213-221.