

Tightened Formulation and Resolution of Energy-Efficient Job-Shop Scheduling

Bing Yan, *Member, IEEE*, Mikhail A. Bragin, *Member, IEEE*, Peter B. Luh, *Life Fellow, IEEE*

Abstract— Job shops are an important production environment for low-volume high-variety manufacturing. When there are urgent orders, the speeds of certain machines can be adjusted with a high energy and wear and tear cost. Scheduling in such an environment is to achieve on-time deliveries and low energy costs. The problem is, however, complicated because part processing time depends on machine speeds, and machines need to be modeled individually to capture energy costs. This paper is to obtain near-optimal solutions efficiently. The problem is formulated as a Mixed-Integer Linear Programming (MILP) form to make effective use of available MILP methods. This is done by modeling machines in groups for simplicity while approximating energy costs, and by linking part processing status and machine speed variables. Nevertheless, the resulting problem is still complicated. The formulation is therefore transformed by extending our previous tightening approach for machines with constant speeds. The idea is that if constraints can be transformed to directly delineate the convex hull, then the problem can be solved by linear programming methods. To solve the problem efficiently, our advanced decomposition and coordination method is used. Numerical results show that near-optimal solutions are obtained, demonstrating significant benefits of our approach on on-time deliveries and energy costs.

Index terms—Job-shop scheduling, energy costs, mixed-integer linear programming, formulation tightening

I. INTRODUCTION

Job shops are an important production environment for low-volume high-variety manufacturing. To meet on-time deliveries, scheduling of parts is critical. In a job shop, machines are usually categorized into different types based on their functions, i.e., which parts/operations a machine can process. When there are urgent orders which have tight due dates, the speeds of certain machines can be adjusted for different parts/operations in a discrete speed level. However, high speeds come with high energy and wear and tear costs. Under the environment of machine speed scaling, the scheduling is to achieve on-time deliveries and low energy costs. The problem is to minimize the total weighted tardiness penalty and energy cost by assigning parts to machines and determining the corresponding machine process speeds while satisfying part processing time, operation precedence, and machine capacity constraints.

The scheduling problem is, however, complicated because the part processing time and machine energy costs depend on machine speeds. In addition, when a machine is idle, it usually consumes much less energy as compared to the processing mode. To capture energy costs under processing or idle modes, machines need to be modeled individually instead of groups.

This paper is to obtain near-optimal solutions efficiently for energy-efficient job-shop to meet product on-time deliveries and reduce energy costs. The problem is formulated as a Mixed-Integer Linear Programming (MILP) form to make effective use of available MILP methods in Section III. To maintain the group concept for complexity reduction, machines are modeled in groups for simplicity while energy costs are approximated for each machine group instead of each machine. Processing time constraints are established by linking part processing status and machine speed variables.

Nevertheless, the resulting problem is still complicated because part processing time depends on machine speeds. The formulation is therefore transformed by extending our previous tightening approach for machines with constant speeds in Section IV. The idea is that if constraints can be transformed to directly delineate the convex hull, then the problem can be solved by linear programming without combinatorial difficulties. In our previous work [2, 3], several processing time related tightened constraints are obtained. With machine and speed dependent processing time, those constraints are modified based on machine speed variables and used as lower or upper bounds, e.g., the earliest possible beginning time based on the shortest processing time.

For the problem under consideration, most constraints are associated with individual parts, and all parts are coupled by machine capacity constraints. By exploiting the exponential reduction of complexity, our advanced decomposition and coordination method with accelerated convergence [4] is used to improve computational efficiency in Section V.

The above method is implemented by using CPLEX, and three examples are presented in Section VI. The first small one is to illustrate the impacts of energy costs on job-shop scheduling. The second is to demonstrate the effectiveness of formulation tightening. The third is to show computational efficiency of the decomposition and coordination method.

Bing Yan is with the Department of Electrical and Microelectronic Engineering, Rochester Institute of Technology, Rochester, NY 14623, USA (e-mail: bxyeee@rit.edu).

Peter B. Luh, and Mikhail A. Bragin are with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT 06269-4157, USA (e-mail: peter.luh@uconn.edu, and mikhail.bragin@uconn.edu).

This work is supported in part by the National Science Foundation (NSF) under the grant ECCS-1810108 and U.S. Department of Energy (DoE)'s Office of Energy Efficiency and Renewable Energy under the Advanced Manufacturing Office Award Number DE-EE0007613. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of NSF or DoE.

II. LITERATURE REVIEW

Existing job-shop scheduling formulations and tightened constraints are reviewed in Subsection A. Solution methodologies are reviewed in Subsection B.

A. Problem formulations and tightened constraints

In [1], an integer programming model was developed for flexible job-shop scheduling under the framework of machine speed scaling to minimize the sum of energy-consumption cost and completion-time cost, where a set of binary variables was used to indicate machine speeds. A similar model was established for multi-objective optimization in [5]. Without considering machine speed scaling, there are also studies focusing on machine on-off decisions to reduce energy costs. Based on the modeling idea of idle time, an integer linear programming (ILP) model with turning off/on strategy for flexible job-shop scheduling was developed in [6]. In [7-9], several ILP models were developed based on two modeling ideas, namely idle time and idle energy, for energy-conscious flexible job-shop scheduling problems. In the above, machines were modeled individually. When the number of machines increases, the problem is difficult to solve.

Obtaining a tight formulation is fundamentally difficult, and it is rarely discussed in the literature. For traditional job-shop scheduling in [10], tightened constraints were developed by identifying a ceiling for inventory shortage, and the longest working procedure sequence till part completion. For flow-shop scheduling, subtour elimination constraints and lower /upper bound mixed-integer inequalities were developed by analyzing formulation structures in [11]. For both studies, testing results based on randomly generated data demonstrate computational efficiency of these tightened constraints.

In our previous work on job-shop scheduling with constant machine speeds and without considering energy costs [2, 3], a few processing time-related constraints were obtained for single parts based on novel “constraint-and-vertex conversion,” “vertex elimination” and “parameterization.” Results show that our formulation tightening is effective in terms of computational efficiency and solution quality.

B. Solution methodologies

While metaheuristic approaches [1, 6, 12] are attractive for job-shop scheduling problems owing to their low computational requirements, within these methods, solution quality cannot be measured and there is typically no systemic way to improve the solution.

The branch-and-cut (B&C) method for job-shop scheduling with ILP formulations has also used [10, 11]. Within the method, after relaxing integrality requirements, the problem is solved by using linear programming methods. If the resulting solution is integral, then the solution is optimal to the original problem. However, this can only be ensured for “totally unimodular” formulations. Therefore, there is generally a need for B&C to cut off the regions outside of the convex hull without cutting off feasible solutions by using cuts. If an integer optimal solution is not obtained as a result of cutting, the method relies on branching operations and heuristics. Moreover, the method does not exploit the “local” problem features such as process time requirements; all

constraints within a problem are handled as “globally.” These constraints thus affect the entire convex hull, influence the solution process, and lead to slow convergence.

To solve job-shop scheduling problems, Lagrangian relaxation has been traditionally used to exploit exponential reduction of complexity after decomposition, to obtain near-optimal solutions with quantifiable quality in a computationally efficient way [13-15]. In reference [13], separability was exploited through decomposition, and then subproblems were solved by dynamic programming and coordinated by Lagrangian multipliers based on violation levels of relaxed constraints. However, convergence of Lagrangian relaxation is very slow because of such major difficulties as high computational requirements to solve all subproblems, and significant zigzagging of multipliers. Within our recent surrogate Lagrangian relaxation (SLR), these difficulties have been overcome. Within the method, the solution of one or few subproblems is sufficient to update multipliers [16]. Moreover, convergence has been proved without requiring the optimal dual value.

To accelerate convergence of SLR, “absolute-value” penalty functions were used [17]. Without considering energy costs, the job-shop scheduling problem was efficiently solved by the resulting Surrogate “Absolute-Value” Lagrangian relaxation (SAVLR) method [2].

III. PROBLEM FORMULATION

Based on our previous work [2, 3], an MILP formulation for energy-efficient job-shop scheduling is established.

A. Machine capacity constraints

Consider a job shop with I parts to be processed, indexed by i . Part i requires J_i operations indexed by j . For easy presentation, (i, j) is used to denote operation j of part i . In the shop, there are M types of machines (machine groups) indexed by m . Each type of machines can adjust its speeds for various parts/operations in a discrete speed level, and part processing time is machine and speed dependent. The scheduling horizon is discretized into T time slots indexed by t , and it is assumed that T is long enough to process all parts.

To capture whether (i, j) is active on machine type m (since the processing time is machine dependent as mentioned earlier) at time t or not, a set of binary decision variables δ_{ijmt} with four indices is introduced as follows:

$$\delta_{ijmt} = \begin{cases} 1, & \text{operation } j \text{ of part } i \text{ is active on machine } m \text{ at time } t; \\ 0, & \text{otherwise.} \end{cases}$$

For machine type m , the total number of active parts cannot exceed its capacity M_m at any time slot t , i.e.,

$$\sum_{(i,j) \in O_m} \delta_{ijmt} \leq M_m, \forall m, \forall t. \quad (1)$$

In the above, O_m denotes the set of (i, j) that can be processed by machine type m . This system-wide machine capacity constraint couples different parts.

B. Machine speed constraints

For machine type m , the total number of speed levels is S_m with the speed level indexed by s . The higher the speed, the shorter the processing time of the part/operation assigned to the machine group. To capture at which speed that (i, j) is

being processed on machine type m , a set of binary decision variables x_{ijms} with four indices is introduced as follows:

$$x_{ijms} = \begin{cases} 1, & \text{operation } j \text{ of part } i \text{ is assigned to machine type } m \\ & \text{to process at a speed of } s; \\ 0, & \text{otherwise.} \end{cases}$$

To capture whether (i, j) is assigned to machine type m , a set of binary variables y_{ijm} is defined as follows:

$$y_{ijm} = \sum_{s=1}^{S_m} x_{ijms}, \forall i, \forall j, \forall m. \quad (2)$$

Not that y_{ijm} is not a decision, and it depends on x_{ijms} . If y_{ijm} is 1, then (i, j) is assigned to machine type m ; and 0, otherwise.

For (i, j) , it can only be processed on one machine at one speed level, i.e.,

$$\sum_{m \in N_{ij}} y_{ijm} = 1, \forall i, \forall j, \quad (3)$$

where N_{ij} is the set of machine types that can process (i, j) .

C. Processing time requirements

Let b_{ij} and c_{ij} denote the beginning and completion time of (i, j) . They are integer decision variables and are linked via processing time requirements. When (i, j) is assigned to machine type m , the reference processing time is p_{ijm} . If speed s is selected, the processing time is assumed as p_{ijms} , where $p_{ijms} = p_{ijm}/s$. Also the speed cannot be changed during the processing of (i, j) . Since the processing is assumed to be “non-preemptive,” a contiguous time block of length p_{ijms} is needed if (i, j) is assigned to machine type m at speed s , i.e.,

$$c_{ij} = b_{ij} + \sum_{m \in M_{ij}} \sum_{s=1}^{S_m} x_{ijms} p_{ijms} - 1, \forall i, \forall j. \quad (4)$$

Within $[b_{ij}, c_{ij}]$, δ_{ijmt} must be 1 if (i, j) is assigned to machine type m , and 0 otherwise, i.e.,

$$\delta_{ijmt} = \begin{cases} 1, & \text{if } b_{ij} \leq t \leq c_{ij}, \text{ and } y_{ijm} = 1; \\ 0, & \text{otherwise.} \end{cases} \quad (5)$$

This logical constraint can be linearized as follows:

$$t \leq c_{ij} + T(1 - \sum_{m \in N_{ij}} \delta_{ijmt}), \forall i, \forall j, \forall t; \quad (6)$$

$$t \geq b_{ij} - T(1 - \sum_{m \in N_{ij}} \delta_{ijmt}), \forall i, \forall j, \forall t; \quad (7)$$

$$\sum_t \delta_{ijmt} = \sum_{s=1}^{S_m} x_{ijms} p_{ijms}, \forall i, \forall j, m \in M_{ij}. \quad (8)$$

The above guarantees $\delta_{ijmt} = 1$ iff $b_{ij} \leq t \leq c_{ij}$ with $y_{ijm} = 1$; and $\delta_{ijmt} = 0$ when $t < b_{ij}$ or $t > c_{ij}$ with $y_{ijm} = 1$, or any t when $y_{ijm} = 0$. Linear constraints (6-8) are equal to logical (5).

D. Operation precedence constraints

For a particular part, it is assumed the operation sequence is fixed. Because of operation precedence requirements, operation $(i, j+1)$ cannot start before (i, j) is completed, i.e.,

$$b_{i,j+1} \geq c_{ij} + 1, \forall i, \forall j. \quad (9)$$

The first operation cannot start until part i is arrived at a_{i1} , i.e.,

$$b_{i1} \geq a_{i1}, \forall i. \quad (10)$$

E. Energy cost

With speed s , the energy cost of a machine in type m is defined as E_{ms} (\$/day). For (i, j) on machine m , if $s > s'$, then

$E_{ms} \times p_{ijms} > E_{ms'} \times p_{ijms'}$. In other words, a higher machine speed can reduce the processing time, but increase the energy cost. In addition, it is assumed that each machine is completely turned off after all parts that are assigned to it are completed. During the idle time, machines in type m are on a stand-by mode with an energy cost of E_m^{StBy} (\$/day). To capture the idle time, machine workload and completion time of the last part that is assigned to individual machines is needed. However, since the machine group concept is used here instead of individual machines for complexity reduction, it is difficult to get the workload and the completion time of the last part on each machine. For simplicity, the average workload of the machine group, and the completion time of the last part assigned to the group is considered for every machine within the same group as approximation.

Average machine workload in type m is obtained as,

$$W_m = \sum_{\forall (i,j) \in O_m} \sum_t \delta_{ijmt} / M_m, \forall m. \quad (11)$$

The completion time of the last part that is assigned to machine type m is derived as follows,

$$c_m \geq c_{ij} - T(1 - y_{ijm}), \forall m, \forall (i, j) \in O_m \quad (12)$$

F. Objective function

The objective function is to minimize the total tardiness penalty and energy cost as presented below:

$$\begin{aligned} & \sum_i \left(\sum_j \sum_{s=1}^{S_m} E_{ms} p_{ijms} x_{ijms} + E_m^{StBy} M_m (c_m - W_m) \right) \\ & + \sum_i \omega_i (\max(c_{ij} - d_i, 0)). \end{aligned} \quad (13)$$

In the above, ω_i is a tardiness penalty weight for part i (\$/day) and d_i denotes its due date. The tardiness function is converted to a piecewise-linear function and linearized by special ordered set techniques [18]. After this conversion, the above objective function is purely linear. For compactness of notation, function $\max$ is kept here and later in the paper.

The energy-efficient job-shop scheduling problem with Eq. (1-4) and (6)-(13) established above is an MILP problem.

IV. FORMULATION TIGHTENING

The general idea of our formulation tightening approach is introduced in Subsection A. Tightened constraints obtained for parts with constant processing time in our previous work [2, 3] are extended for parts with machine and speed dependent processing time in Subsection B.

A. Formulation tightening

With system-level machine capacity constraints relaxed, the single-part formulation is tightened by using our systematic approach [2, 3]. Given part parameters (e.g., processing time p) in numerical values, tightened constraints are established through four steps.

In the first step, integrality requirements on integer variables are relaxed. For the linear relaxation problem, vertices of the convex hull are generated from constraints via algebraic manipulation of part parameters with well established algorithms [19]. The second step is to obtain the vertices of the convex hull of the original problem by simply eliminating fractional vertices with proved tightness in [3]. These vertices are then converted back to tight constraints

with numerical coefficients in the third step as a reverse process of the first step. For these constraints, coefficients depend on part parameters, as well as the numbers of tight constraints and variables. For general use purposes, the numerical coefficients of tight constraints are characterized by analyzing constraint structures and relationships between coefficients and part parameters in the last step.

B. Tightened constraints

In [2, 3], several sets of processing time-related tightened constraints were obtained for individual parts by using the systematic approach described in Subsection A. They are extended to the problem under consideration as follows.

a) Part status related requirements

The small scheduling problem used in [2, 3] has one part and one operation with $p = 3$ and $T = 8$ (processing time is not machine or speed dependent). Decision variables include part status δ_i , beginning time b , and completion time c . By using the systematic tightening approach, a set of processing time-related tightened constraints are obtained as follows,

$$\delta_1 + \delta_4 + \delta_7 = 1, \delta_2 + \delta_5 + \delta_8 = 1, \delta_3 + \delta_6 = 1. \quad (14a-c)$$

Required by processing time constraints, three consecutive δ must be 1 among time slots 1 - 8. Thus one δ from the 1st, 4th, and 7th time slots must be 1 as shown in Eq. (14a). Similarly, one δ from time slots 2, 5 and 7 must be 1 as shown in Eq. (14b), and one δ from time slots 3 and 6 must be 1 as shown in Eq. (14c). This set of tightened constraints can be generalized for all operations with different processing time as follows,

$$\sum_{\tau=\lfloor K/p \rfloor p_{m\tau} + t \leq K}^{\tau=\lfloor K/p \rfloor p_{m\tau} + t \leq K} \delta_{t+p\tau} = 1, t \in [1, p]. \quad (15)$$

For energy-efficient job-shop scheduling under consideration, processing time is machine/speed dependent. The above tightened constraint can be extended as follows,

$$\sum_{\tau=\lfloor K/p_{ms} \rfloor p_{m\tau} + t \leq K}^{\tau=\lfloor K/p_{ms} \rfloor p_{m\tau} + t \leq K} \delta_{m,t+p\tau} \leq x_{ms} + p_{ms}(1 - x_{ms}), t \in [1, p_{ms}], \forall m, \forall s. \quad (16a)$$

$$\sum_{\tau=\lfloor K/p_{ms} \rfloor p_{m\tau} + t \leq K}^{\tau=\lfloor K/p_{ms} \rfloor p_{m\tau} + t \leq K} \delta_{m,t+p\tau} \geq x_{ms}, t \in [1, p_{ms}], \forall m, \forall s. \quad (16b)$$

When $x_{ms} = 1$, machine type m is scheduled to process the part with speed s , so that the summation of the selected δ should be 1 as explained early. When $x_{ms} = 0$, machine type m is not scheduled to process the part with speed s (the part could be assigned to machine type m with other speed levels, or assigned to other machine types), so that the summation of the selected δ should be less than the processing time.

b) Beginning time related tightened constraints

By analyzing the same problem mentioned in Subsubsection a, one beginning time related tightened constraint is obtained as follows,

$$b = T - p + 1 - 2(\delta_1 + \delta_2) - (\delta_3 + \delta_4 + \delta_5) - 0(\delta_6 + \delta_7 + \delta_8). \quad (17)$$

Since the processing time is p and the part must be completed within the scheduling horizon, the largest beginning time is $T - p + 1$ with δ_6, δ_7 and δ_8 as 1 as implied in Eq. (17). When the starting of nonzero δ moves earlier, b gets smaller. The earlier the δ , the larger the impacts on b .

The above tightened constraint can be generalized for all operations with different processing time as follows,

$$b = T - p + 1 - \sum_{n=0}^{\lfloor T/p \rfloor} n \left(\sum_{\tau=0}^{\tau=p-\lfloor T/p \rfloor + n-1} \delta_{T-np-\tau} \right). \quad (18)$$

For energy-efficient job-shop scheduling, the above tightened constraint can be extended as follows,

$$b \geq T - \sum_m p_{m,s_m} x_{m,s_m} + 1 - \sum_m \sum_{n=0}^{\lfloor T/p_{m,s_m} \rfloor} n \left(\sum_{\tau=0}^{\tau=p_{m,s_m}-\lfloor T/p_{m,s_m} \rfloor + n-1} \delta_{T-np_{m,s_m}-\tau} \right). \quad (19)$$

Since when the largest speed is considered, the processing time is the smallest. With the shortest possible processing time, the beginning time obtained from the right hand side is the smallest.

The above tightened constraints (16) and (19) directly constrain variables δ and b , and tighten the formulation

V. SOLUTION METHODOLOGY

This section is to solve the problem by using our recent Surrogate Absolute-Value Lagrangian Relaxation [4].

A. Surrogate Lagrangian Relaxation (SLR) [17]

After system-wide machine capacity constraints (1) that couple individual parts are relaxed by using Lagrangian multipliers λ , the relaxed problem can be written as:

$$\min \left\{ \sum_i C_i^{ProE} + C^{StByE} + \sum_i C_i^{Penalty} + \sum_{i,m} \lambda_{im}^k \left(\sum_{\forall (i,j) \in O_m} \delta_{ijmt} - M_m \right) \right\}$$

$$\text{where } C_i^{ProE} \equiv \sum_m \sum_{j=1}^{J_i} \sum_{s=1}^{S_m} E_{ms} P_{ijms} x_{ijms}$$

$$C^{StByE} \equiv \sum_m E_m^{StBy} M_m (C_m - W_m)$$

$$C_i^{Penalty} \equiv \omega_i (\max(c_{J_i} - d_i, 0)). \quad (20)$$

In the above, for easy presentation, C_i^{ProE} , C^{StByE} and $C_i^{Penalty}$ are defined to represent the total energy cost to process part i , the total energy cost for machine stand-by, and tardiness penalty cost of part i , respectively.

The relaxed problem is additive and it thus can be decomposed into individual part subproblems as:

$$\min \left\{ C_i^{ProE} + C^{StByE} + C_i^{Penalty} + \sum_{i,m} \lambda_{im}^k \sum_{\forall j:(i,j) \in O_m} \delta_{ijmt} \right\} \quad s.t. (2-4, 6-12, 16, 19). \quad (21)$$

Subproblems are much easier to solve as compared to the original problem. Parts are coordinated by updating multipliers based on stepsizes and violations of machine capacity constraints as:

$$\lambda_{im}^{k+1} = [\lambda_{im}^k + s^k \tilde{g}(\delta^k)]^+, \quad (22)$$

where δ^k are latest available values of decision variables δ_{ijmt} ; stepsizes s^k are set as in [17]; and *surrogate subgradient directions* is used after one or few subproblems are solved at a time and defined as:

$$\tilde{g}(\delta^k) = \sum_{\forall (i,j) \in O_m} \delta_{ijmt}^k - M_m. \quad (23)$$

B. Surrogate Absolute-Value Lagrangian Relaxation (SAVLR) [4]

To speed up convergence of SLR, violations of relaxed constraints (1) are penalized through "absolute-value" penalty terms with positive penalty coefficients v^k . The resulting "absolute-value" subproblem then becomes:

$$\min_{c, \delta, z} \left\{ C_i^{ProE} + C_i^{StByE} + C_i^{Penalty} + \sum_{t,m} \lambda_{tm}^k \sum_{\forall j: (i,j) \in O_n} \delta_{jmt} + \frac{v^k}{2} \max \left(\sum_{\forall (i,j) \in O_n, i \neq j} \delta_{jmt} + \sum_{\forall j: (i,j) \in O_n} \delta_{jmt} - M_m, 0 \right) \right\}. \quad (24)$$

Subproblems are linearized exactly and subproblem i can be written in an MILP form after introducing integer decision variables q_{tm} as:

$$\min_{c, \delta, z} \left\{ C_i^{ProE} + C_i^{StByE} + C_i^{Penalty} + \sum_{t,m} \lambda_{tm}^k \sum_{\forall (i,j) \in O_n} \delta_{jmt} + \frac{v^k}{2} \sum_{t,m} q_{tm} \right\}, \quad (25)$$

$$s.t. (2-4, 6-12, 16, 19), \sum_{\forall (i,j) \in O_n, i \neq j} \delta_{jmt} + \sum_{\forall j: (i,j) \in O_n} \delta_{jmt} - M_m \leq q_{tm}. \quad (26)$$

Subproblems (25)-(26) are MILP problems and are solved very efficiently by using B&C since their complexity is significantly reduced after decomposition.

VI. NUMERICAL RESULTS

The above approach is implemented by using IBM ILOG CPLEX Optimization Studio V 12.8.0.0 [20] on a PC with 2.40GHz Intel Xeon(R) E-2286M CPU and 32G RAM. Three examples are presented.

Example 1: A small problem

This small example is to illustrate the impacts of energy costs on job-shop scheduling. Consider a job shop with two machines. The first cannot change speeds, and the energy cost is \$0.5/day when processing and \$0.12/day when stand-by. The second has two speeds with the energy cost of \$0.55/day and \$1.5/day, and the energy cost is \$0.14/day when stand-by. There are four parts: the first two have two operations, and the other two have one. Processing time, due dates and tardiness weights of the four parts are presented in Table I below. The time horizon is 10 days so that all parts can be processed.

TABLE I EX1: PART PROCESSING TIME, DUE DATES AND TARDINESS WEIGHTS

			Part1		Part2		Part3	Part4
			O1	O2	O1	O2	O1	O1
P_{ijms} (day)	Machine1	Speed1	/	4	2	/	4	5
	Machine2	Speed1	3	/	/	2	4	5
		Speed2	2	/	/	1	2	3
d_i (day)			5		4		10	8
ω_i (\$/day)			10		10		1	10

The problem is first solved with fixed machine speeds, i.e., machine 2 only has speed 1, and energy costs are not considered. The optimized schedule is shown in Fig. 1 below.

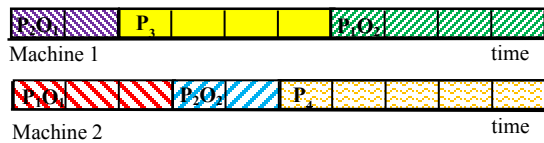


Figure 1. EX1: Schedule with fixed machine speeds

For part 1, operation 1 can only be processed on machine 2, and operation 2 can only be processed on machine 1. Part 2 is opposite. Parts 3 and 4 can be processed on machines 1 and 2 respectively. To guarantee that all the parts are processed within the scheduling horizon, part 3 is processed on machine 1 before operation 2 of part 1 since part 1 is still on machine 2 at day 3. The total tardiness cost is \$80.

Considering different speed levels of machine 2 and energy costs, the optimized schedule is shown in Fig. 2 below.



Figure 2. EX1: Schedule with different machine speeds and energy cost

As compared with Fig. 1, operation 1 of part 1 and part 4 are processed on machine 2 under the fast speed to reduce tardiness. Operation 2 of part 2 is processed under the low speed since the due day can still be met with a lower energy cost. The process sequence of part 3 and operation 2 of part 1 is swapped to reduce tardiness since operation 1 of part 1 is finished by the end of day 2. For this case, the total tardiness cost is \$10, while the energy cost is \$5.05. It can be seen that flexible machine speeds can improve on-time deliveries. When energy costs are not considered in the objective, the total tardiness cost is still \$10, while the energy cost is \$7. This demonstrates that under the same delivery requirements, energy costs can be reduced by changing machine speeds.

If the due date of part 3 is 7 instead of 10, the optimized schedule without considering energy costs is shown in Fig. 3.

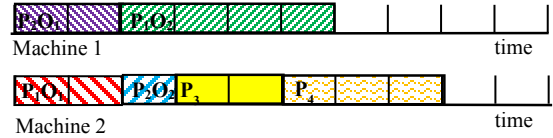


Figure 3. EX1: Schedule with different machine speeds and new due dates

As compared with Fig. 2, Operation 2 of part 2 and part 3 are processed on machine 2 under the fast speed to improve on-time deliveries. Considering energy costs, the optimized schedule is the same as Fig. 3. The reason is that the energy cost cannot be reduced under the same delivery dates since on-time deliveries have more priority. It implies that when due dates are tight, flexible machine speeds can still improve on-time deliveries, but the energy cost cannot be much reduced. If the energy priority is increased, a schedule with a lower energy cost can be obtained by postponing deliveries.

All the above problems can be solved by using CPLEX in less than 1 second with a MIP gap of 0 %.

Example 2: Medium-sized problem

This example is to demonstrate effectiveness of formulation tightening. The instance is created based on the first 50 parts and all machines in [13]. According to which parts/operations that machines can process, machines are categorized into 19 types, and each type has 1 to 6 machines. The number of time slots under consideration is 200 so that all the parts can be processed. Machines are assumed always available for simplicity, and each machine has three speed levels. There are three values for tardiness weights, \$1, \$10, and \$100, and they are randomly assigned to parts with percentage of 50%, 40% and 10%, respectively. Before and after adding tightened constraints, the overall job-shop scheduling problems are solved by using B&C, and results are shown in Table II below. Stopping criteria are 1200 second (s) CPU time or 1% MIP gap (as the gap may not be reduced in a continuous way, the final gap might be less than 1%).

According to Table II, the CPU time is much reduced by adding tightened constraints Eqs. (16) and (19), while the

solution quality is still high. With the original formulation, a feasible solution with a total tardiness cost of \$35,655 and an energy cost of \$1,803 is obtained in 789s. After tightening, a feasible solution with a little lower total tardiness cost and a higher energy cost is obtained in 72s, demonstrating the effectiveness of formulation tightening.

TABLE II EX2: COMPARISON OF FORMULATIONS

Formulation	Tardiness (\$)	Energy (\$)	MIP gap (%)	CPU (s)
(a): Original	35,655	1,803	1	789
(b): (a) + (16a, b)	35,715	1,855	0.85	114
(c): (b) + (19)	35,612	1,890	0.83	72

Example 3: Large-sized problem

This example is to demonstrate computational efficiency of SAVLR + B&C. The problem is taken from [13] with 127 parts, and 19 machines with five speed levels. The number of time slots under consideration is 300. Other settings are the same as in Example 2. With and without tightening, the problem is first solved by using B&C, and no solution is found after one hour. Then the problem is solved by using SAVLR + B&C, where the stopping criterion is 0 for constraint violations. For subproblems, the stopping time and gap is 60s and 0.5%. Testing results are shown in Fig. 4 as follows.

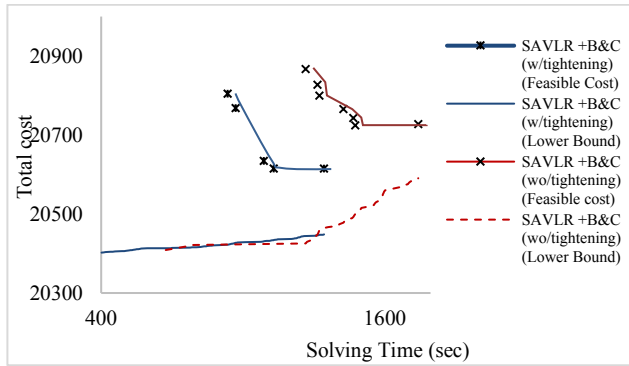


Figure. 4. EX3: Results of SAVLR+B&C

With the original formulation, SAVLR + B&C obtains a feasible solution of \$20,724 (the total tardiness penalty and energy cost) with a duality gap of 0.6 %, and the solving time (exclude model/data loading and data outputting time from CPU time) is 1884 s. After tightening, a feasible solution of \$20,614 with a duality gap of 0.8 % is obtained and the solving time is 1188 s. Results demonstrate computational efficiency of SAVLR + B&C since B&C cannot even find a feasible solution in one hour, as well as significant benefits of tightening on computational efficiency and solution quality.

VII. CONCLUSION

This paper is to obtain near-optimal solutions efficiently for energy-efficient job-shop scheduling to achieve on-time deliveries and low energy costs. The problem is formulated in an MILP form to make effective use of available MILP methods. This is done by modeling machines in groups for simplicity while approximating energy costs, and by linking part processing status and machine speed variables. The formulation is then transformed for better computational efficiency by extending our previous tightening approach for machines with constant speeds. To solve the problem more

efficiently, our advanced decomposition and coordination method is used. Numerical results show near-optimal solutions are obtained, demonstrating significant benefits of our approach on on-time deliveries and energy costs. For future work, approximation of energy costs will be improved, and comparison with existing models and methods will be performed for validation purposes.

REFERENCES

- [1] T. Jiang, C. Zhang, H. Zhu, J. Gu, and D. Deng, "Energy-efficient scheduling for a job shop using an improved whale optimization algorithm," *Mathematics*, Vol. 6, No. 11, 2018.
- [2] B. Yan, M. A. Bragin, and P. B. Luh, "Novel Formulation and Resolution of Job-Shop Scheduling Problems," *IEEE Robot. Autom. Lett.*, Vol. 3, Issue 4, pp. 3387 - 3393, 2018.
- [3] B. Yan, M. A. Bragin, and P.B. Luh, "An Innovative Formulation Tightening Approach for Job-Shop Scheduling," to be submitted to *IEEE Trans. Autom. Sci. Eng.*
- [4] M. A. Bragin, P. B. Luh, B. Yan, and X. Sun "A Scalable Solution Methodology for Mixed-Integer Linear Optimization Problems Arising in Automation," *IEEE Trans. Autom. Sci. Eng.*, Vol.16, Issue 2, pp. 531 - 541, 2019.
- [5] L. Yin, X. Li, L. Gao, C. Lu, and Z. Zhang, "Energy-efficient job shop scheduling problem with variable spindle speed using a novel multi-objective algorithm," *Adv. Mech. Eng.*, Vol. 9, No. 4, pp. 1-1, 2017.
- [6] L. Zhang, Q. Tang, Z. Wu, and F. Wang, "Mathematical modeling and evolutionary generation of rule sets for energy-efficient flexible job shops," *Energy*, vol. 138, pp. 210-227, Nov. 2017.
- [7] L. Meng, C. Zhang, B. Zhang, and Y. Ren, "Mathematical modeling and optimization of energy-conscious flexible job shop scheduling problem with worker flexibility," *IEEE Access*, Vol. 7, pp. 68043-68059, 2019.
- [8] L. Meng, C. Zhang, X. Shao, and Y. Ren, "MILP models for energy-aware flexible job shop scheduling problem," *J. Clean. Prod.*, Vol. 210, pp.710-723, 2019.
- [9] L. Meng, C. Zhang, Y. Ren, B. Zhang, and V. Lv, "Mixed-integer linear programming and constraint programming formulations for solving distributed flexible job shop scheduling problem," *Comput. Ind. Eng.*, Vol. 142, 106347, 2020.
- [10] M. Karimi-Nasab, and M. Modarres, "Lot sizing and job shop scheduling with compressible process times: a cut and branch approach," *Comput. Ind. Eng.*, Vol. 85, pp. 196-205, 2015.
- [11] R. Z. Rios-Mercado, and J. F. Bard, "Computational experience with a branch-and-cut algorithm for flowshop scheduling with setups," *Comput. Oper. Res.*, Vol. 25, No. 5, pp. 351-366, 1998.
- [12] M. Nouri, A. Bekrar, A. Jemai, S. Niar, and A. C. Ammari, "An effective and distributed particle swarm optimization algorithm for flexible job-shop scheduling problem," *J. Intell. Manuf.*, 2015 Feb., pp. 1-13.
- [13] D. J. Hootomt, P. B. Luh, K. R. Pattipati, "A practical approach to job shop scheduling problems," *IEEE Trans. Autom. Sci. Eng.*, Vol. 9, No. 1, pp. 1-13, 1993.
- [14] C. A. Kaskavelis, and M. C. Caramanis, "Efficient Lagrangian relaxation algorithms for industry size job-shop scheduling problems," *IIE transacts.*, Vol. 30, No. 11, pp. 1085-1097, 1998.
- [15] H. Chen, C. Chu, and J. M. Proth, "An improvement of the Lagrangean relaxation approach for job shop scheduling: a dynamic programming method," *IEEE Trans. Autom. Sci. Eng.*, Vol. 14, No. 5, pp. 786-795, 1998.
- [16] M. A. Bragin, P. B. Luh, J. H. Yan, N. Yu, and G. A. Stern, "Convergence of the surrogate Lagrangian relaxation method," *J. Optimiz. Theory. App.*, Vol. 164, No. 1, pp. 173-201, 2015.
- [17] M. A. Bragin, P. B. Luh, B. Yan, and X. Sun, "A scalable solution methodology for mixed-integer linear programming problems arising in automation," *IEEE Trans. Autom. Sci. Eng.*, Vol. 16, No. 2, pp. 531 - 541, 2019.
- [18] E. M. L. Beale and J. J. H. Forrest, "Global optimization using special ordered sets," *Math. Program.*, Vol. 10, No. 1, pp. 52-69, 1976.
- [19] Heidelberg University, <http://www.iwr.uni-heidelberg.de/groups/comopt/software/PORTA/>
- [20] IBM ILG CPLEX V 12.1 User's Manual.