Energy-Aware Production Scheduling with Power-Saving Modes

Ondřej Benedikt[1], Přemysl Šůcha[1], István Módos[1], Marek Vlk[1 2], and Zdeněk Hanzálek[1]

[1] Czech Technical University in Prague, Czech Republic
benedond@fel.cvut.cz
{premysl.sucha,istvan.modos,zdenek.hanzalek}@cvut.cz
[2] Charles University, Czech Republic
vlk@ktiml.mff.cuni.cz

Abstract. This study addresses optimization of production processes where machines have high energy consumption. One efficient way to reduce the energy expenses in production is to turn a machine off when it is not being used or switch it into an energy-saving mode. If the production has several machines and production demand that varies in time, the energy saving can be substantial; the cost reduction can be achieved by an appropriate production schedule that could control the switching between the energy modes with respect to the required production volume. Therefore, inspired by real production processes of glass tempering and steel hardening, this paper addresses the scheduling of jobs with release times and deadlines on parallel machines. The objective is to find a schedule of the jobs and a switching between the power modes of the machines so that the total energy consumption is minimized. Moreover, to further generalize the scheduling problem to other production processes, we assume that the processing time of the jobs is mode-dependent, i.e., the processing time of a job depends on the mode in which a machine is operating. The study provides an efficient Branch-and-Price algorithm and compares two approaches (based on Integer Linear Programming and Constraint Programming) for solving the subproblem.

Keywords: Production scheduling, Energy, Branch-and-Price, Integer Linear Programming, Constraint Programming

1 Introduction

This research is inspired by two existing energy-demanding production processes. The first one is a glass tempering in ERTL Glas company and the second one is steel hardening in ŠKODA AUTO company. In both processes, the material is heated in one of the identical furnaces to a high temperature (hundreds of °C) which consumes a substantial amount of energy. Typically, the furnaces are turned on at the beginning of a scheduling horizon and then continuously operate until its end when they are turned off. If the production demand varies within the scheduling horizon, the furnaces remain in an energy-demanding mode even if nothing is being produced and, therefore, wasting the energy.

As identified in [1], a significant energy cost savings could be achieved in manufacturing facilities by switching machines to a power-saving mode when nothing is being produced. Likewise, our preliminary feasibility study for ŠKODA AUTO has shown that about 6 % of the production line consumption can be saved using the power-saving modes. However, in the above-mentioned processes, switching to and from the power-saving mode is not immediate since the furnaces in the power-saving mode operate in a lower temperature and re-heating them back to the operational temperature can take dozens of minutes. Thus, the switching has to be planned carefully. Moreover, in some production processes a machine operating in a power-saving mode can still process a material, albeit slower. To take into consideration such processes, we will assume that the processing time is mode-dependent, i.e., the processing time depends on the mode in which a machine is operating.

The problem of scheduling jobs on machines having different energy modes has been already studied in the literature to some extent. The existing works either study a single machine problem [1–3] or propose a time-indexed Mixed Integer Linear Programming formulation [4, 5] which can optimally solve only small instances. The scheduling with mode-dependent processing times of the jobs is similar to a dynamic voltage scaling [6] in embedded systems, where the processing times of the jobs depend on the operating frequencies of the processors. Since the schedules in the embedded systems are usually event-triggered and the transition times between different operating frequencies is in the order of microseconds, the research results cannot be directly applied to the production processes.

The contribution of this work-in-progress paper is both a formulation and algorithm for solving a general multi-machine scheduling problem with energy modes, where the objective is to minimize the total energy consumption. For this problem, we propose an efficient Branch-and-Price algorithm that breaks the symmetries arising due to the identical parallel machines. The algorithm also restricts the structure of transitions between the modes to respect the technological requirements. For the experimental comparison, the subproblem in the Branch-and-Price algorithm is formulated as both Integer Linear Programming (ILP) and Constraint Programming (CP) problems.

2 Problem Statement

Let $M = \{1, \dots, m\}$ be a set of identical, parallel machines. In every time instant, every machine $i \in M$ is operating in some *mode* $\omega \in \Omega$. While a machine is operating in mode $\omega \in \Omega$, it demands a constant *power* $P_\omega \in \mathcal{R}_{\geq 0}$. The mode of a machine can be switched from one mode to another, however, this transition may take some time during which the machine is not operational, and it incurs a cost in the form of consumed energy. Therefore, for every pair of modes $\omega, \omega' \in \Omega$ we define *transition time* $t^{\text{trans}}_{\omega,\omega'} \in \mathcal{Z}_{\geq 0} \cup \{\infty\}$ and *transition cost*

$c^{\text{trans}}_{\omega,\omega'} \in \mathcal{R}_{\geq 0} \cup \{\infty\}$. If $t^{\text{trans}}_{\omega,\omega'} = c^{\text{trans}}_{\omega,\omega'} = \infty$ for some pair of modes $\omega, \omega' \in \Omega$, then a machine cannot be directly switched from ω to ω'. If a machine is operating in some mode $\omega \in \Omega$ for a total time of t, then the *operating cost* is computed as $P_\omega \cdot t$.

Let $J = \{1, \dots, n\}$ be a set of jobs. The jobs have to be processed on some machine within scheduling *horizon* $H \in \mathcal{Z}_{>0}$, each machine can process at most one job at a time, and the jobs cannot be preempted. A *processing time* of job $j \in J$ depends on the mode $\omega \in \Omega$ in which the assigned machine is operating during processing of the job and is denoted as $p_{j,\omega} \in \mathcal{R}_{\geq 0} \cup \{\infty\}$. If $p_{j,\omega} = \infty$, then job $j \in J$ cannot be processed while the assigned machine is operating in mode $\omega \in \Omega$. A job cannot be processed on a machine during the transition from one mode to another and once a machine starts processing a job, it cannot change its mode until the job completes. Moreover, each job $j \in J$ has *release time* $r_j \in \mathcal{Z}_{\geq 0}$ and *deadline* $d_j \in \mathcal{Z}_{\geq 0}$. The release time and deadline of a job define the time window within which it must be processed.

A *solution* is a tuple $(\boldsymbol{s}, \boldsymbol{\mu}, \boldsymbol{\pi}_1, \dots, \boldsymbol{\pi}_m, \boldsymbol{t}^{\text{mode}}_1, \dots, \boldsymbol{t}^{\text{mode}}_m)$, where $\boldsymbol{s} \in \mathcal{Z}^n_{\geq 0}$ is a vector of jobs *start times*, $\boldsymbol{\mu} \in M^n$ is a vector of jobs *assignment to the machines*, $\boldsymbol{\pi}_i \in \bigcup_{l \in \mathcal{Z}_{>0}} \Omega^l$ is a *profile* of machine $i \in M$ and $\boldsymbol{t}^{\text{mode}}_i \in \bigcup_{l \in \mathcal{Z}_{>0}} \mathcal{Z}^l_{\geq 0}$ are the *operating times* of machine $i \in M$. Profile $\boldsymbol{\pi}_i$ is a finite sequence of modes which are followed by machine $i \in M$ in the solution. The profiles represent the transition from one mode to another; they do not inform about the time spent operating in a particular mode of a profile. Operating times $\boldsymbol{t}^{\text{mode}}_i$ are a finite sequence of non-negative integers such that $t^{\text{mode}}_{i,k}$ is the operating time of machine $i \in M$ in mode $\pi_{i,k}$. It holds that (i) the length of a profile is the same as the length of the corresponding operating times, i.e., $|\boldsymbol{\pi}_i| = |\boldsymbol{t}^{\text{mode}}_i|$, $\forall i \in M$ (operator $|\cdot|$ represents the length of a sequence) and (ii) the sum of the total transition times plus the total operating times of a profile equals to the length of the horizon, i.e., $\sum_{k=1}^{|\boldsymbol{\pi}_i|-1} t^{\text{trans}}_{\pi_{i,k},\pi_{i,k+1}} + \sum_{k=1}^{|\boldsymbol{\pi}_i|} t^{\text{mode}}_{i,k} = H$, $\forall i \in M$.

The goal of this scheduling problem is to find a solution which minimizes the consumed energy, i.e., the sum of the total transition cost plus the total operating cost

$$\sum_{i \in M} \sum_{k=1}^{|\boldsymbol{\pi}_i|-1} c^{\text{trans}}_{\pi_{i,k},\pi_{i,k+1}} + \sum_{i \in M} \sum_{k=1}^{|\boldsymbol{\pi}_i|} P_{\pi_{i,k}} \cdot t^{\text{mode}}_{i,k} \,. \tag{1}$$

This scheduling problem is strongly $\mathcal{NP}$-hard due to the underlying strongly $\mathcal{NP}$-complete scheduling problem $1|r_j, d_j|-$ [7].

As an example, consider a single machine problem with 3 jobs, $H = 28$ and $\Omega = \{\text{OFF}, \text{SLOW}, \text{FAST}\}$. Assume that for all jobs $j \in J$ the processing times are $p_{j,\text{OFF}} = \infty$, $p_{j,\text{SLOW}} = 5$ and $p_{j,\text{FAST}} = 2$. All jobs are released at time 0 and the deadline of all jobs is H. A possible solution (not optimal) to this simple problem is $((11, 21, 8), (1, 1, 1), (\text{OFF}, \text{FAST}, \text{SLOW}, \text{FAST}, \text{OFF}), (3, 3, 6, 5, 2))$, which is illustrated in Fig. 1. The figure shows a schedule of the jobs with the visualization of the transitions between the modes of the solution profile.

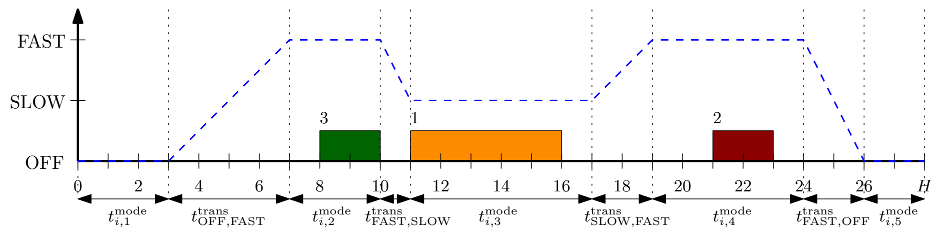

Fig. 1. The solution with profile $(\text{OFF}, \text{FAST}, \text{SLOW}, \text{FAST}, \text{OFF})$ for the example problem. The dashed polyline represents the profile and the mode transitions.

3 Solution Approach

Although it is possible to model the scheduling problem introduced in Sect. 2 using either ILP or CP, the resulting model would be very large and could solve only small instances. Moreover, due to the parallel identical machines, the scheduling problem has symmetries, which significantly degrade the efficiency of the models. Therefore, in this study, we solve the scheduling problem using Branch-and-Price (BaP) methodology [8] which is specifically designed for solving such large-scale optimization problems.

Algorithms based on BaP combine the Branch-and-Bound algorithm with Column Generation (CG) [9]. The CG is a technique for solving Linear Programming (LP) models with many variables. The variables (and the corresponding columns of the constraint matrix) are added lazily until a solution with the restricted set of the variables is optimal for the dual formulation of the model. If the solution is an integer, the whole algorithm terminates. Otherwise, branching on the fractional variables is performed.

The LP model used in the CG is based on a set covering model. Let $A \subseteq \{0,1\}^n$ be a set of columns, where each *column* $\boldsymbol{a}_l \in A$ represents a particular assignment of the jobs on a machine. For each $\boldsymbol{a}_l \in A$ we can compute its *cost* c_l^{col}, which corresponds to the optimal solution of the single machine scheduling problem with jobs for which $a_{l,j} = 1$. The goal of the *master problem* is to select a subset of columns from A such that every job is assigned to some machine and the total cost of the selected columns is minimized. Since the number of columns is exponential in the number of jobs, we use *restricted column set* $A' \subseteq A$, which is lazily expanded using the CG. The master problem *restricted* to A' then can be modeled as

$$\min \quad \sum_{\boldsymbol{a}_l \in A'} c_l^{\text{col}} \cdot x_l \tag{2}$$

$$\text{s.t.} \quad \sum_{\boldsymbol{a}_l \in A'} a_{l,j} \cdot x_l \geq 1,\ j \in J \tag{3}$$

$$\sum_{\boldsymbol{a}_l \in A'} x_l \leq m \tag{4}$$

$$x_l \geq 0,\ \boldsymbol{a}_l \in A'\ . \tag{5}$$

Variable x_l denotes, whether column $\boldsymbol{a}_l$ is selected in the solution. Notice that the problematic machine symmetries are broken since an assignment of a column to a specific machine is not important and, therefore, not modeled.

The CG operates on the dual formulation of the restricted master problem; consequently, the variables x_l diminish. Instead, new variables $\boldsymbol{\lambda} \in \mathcal{R}_{\leq 0} \times \mathcal{R}_{\geq 0}^n$ arise in the dual formulation. If the optimal solution to the dual restricted master problem is feasible for the unrestricted one, the CG terminates, otherwise, column $\boldsymbol{a}_l \in A \setminus A'$ with a negative *reduced cost* has to be found, i.e., $0 > c_l^{\text{col}} - \sum_{j \in J} a_{l,j} \cdot \lambda_j - \lambda_0$. A new column for the restricted master problem is found using another optimization model called a *subproblem* in which the dual variables $\boldsymbol{\lambda}$ are fixed. Solving such a problem is very similar to solving a single machine variant of the scheduling problem introduced in Sect. 2. The difference is that the subproblem selects a subset of jobs, i.e., new column $\boldsymbol{a}_l$, such that the reduced cost is minimized.

However, such a subproblem has still a very large solution space, thus selecting the machine profiles is a hard combinatorial problem. Moreover, due to technological reasons, such as machine wear, the optimal solution to the scheduling problem with many transitions might not be feasible in practice. A solution to both these issues is to require the machine to follow one of the predetermined, technologically feasible profiles $\Pi \subset \bigcup_{l \in \mathcal{Z}_{>0}} \Omega^l$. The idea is then to solve the subproblem for every $\boldsymbol{\pi} \in \Pi$ and select the one with the minimum objective. We solve the subproblem using the following ILP model

$$\min \quad \sum_{i \in M} \sum_{k=1}^{|\boldsymbol{\pi}|-1} c^{\text{trans}}_{\pi_k, \pi_{k+1}} + \sum_{i \in M} \sum_{k=1}^{|\boldsymbol{\pi}|} P_{\pi_k} \cdot t_k^{\text{mode}} - \sum_{j \in J} \sum_{k=1}^{|\boldsymbol{\pi}|} y_{j,k} \cdot \lambda_j - \lambda_0 \tag{6}$$

$$\sum_{k=1}^{|\boldsymbol{\pi}|} y_{j,k} \leq 1, \ j \in J \tag{7}$$

$$s_1^{\text{mode}} = 0 \tag{8}$$

$$s_k^{\text{mode}} = s_{k-1}^{\text{mode}} + t_{k-1}^{\text{mode}} + t^{\text{trans}}_{\pi_{k-1}, \pi_k}, \ k \in \{2, \dots, |\boldsymbol{\pi}|\} \tag{9}$$

$$s_{|\boldsymbol{\pi}|}^{\text{mode}} + t_{|\boldsymbol{\pi}|}^{\text{mode}} = H \tag{10}$$

$$r_j \leq s_j, \ j \in J \tag{11}$$

$$s_j + \sum_{k=1}^{|\boldsymbol{\pi}|} y_{j,k} \cdot p_{j,\pi_k} \leq d_j, \ j \in J \tag{12}$$

$$s_k^{\text{mode}} \leq s_j + \mathcal{M} \cdot (1 - y_{j,k}), \ j \in J, k \in \{1, \dots, |\boldsymbol{\pi}|\} \tag{13}$$

$$s_j + p_{j,\pi_k} \leq s_k^{\text{mode}} + t_k^{\text{mode}} + \mathcal{M} \cdot (1 - y_{j,k}), \quad j \in J, k \in \{1, \dots, |\boldsymbol{\pi}|\} \tag{14}$$

$$s_j + p_{j,\pi_k} \leq s_{j'} + \mathcal{M} \cdot (3 - y_{j,k} - y_{j',k} - z_{j,j'}), \quad j, j' \in J, j < j', k \in \{1, \dots, |\boldsymbol{\pi}|\} \tag{15}$$

$$s_{j'} + p_{j',\pi_k} \leq s_j + \mathcal{M} \cdot (2 - y_{j',k} - y_{j,k} + z_{j,j'}), \quad j, j' \in J, j < j', k \in \{1, \dots, |\boldsymbol{\pi}|\} \ . \tag{16}$$

The program uses the following variables: (i) $y_{j,k} \in \{0,1\}$ denoting whether $j \in J$ is assigned to k-th mode of $\boldsymbol{\pi}$, (ii) $s_k^{\text{mode}} \in \mathcal{Z}_{\geq 0}$ is the start time of the time interval in which the machine is operating in k-th mode of profile $\boldsymbol{\pi}$, (iii) $t_k^{\text{mode}} \in \mathcal{Z}_{\geq 0}$ is the operating time of k-th mode of profile $\boldsymbol{\pi}$, (iv) $s_j \in \mathcal{Z}_{\geq 0}$ is the start time of job $j \in J$ and (v) $z_{j,j'} \in \{0,1\}$ denotes the relative order between jobs $j, j' \in J$. To see that this subproblem selects a column, notice that $a_{l,j} = \sum_{k=1}^{|\boldsymbol{\pi}|} y_{j,k}$. Constraint (7) allows each job to be assigned to at most one mode, constraints (8)–(10) set the start time and operating time of k-th mode, constraints (11)–(12) assure that the jobs are processed in between its release time and deadline, constraints (13)–(14) ensure that the jobs are fully contained in the time period of k-th mode to which they are assigned, and constraints (15)–(16) ensure that the jobs are not overlapping. To speed-up solving the model, we generate constraints (15)–(16) using *lazy constraints generation.*

The initial set of columns is created using a simple heuristic based on Earliest Deadline First strategy. Since even problem $1|r_j, d_j|-$ is strongly $\mathcal{NP}$-complete, there is no guarantee that any heuristic will find the initial A' that will represent a feasible solution (although the selected Earliest Deadline First strategy is generally better aimed at satisfying the deadlines than cost-based heuristics). Therefore, if the heuristic cannot find a feasible solution, it generates A' such that it contains columns covering all the jobs. In such a situation, it may happen that the master model may not find a feasible solution because of constraint (4). Therefore the master model has to assume this constraint in a slightly different form, i.e., $\sum_{\boldsymbol{a}_l \in A'} x_l \leq m + q$, where $q \geq 0$ is a new decision variable indicating whether the solution of the master problem is feasible or not. Finally, the objective function of the master model is $\sum_{\boldsymbol{a}_l \in A'} c_l^{\text{col}} \cdot x_l + q \cdot \mathcal{C}$, such that $\mathcal{C}$ is much larger than the cost of any feasible column, e.g., $\mathcal{C} = m \cdot H \cdot \max_{\omega \in \Omega} P_\omega + \max_{\boldsymbol{\pi} \in \Pi} |\boldsymbol{\pi}| \cdot \max_{\omega,\omega' \in \Omega} c_{\omega,\omega'}^{\text{trans}}$.

If the optimal solution to the master problem is fractional at the end of the CG, a branching is required. The used branching scheme selects a pair of jobs (j, j') that have not been selected before and have the largest overlap of intervals $[r_j, d_j]$ and $[r_{j'}, d_{j'}]$ (according to preliminary experiments, this branching scheme performed better than random selection). Then the scheme creates two branches, where: (i) jobs j and j' are forbidden to be processed on the same machine and (ii) the same two jobs are required to be processed on the same machine. This scheme generates simple logical constraints that can be included into the subproblem. For each new branch it is necessary to filter out columns $\boldsymbol{a}_l \in A'$ that violate the particular branching decision. However, this step may result in a column set which is not covering all the jobs. Therefore, as in the initialization phase, the algorithm has to add columns such that all jobs are present in A'.

Alternatively, the subproblem can be easily modeled using CP, where efficient filtering techniques for unary resources with optional jobs are employed [10]. For each $k \in \{1, \ldots, |\boldsymbol{\pi}|\}$, let us introduce interval variable I_k^{mode}, and for each $j \in J, k \in \{1, \ldots, |\boldsymbol{\pi}|\}$, let us introduce interval variable $I_{j,k}$ that is set to be optional, which means that the presence of $I_{j,k}$ in a schedule is to be decided. The length of $I_{j,k}$ is fixed to p_{j,π_k}, whereas the length of I_k^{mode} is to be determined. Constraints (15)–(16) are substituted by no-overlap constraints, i.e., for each $k \in \{1, \ldots, |\boldsymbol{\pi}|\}$, we add constraint $\mathit{NoOverlap}(\bigcup_{j\in J} I_{j,k})$. The other constraints are straightforward. Note that the state function variable [11] for modeling the modes of a machine cannot be efficiently used as the lengths of the modes are involved in the objective function and the transition times between modes are fixed, thus the interval variables for modes are inevitable.

4 Preliminary Experiments

We evaluated the proposed Branch-and-Price algorithm (with subproblem implemented as both ILP and CP model) on a set of random problem instances that were generated as follows. The scheduling horizon was fixed to $H = 1000$ and the set of assumed machines modes was chosen as $\Omega = \{\text{OFF}, \text{IDLE}, \text{ON}\}$, i.e., the machines have one power-saving mode IDLE and the jobs can be processed only in mode ON. The set of technologically feasible profiles is $\Pi = \{(\text{OFF}, \text{ON}, \text{OFF}), (\text{OFF}, \text{ON}, \text{IDLE}, \text{ON}, \text{OFF})\}$. The processing time of the jobs in mode ON was randomly sampled from discrete uniform distribution $\mathcal{U}\{1, 100\}$. The release times and deadlines were randomly generated in such a way that the generated instances were feasible and the jobs had non-zero overlap.

For each pair $n \in \{15, 20, 25\}, m \in \{1, 2, 3, 4\}$, 4 random instances were generated using the scheme described above; each instance had time-limit of 3600 s. The experiments were carried out on an Intel® Core$^{\text{TM}}$ i5-3320M CPU @ 2.6 GHz computer with 8 GB RAM. For solving the ILP and CP models, we used Gurobi 7.5 and IBM CP Optimizer 12.7.1 solvers, respectively.

The results shown in Tab. 1 clearly indicate that the Branch-and-Price with ILP subproblem (BaP+ILP) outperforms the CP subproblem (BaP+CP). Using continuous variables for the start times of the jobs instead of integer variables led to only a slight deterioration in the computational time, while the number of nodes and the number of columns slightly decreased.

Although it could be possible to compare the proposed algorithm with the time-indexed Mixed Integer Linear Programming formulation from the literature [2], the resulting model would be huge for the scheduling granularity usually used in production scheduling (1 minute). For example, the time-indexed Mixed Integer Linear Programming formulation would require $4 \cdot 25 \cdot 1000 = 100000$ binary variables just to represent the start times of the jobs for the largest problem instance from Tab. 1.

Table 1: Experimental results.

Instance	Parameters		BaP+ILP			BaP+CP		
	m	n	Computational time [s]	Nodes	Columns	Computational time [s]	Nodes	Columns
1	1	15	9.50	1	113	58.57	1	115
2	1	15	9.58	1	116	92.80	1	124
3	1	15	9.38	1	102	41.77	1	82
4	1	15	11.37	1	132	50.71	1	118
5	2	15	31.15	13	253	179.53	13	270
6	2	15	14.91	1	93	65.10	1	66

7	2	15	8.66	1	97	45.95	1	90
8	2	15	10.23	1	87	79.79	1	102
9	3	15	40.05	17	173	177.20	17	156
10	3	15	41.59	35	301	210.12	35	341
11	3	15	5.29	1	47	60.41	1	48
12	3	15	24.37	15	200	188.80	15	187
13	4	15	32.43	31	243	278.96	33	213
14	4	15	4.82	1	51	44.23	1	54
15	4	15	7.46	1	77	87.70	1	75
16	4	15	5.43	1	38	45.72	1	42
17	1	20	38.78	1	298	250.04	1	250
18	1	20	79.39	1	323	406.87	1	302
19	1	20	86.69	1	429	453.85	1	437
20	1	20	17.57	1	166	267.30	1	252
21	2	20	372.76	71	1,357	1,719.14	79	1,263
22	2	20	35.91	1	187	169.87	1	118
23	2	20	97.80	3	345	355.56	3	291
24	2	20	195.30	15	413	927.75	15	465
25	3	20	65.91	1	146	288.09	1	144
26	3	20	35.79	1	143	179.21	1	114
27	3	20	38.25	1	156	193.89	1	134
28	3	20	161.99	51	537	1,803.98	385	1,263
29	4	20	30.26	1	94	254.23	1	119
30	4	20	143.86	61	511	965.07	71	623
31	4	20	27.83	1	81	155.68	1	72
32	4	20	81.48	15	213	696.28	15	265
33	1	25	99.17	1	274	383.83	1	258
34	1	25	181.60	1	329	1,008.59	1	288
35	1	25	58.33	1	249	410.03	1	262
36	1	25	87.39	1	287	266.63	1	249
37	2	25	2,362.55	89	4,685	$>$ 3,600.00	1	168
38	2	25	125.78	1	319	647.76	1	314
39	2	25	795.59	101	2,164	2,383.94	53	1,438
40	2	25	111.70	1	273	714.10	1	234
41	3	25	516.39	21	806	1,913.10	21	794
42	3	25	277.58	15	634	1,142.37	15	530
43	3	25	2,657.06	181	4,393	$>$ 3,600.00	61	1,172
44	3	25	1,184.67	39	914	$>$ 3,600.00	39	730
45	4	25	162.21	1	230	755.30	1	258
46	4	25	226.80	49	663	$>$ 3,600.00	221	1,758
47	4	25	99.28	1	170	727.78	1	223
48	4	25	596.11	33	663	3,472.82	79	1,349

5 Conclusion

Reducing energy consumption costs of manufacturing processes can be a significant competitive advantage for producers. This work-in-progress paper provides a Branch-and-Price algorithm for a multi-machine production scheduling problem minimizing energy consumption. The experimental results show that the algorithm can solve instances with four machines and up to 25 jobs in a reasonable time. To be able to solve real production instances, the algorithm can be easily transformed into a heuristic, e.g., by reducing branching. Nevertheless, we work on further improvements of the exact algorithm to make it applicable to larger problem instances. We aim to reduce the depth of the search tree generated by the Branch-and-Bound by inferring infeasible branches using constraint propagation. Moreover, the computational time of the subproblem could be decreased by an online machine learning algorithm [12] that reuses the results of previously solved subproblems.

References

1. Mouzon, G., Yildirim, M.B., Twomey, J.: Operational methods for minimization of energy consumption of manufacturing equipment. International Journal of Production Research **45**(18–19) (2007) 4247–4271
2. Shrouf, F., Ordieres-Meré, J., García-Sánchez, A., Ortega-Mier, M.: Optimizing the production scheduling of a single machine to minimize total energy consumption costs. Journal of Cleaner Production **67**(Supplement C) (2014) 197 – 207
3. Gong, X., der Wee, M.V., Pessemier, T.D., Verbrugge, S., Colle, D., Martens, L., Joseph, W.: Integrating labor awareness to energy-efficient production scheduling under real-time electricity pricing: An empirical study. Journal of Cleaner Production **168**(Supplement C) (2017) 239 – 253
4. Selmair, M., Claus, T., Trost, M., Bley, A., Herrmann, F.: Job shop scheduling with flexible energy prices. In: European Conference for Modelling and Simulation. (2016)
5. Mitra, S., Sun, L., Grossmann, I.E.: Optimal scheduling of industrial combined heat and power plants under time-sensitive electricity prices. Energy **54**(Supplement C) (2013) 194 – 211
6. Kong, F., Wang, Y., Deng, Q., Yi, W.: Minimizing multi-resource energy for real-time systems with discrete operation modes. In: 2010 22nd Euromicro Conference on Real-Time Systems. (July 2010) 113–122
7. Lenstra, J., Kan, A.R., Brucker, P.: Complexity of machine scheduling problems. In Hammer, P., Johnson, E., Korte, B., Nemhauser, G., eds.: Studies in Integer Programming. Volume 1 of Annals of Discrete Mathematics. Elsevier (1977) 343 – 362
8. Feillet, D.: A tutorial on column generation and branch-and-price for vehicle routing problems. 4OR **8**(4) (Dec 2010) 407–424
9. Desrosiers, J., Lübbecke, M.E. In: A Primer in Column Generation. Springer US, Boston, MA (2005) 1–32
10. Vilím, P., Barták, R., Čepek, O.: Extension of o (n log n) filtering algorithms for the unary resource constraint to optional activities. Constraints **10**(4) (2005) 403–425
11. Laborie, P., Rogerie, J., Shaw, P., Vilím, P.: Reasoning with conditional time-intervals. part ii: An algebraical model for resources. In: FLAIRS conference. (2009) 201–206
12. Václavík, R., Novák, A., Šůcha, P., Hanzálek, Z.: Accelerating the branch-and-price algorithm using machine learning. European Journal of Operational Research (Oct 2017) under review.