Robust Scheduling for Manufacturing with Energy Consumption Limits

István Módos*†, Přemysl Šůcha† and Zdeněk Hanzálek*†
*Czech Institute of Informatics, Robotics and Cybernetics
Czech Technical University in Prague
Czech Republic
†Department of Control Engineering, Faculty of Electrical Engineering
Czech Technical University in Prague
Czech Republic

***Abstract*—Our work considers a scheduling problem in which manufacturing companies with large energy demand are obligated to comply with total energy consumption limits in specified time intervals, e.g. 15 minutes. Moreover, the problem is complicated by the fact that in reality the production schedules are not executed exactly as planned due to unexpected disturbances such as machine breakdowns or material unavailability. Therefore, the goal is to find a robust schedule which guarantees that the energy consumption limits are not violated if the start times of operations are arbitrary delayed within a given limit. To circumvent the problem of an exponential number of constraints in the mixed integer linear programming formulation, we propose an exact algorithm based on a decomposition approach. The decomposition approach exploits the fact that the robustness of a given schedule can be checked in a pseudo-polynomial time. We evaluated the proposed algorithm on instances with varying bound of the start times delays.**

I. Introduction

This work focuses on a scheduling problem faced by manufacturing companies with large energy consumption. In this problem some of the operations that have to be scheduled on the available machines consume a substantial amount of electric energy. Therefore, the electricity provider needs to know the amount of electric energy required by the manufacturing company so that the maximum demand of the company can be guaranteed. This information is provided by the company to the electricity provider in form of *load curves*, which represent how much energy is demanded in specific time intervals during the day. Based on the load curves, the electricity provider computes a *maximum energy consumption limit* in each 15 minutes *metering interval* that must not be exceeded by the customer otherwise he pays a large penalty fee. Therefore, an energy-aware schedule of the operations for the machines is needed so that the energy limits are not violated; such a schedule is called *baseline*. The baseline schedule may contain idle times between the operations with a high energy demand so that the energy consumption is spread among neighbouring 15 minutes metering intervals, e.g. see Fig. 1a.

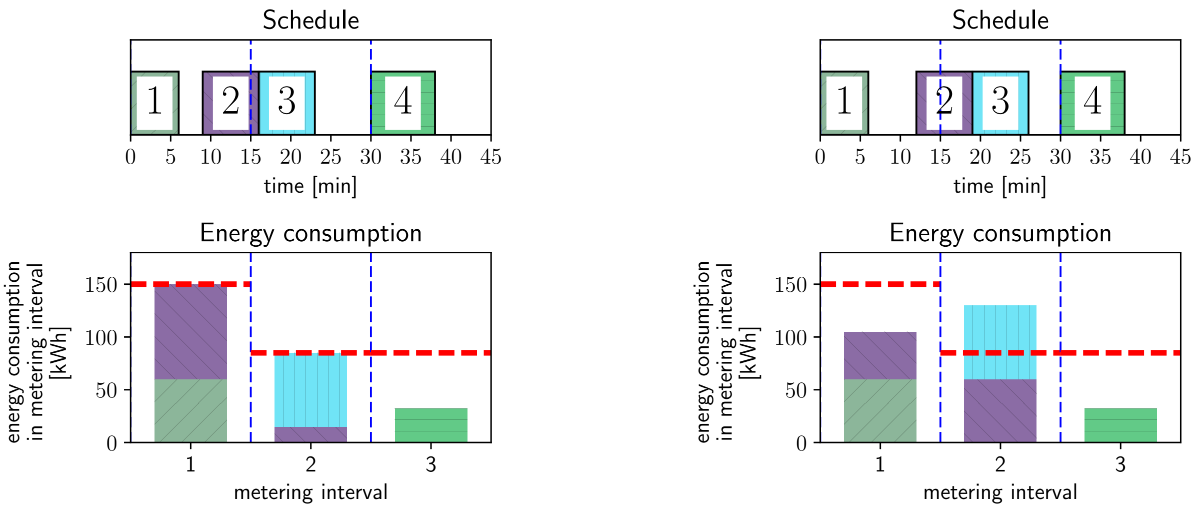
However, in reality unexpected events (see [1]) such as a machine breakdown or a delay caused by material unavailability may occur, which can cause a deviation of the baseline start times of the operations. It is possible that in such a *realised schedule* some energy demanding operations will get closer to each other in some metering interval and therefore increasing the total consumed energy in that metering interval above the energy limit, e.g. see Fig. 1b that shows a realised schedule in which operation 2 deviated from its baseline start time by 3 time units.

A possible way of dealing with the unexpected events is to monitor the manufacturing process by a scheduling system and when an operation deviates, the remaining ones are rescheduled so that the energy consumption limits are not violated in the new baseline schedule. For example, consider the schedule in Fig. 1b. The scheduling system detects that starting operation 3 at completion of deviated operation 2 would violate the energy consumption limit, therefore it should shift operation 3 further to the future so that part of it will be allocated in metering interval 3 thus decreasing the total energy consumption in metering interval 2. However, if the initial baseline schedule proposed by the scheduling system is not devised in a robust way, e.g. energy demanding operations are scheduled in close time proximity to each other, then even in case of small deviations a large idle times could be needed. Moreover, it is not always the case that the manufacturing process is controlled by some on-line scheduling system, i.e. the manufacturing process is only represented by one baseline schedule that workers receive at the start of their shift. By inserting small idle times and having a more suitable order of the operations, it is possible to make the baseline schedule *robust* against such disturbances, i.e. a robust baseline schedule guarantees that if the start times are arbitrary delayed within the given limits, then the energy consumption limits are not violated.

In this work we consider that the power consumption of the operations is not changing over time; incorporating nonlinear power consumption is a possible direction for future research.

A. Motivation

Motivation of the problem comes from a glass production company. This company has production orders that represent a final products from a glass panel. To complete an order, a glass panel goes through two stages: preprocessing and tempering. In the preprocessing stage, the glass panels are cut and drilled while in the tempering stage, the preprocessed glass panels



(a) Example of a feasible baseline schedule.

(b) Example of a realised schedule in which the energy consumption limit is violated in metering interval 2.

Figure 1: Example of a baseline and realised schedule. Energy consumption limits are denoted by dashed horizontal red lines.

are heated in a furnace up to 620 °C and then rapidly cooled down. The contract with the electricity provider requires that the energy consumption of the production in each 15 minutes metering interval is less than the given energy consumption limit.

Since the energy consumption of the preprocessing stage is negligible compared to the energy demand of the tempering stage, we consider scheduling of the operations only in the tempering stage. However, the preprocessing stage is not completely omitted because the completion time of the last step in the preprocessing stage represents the time when a preprocessed glass is available for tempering, i.e. it represents a *release time*.

Due to logistical reasons, groups of orders should be completed up to some due date. Therefore, our goal is to provide a robust baseline schedule for a single machine that minimises the sum of *tardiness*.

Although the problem is motivated by a specific real-world problem from the production domain, it is possible to apply the results to similar scheduling problems where availability limits on partially renewable resources can be represented by energy consumption limits.

B. Related Work

In general, the significance of energy-aware scheduling increases due to scarcity of energy resources [2]. The energy consumption limits have been previously studied in [3], [4], [5], [6]. In [3], an integer linear programming formulation was proposed which was improved in [4]. A two-stage approach combining integer linear programming and constraint programming was presented in [5]. A similar problem to energy consumption limits is the problem of a maximal instantaneous power consumption [7], [8]. However, start times deviations are not considered in any of these works.

Robust scheduling is a well-studied problem in the domain of project scheduling [9], [10]. The robustness is achieved either by a robust resource allocation or inserting time buffers between activities. However, the existing literature mostly focuses on stochastic optimisation of expected deviation of the realised start times from the baseline start times, whereas we are concerned with satysfing the energy limit in all possible discrete situations. This problem is known in the literature as *optimisation with uncertainty sets*, e.g. in [11] a logic-based Benders decomposition approach is employed in a scheduling problem with release time delays, which are modelled using uncertainty sets.

C. Contribution and Outline

Our contributions are the following: (i) a pseudo-polynomial algorithm for checking whether a given baseline schedule is robust and (ii) a relaxation of the problem and feasibility cuts. Both contributions are the foundations of a decomposition approach that we use for solving the proposed scheduling problem.

The paper is organised as follows. Section II states the problem in formal way. Section III describes the solution approach and which is then evaluated in Sect. IV. Section V discusses the integration of the proposed algorithm into production process. Finally, the last section concludes the paper.

II. Problem Statement

First, the scheduling problem considering only the energy consumption limits without robustness will be described. Then, the scheduling problem is extended with the deviation of the start times.

A. Non-robust Scheduling with Energy Consumption Limits

Let $\mathcal{J} = \{1, 2, \ldots, n\}$ be a *set of operations* that have to be scheduled on a single machine without preemption. For each *operation* $j \in \mathcal{J}$ we define *release time* $r_j \in \mathbb{N}$, *processing time* $p_j \in \mathbb{N}_{>0}$ and *due date* $d_j \in \mathbb{N}$. For each operation j we also define *power consumption* P_j that represents an instantaneous power consumption of the machine when processing operation j.

The operations have to be scheduled within *scheduling horizon* H, i.e. the operations must complete at most at time H. The scheduling horizon is divided into *set of metering intervals* $\Omega = \{1, 2, \ldots, \frac{H}{D}\}$ with equal *length* of $D \in \mathbb{N}_{>0}$ (it is assumed that H is a multiple of D). For each metering interval $\omega \in \Omega$, *maximum energy consumption* is denoted as $E_\omega^{\max}$, which represents the energy consumption limit on the total energy consumption in metering interval ω. Moreover, let us denote *start* and *end* of interval ω as $\omega_\text{s} = (\omega - 1) \cdot D$ and $\omega_\text{e} = \omega_\text{s} + D$, respectively.

Baseline start times $BS \in \mathbb{N}^n$ is a vector, where each element BS_j represents the baseline start time of operation j. Baseline start times BS are called *baseline schedule* if the operations are not overlapping and for all operations j holds that $r_j \leq BS_j \leq H - p_j$. If operation j starts at time BS_j in some schedule BS, then a *processing time in metering interval* ω is denoted as $p_{j,\omega}^{BS}$, which represents the length of the overlap between metering interval ω and the interval in which the operation j is scheduled if started at time BS_j. A *feasible baseline schedule* is a baseline schedule such that in each metering interval ω the total energy consumption is at most $E_\omega^{\max}$, i.e.

$$\sum_{j \in \mathcal{J}} p_{j,\omega}^{BS} \cdot P_j \leq E_\omega^{\max}, \ \ \omega \in \Omega \tag{1}$$

The goal is to find a feasible baseline schedule which minimises the sum of *tardiness* $\sum_{j \in \mathcal{J}} T_j$, where $T_j = \max\{0, (BS_j + p_j) - d_j\}$. We describe this problem in Graham's notation as $1|r_j, E_\omega^{\max}| \sum T_j$.

B. Robust Scheduling with Energy Consumption Limits

Let $\delta^{\max} \in \mathbb{N}$ be a *maximum deviation* of any operation. Then $\Delta = \{0, 1, \ldots, \delta^{\max}\}^n = \{\delta_1, \delta_2, \ldots, \delta_{|\{0,1,\ldots,\delta^{\max}\}^n|}\}$ is a *set of all deviation situations*. *Deviation situation* $\delta_i \in \Delta$ is a vector where each element $\delta_{i,j}$ represents a *deviation* of operation j in deviation situation δ_i.

Let BS be some baseline schedule, then $\pi \in \Pi(\mathcal{J})$ is a *permutation of operations* corresponding to the order of operations defined by schedule BS, where $\Pi(\mathcal{J})$ is a set of all permutations of $\mathcal{J}$; $\pi(k)$ represents the operation at k-th position in permutation π. Let δ_i be an arbitrary deviation situation, then we define the corresponding *realised schedule* RS_i as follows

$$RS_{i,\pi(1)} = BS_{\pi(1)} + \delta_{i,\pi(1)} \tag{2}$$

$$RS_{i,\pi(k)} = \max\{BS_{\pi(k)}, RS_{i,\pi(k-1)} + p_{\pi(k-1)}\} + \delta_{i,\pi(k)} \quad k \in \mathcal{J}, k \neq 1 \tag{3}$$

where $RS_{i,\pi(k)}$ is the realised start time of operation $\pi(k)$ in realised schedule RS_i.

Eq. (2) and (3) state that an operation must start as early as possible but not before its baseline start time and not before the end of the previous operation. Therefore, it represents a *reactive policy* [9] that is very simple to follow by the workers.

For the purpose of Sect. III we also define a *latest start time* of operation $\pi(k)$ as $LS_{\pi(k)} = RS_{i,\pi(k)}$, where $\delta_i = (\delta^{\max}, \delta^{\max}, \ldots, \delta^{\max})$. The latest start time represents the maximum starting time over all realised schedules for a fixed baseline schedule.

The goal is to find feasible baseline schedule BS such that the energy consumption limits are not violated in any metering interval for any deviation situation, i.e.

$$\sum_{j \in \mathcal{J}} p_{j,\omega}^{RS_i} \cdot P_j \leq E_\omega^{\max}, \ \ \omega \in \Omega, \delta_i \in \Delta \tag{4}$$

and for which $\sum_{j \in \mathcal{J}} T_j$ is minimal. Such baseline schedule is called *robust*. We describe this problem in Graham's notation as $1|r_j, E_\omega^{\max}, \delta_j^{\max} = \delta^{\max}| \sum T_j$.

We illustrate the notation on a simple example with 5 operations $\mathcal{J} = \{1, 2, 3, 4, 5\}$. Let $D = 15$, $\Omega = \{1, 2\}$, $\delta^{\max} = 3$ and $E_1^{\max} = E_2^{\max} = 120$. The values of the operations are provided in Tab. I. A feasible baseline schedule is shown in Tab. II and a realised schedule computed using Eq. (2) and (3) for deviation situation $\delta_i = (3, 0, 3, 2, 0)$ is provided in Tab. III. Notice that $RS_{i,4} - BS_4 = 5 > \delta^{\max}$ since operation 4 is shifted by operation 3. Moreover, $RS_{i,5} - BS_5 > 0$ even though $\delta_{i,5} = 0$; this is due to deviation of the preceding operations. The total tardiness equals to 4 and the total energy consumption in the realised schedule in metering intervals 1 and 2 are 69 and 117, respectively. The visualisation of the baseline and realised start times are in Fig. 2.

Problem $1|r_j, E_\omega^{\max}, \delta_j^{\max} = \delta^{\max}| \sum T_j$ is $\mathcal{NP}$-hard which can be shown by a reduction from problem $1||\sum_j T_j$ which is also $\mathcal{NP}$-hard [12]. Problem $1|r_j, E_\omega^{\max}, \delta_j^{\max} = \delta^{\max}| \sum T_j$ solves any instance of problem $1||\sum_j T_j$ if $D = \sum_{j \in \mathcal{J}} p_j$, $\Omega = \{1\}$, $E_1^{\max} = \sum_{j \in \mathcal{J}} p_j \cdot P_j$, $\delta^{\max} = 0$ and $\forall j \in \mathcal{J} : r_j = 0$.

III. Solution Approach

First, we formulated the problem as a monolithic *Mixed Integer Linear Programming* (MILP) model. The underlying concept is that for each deviation situation, an independent realised schedule is constructed which is checked against the energy consumption limits. The main issue with the monolithic model is that due to size of Δ, the number of constraints grows exponentially in the number of the operations.

Therefore, a decomposition scheme is used in which the original problem is split into a *master problem* and a *subproblem*. The master problem solves optimally a relaxed problem of the original problem. Let BS' be an optimal solution to the master problem. The subproblem checks whether schedule

Table I
Values of the operations.

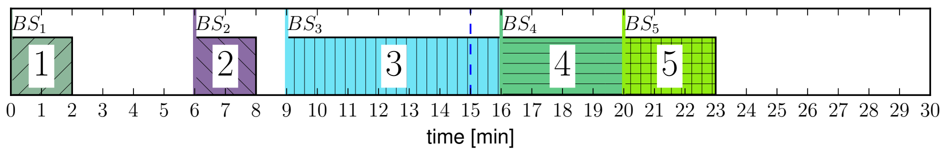
j	r_j	d_j	p_j	P_j
1	0	5	2	5
2	0	10	2	7
3	0	15	7	15
4	0	17	4	12
5	0	30	3	3

Table II
Baseline schedule.

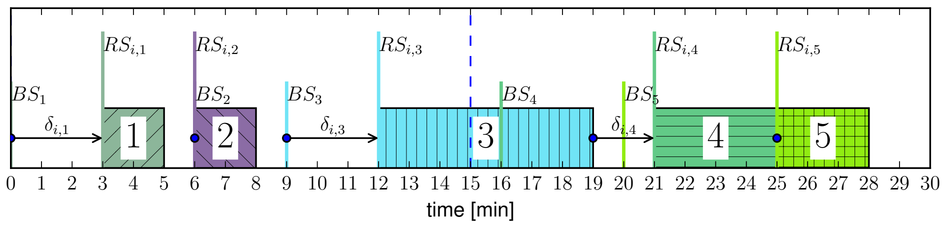
j	BS_j
1	0
2	6
3	9
4	16
5	20

Table III
Realised schedule for $\delta_i = (3, 0, 3, 2, 0)$.

j	$RS_{i,j}$	$p_{j,1}^{RS_i}$	$p_{j,2}^{RS_i}$	$p_{j,1}^{RS_i} \cdot P_j$	$p_{j,2}^{RS_i} \cdot P_j$
1	3	2	0	10	0
2	6	2	0	14	0
3	12	3	4	45	60
4	21	0	4	0	48
5	25	0	3	0	9



(a) Baseline schedule.



(b) Realised schedule for $\delta_i = (3, 0, 3, 2, 0)$.

Figure 2: Visualisation of the example.

BS' is feasible in the original problem. If yes, then since the master problem is a relaxation of the original problem, schedule BS' is optimal in the original problem. If no, the subproblem generates *cutting constraints* based on schedule BS' which are then added to the master problem. The master problem is then solved again and the whole process repeats unless the master problem is infeasible or the subproblem proves that the current schedule is feasible in the original problem. The decomposition is a special case of *logic-based Benders decomposition* [13], in which *bounding functions* only determine, whether the solution is feasible or not.

The key aspects to identify are how the original problem should be relaxed in the master problem and what cutting constraints should be generated.

A. Master Problem

Essentially, the master problem is $1|r_j, E_\omega^{\max}|\sum T_j$ which is modelled as MILP using time-indexed formulation with additional constraints for deviations caused by a single operation. We use the time-indexed formulation because it is more suitable for generating the cutting constraints.

There are three types of variables in the master problem: (i) a *binary baseline start time* of operation j in time t defined as $bs_{j,t} = 1$ iff j starts at t, (ii) an *energy consumed in time* t denoted as E_t^{time}, and (iii) tardiness variables T_j. To ensure that no operation could complete outside of the horizon even if all operations deviated, we define *maximum start time* $BS^{\max} = H - (n \cdot \delta^{\max} + \max_{j \in \mathcal{J}} p_j)$.

The MILP model of the master problem is in Fig. 3. The objective of the master problem is minimisation of the sum of tardiness. Constraint (6) ensures that each operation starts in some time that is at least its release time and at most the maximum start time. Constraint (7) enforces that each time unit can be occupied by at most one operation. Computation of consumed energy in time t is in Constraint (8). Constraint (9) ensures that the energy consumption limit in each metering interval is not violated.

The master model in Fig. 3 does not take the deviations of the start times into account. However, we can strengthen the model by adding the following constraint

$$\sum_{t=\max\{0,\omega_s-\delta\}}^{\omega_e-1-\delta} E_t^{\text{time}} \leq E_\omega^{\max}, \quad \forall \delta \in \{1, \dots, \delta^{\max}\}, \forall \omega \in \Omega \tag{12}$$

that considers deviations of a single operation. The idea is the following: assume that j is the first operation that has non-zero intersection with interval $[\omega_s - \delta, \omega_s)$ in baseline schedule BS (see Fig. 4a). Let RS_i be a realised schedule corresponding to deviation situation δ_i defined as $\delta_{i,j'} = \delta$ if $j' = j$, $\delta_{i,j'} = 0$ otherwise (see Fig. 4b). All parts of the operations that are allocated in time interval $[\omega_s - \delta, \omega_e - \delta)$ in baseline schedule BS (the operation parts are enclosed by the hatched grey box in Fig. 4a) will be right-shifted into interval $[\omega_s, \omega_e)$ in realised schedule RS_i. The constraint is also valid when there are idle times between the operations, although the constraint is weaker in a such case.

$$\min \sum_{j \in \mathcal{J}} T_j \tag{5}$$

$$\text{s.t.} \sum_{t=r_j}^{BS^{\max}} bs_{j,t} = 1, \ \ \forall j \in \mathcal{J} \tag{6}$$

$$\sum_{j \in \mathcal{J}} \sum_{t'=\max\{0, t-p_j+1\}}^{t} bs_{j,t'} \leq 1, \ \ \forall t \in \{0, \ldots, H-1\} \tag{7}$$

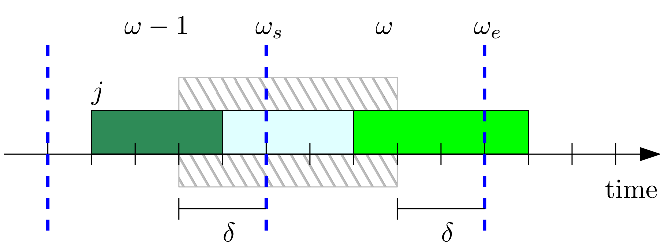
$$\sum_{j \in \mathcal{J}} \sum_{t'=\max\{0, t-p_j+1\}}^{t} bs_{j,t'} \cdot P_j = E_t^{\text{time}}, \ \ \forall t \in \{0, \ldots, H-1\} \tag{8}$$

$$\sum_{t=\omega_{\text{s}}}^{\omega_{\text{e}}-1} E_t^{\text{time}} \leq E_\omega^{\max}, \ \ \forall \omega \in \Omega \tag{9}$$

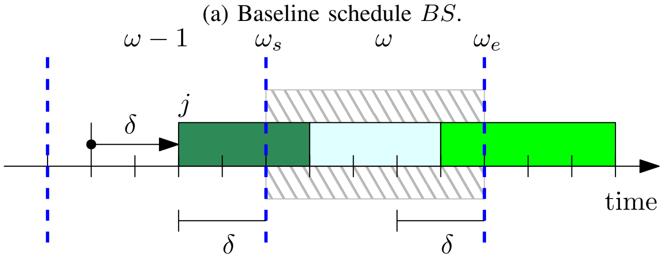
$$T_j \geq \left(p_j + \sum_{t=0}^{BS^{\max}} t \cdot bs_{j,t} \right) - d_j, \ \ \forall j \in \mathcal{J} \tag{10}$$

$$T_j \geq 0, \forall j \in \mathcal{J} \ \ . \tag{11}$$

Figure 3: MILP formulation of the master model.



(a) Baseline schedule BS.



(b) Realised schedule RS_i.

Figure 4: Illustration of the concept behind Constraint (12), $\delta = 2$, $D = 5$.

B. Subproblem

1) Feasibility check: In this section we provide a pseudo-polynomial algorithm that is able to detect whether baseline schedule BS found by the master problem is not violating the energy limit in any metering interval for any deviation situation.

The following theorem shows that whenever a baseline schedule violates the energy consumption limit in some deviation situation δ_{i_1}, there exists deviation situation δ_{i_2} that has a specific form and in which the energy consumption limit is also violated (if δ_{i_1} has this specific form then $\delta_{i_1} = \delta_{i_2}$). Therefore, the main idea behind the algorithm is that only the deviation situations of the specific form are checked.

Theorem 1. *Let BS be the baseline schedule and π be a corresponding permutation of operations. Let δ_{i_1} be a deviation situation in which the corresponding realised schedule RS_{i_1} violates the energy consumption limit in some metering interval ω. Then there exists deviation situation δ_{i_2} and $k^{arb} \in \mathcal{J}$ defined as*

1) $\forall k \in [1, k^{arb} - 1] : \delta_{i_2, \pi(k)} = \delta^{max}$
2) $0 \leq \delta_{i_2, \pi(k^{arb})} \leq \delta^{max}$
3) $\forall k \in [k^{arb} + 1, n] : \delta_{i_2, \pi(k)} = 0$

such that realised schedule RS_{i_2} also violates the energy consumption limit in metering interval ω.

Proof. Let k^{first} be the first operation that has non-zero intersection with metering interval ω in RS_{i_1}. For given permutation π we define function $\varphi : \mathcal{J} \times \{0, \ldots, \delta^{\max}\} \rightarrow \{1, 2, \ldots, |\Delta|\}$ such that

$$\delta_{\varphi(k,\delta), \pi(k')} = \begin{cases} \delta^{\max} & k' \in [1, k-1] \\ \delta & k' = k \\ 0 & k' \in [k+1, n] \end{cases} \ . \tag{13}$$

For any $k \in [1, k^{\text{first}}]$ and $0 \leq \delta < \delta^{\max}$ we prove the following properties

(P1) $RS_{\varphi(1,0), \pi(k^{\text{first}})} \leq RS_{i_1, \pi(k^{\text{first}})}$: holds from

$$BS_{\pi(k^{\text{first}})} \leq RS_{i_1, \pi(k^{\text{first}})} \tag{14}$$

and

$$RS_{\varphi(1,0), \pi(k^{\text{first}})} = BS_{\pi(k^{\text{first}})} \ \ . \tag{15}$$

(P2) $RS_{\varphi(k^{\text{first}}, \delta^{\max}), \pi(k^{\text{first}})} \geq RS_{i_1, \pi(k^{\text{first}})}$: holds from

$$LS_{\pi(k^{\text{first}})} \geq RS_{i_1, \pi(k^{\text{first}})} \tag{16}$$

and

$$RS_{\varphi(k^{\text{first}},\delta^{\max}),\pi(k^{\text{first}})} = LS_{\pi(k^{\text{first}})} \ . \tag{17}$$

(P3) $RS_{\varphi(k,\delta),\pi(k^{\text{first}})} \leq RS_{\varphi(k,\delta+1),\pi(k^{\text{first}})}$: Let $k' \in [k+1, k^{\text{first}}]$ be a first position such that

$$RS_{\varphi(k,\delta),\pi(k')} > RS_{\varphi(k,\delta+1),\pi(k')} \ ; \tag{18}$$

obviously this cannot happen for $k' = k$. From the assumption it holds that

$$RS_{\varphi(k,\delta),\pi(k'-1)} \leq RS_{\varphi(k,\delta+1),\pi(k'-1)} \ , \tag{19}$$

therefore we get a contradiction

$$\begin{aligned} &RS_{\varphi(k,\delta),\pi(k')} = \\ &= \max\{BS_{\pi(k')}, RS_{\varphi(k,\delta),\pi(k'-1)} + p_{\pi(k'-1)}\} \\ &\leq \max\{BS_{\pi(k')}, RS_{\varphi(k,\delta+1),\pi(k'-1)} + p_{\pi(k'-1)}\} \\ &= RS_{\varphi(k,\delta+1),\pi(k')} \ . \end{aligned} \tag{20}$$

(P4) $RS_{\varphi(k,\delta+1),\pi(k^{\text{first}})} \leq RS_{\varphi(k,\delta),\pi(k^{\text{first}})} + 1$: Let $k' \in [k+1, k^{\text{first}}]$ be a first position such that

$$RS_{\varphi(k,\delta+1),\pi(k')} > RS_{\varphi(k,\delta),\pi(k')} + 1 \ ; \tag{21}$$

obviously this cannot happen for $k' = k$. From the assumption it holds that

$$RS_{\varphi(k,\delta+1),\pi(k'-1)} \leq RS_{\varphi(k,\delta),\pi(k'-1)} + 1 \ , \tag{22}$$

therefore we get a contradiction

$$\begin{aligned} &RS_{\varphi(k,\delta+1),\pi(k')} = \\ &= \max\{BS_{\pi(k')}, RS_{\varphi(k,\delta+1),\pi(k'-1)} + p_{\pi(k'-1)}\} \\ &\leq \max\{BS_{\pi(k')}, RS_{\varphi(k,\delta),\pi(k'-1)} + 1 + p_{\pi(k'-1)}\} \\ &\leq \max\{BS_{\pi(k')}, RS_{\varphi(k,\delta),\pi(k'-1)} + p_{\pi(k'-1)}\} + 1 \\ &= RS_{\varphi(k,\delta),\pi(k')} + 1 \ . \end{aligned} \tag{23}$$

The following property holds only for $k \in [1, k^{\text{first}} - 1]$

(P5) $RS_{\varphi(k,\delta^{\max}),\pi(k^{\text{first}})} \leq RS_{\varphi(k+1,1),\pi(k^{\text{first}})} \leq RS_{\varphi(k,\delta^{\max}),\pi(k^{\text{first}})} + 1$: since

$$RS_{\varphi(k,\delta^{\max}),\pi(k^{\text{first}})} = RS_{\varphi(k+1,0),\pi(k^{\text{first}})} \ , \tag{24}$$

this was already proven.

Construct a sequence of pairs (k, δ)

$$(1,0),(1,1),(1,2)\ldots,(1,\delta^{\max}),(2,1),(2,2),\ldots,(k^{\text{first}},\delta^{\max}) \tag{25}$$

Properties (P3), (P4) and (P5) ensure that if $(k,\delta),(k',\delta')$ are two consecutive pairs, then

$$RS_{\varphi(k,\delta),\pi(k^{\text{first}})} \leq RS_{\varphi(k',\delta'),\pi(k^{\text{first}})} \leq RS_{\varphi(k,\delta),\pi(k^{\text{first}})} + 1 \ . \tag{26}$$

Since $RS_{\varphi(1,0),\pi(k^{\text{first}})} \leq RS_{i_1,\pi(k^{\text{first}})}$ and $RS_{\varphi(k^{\text{first}},\delta^{\max}),\pi(k^{\text{first}})} \geq RS_{i_1,\pi(k^{\text{first}})}$, there must be a pair $(k^{\text{arb}}, \delta^{\text{arb}})$ such that $RS_{\varphi(k^{\text{arb}},\delta^{\text{arb}}),\pi(k^{\text{first}})} = RS_{i_1,\pi(k^{\text{first}})}$. Therefore we set $\delta_{i_2} = \delta_{\varphi(k^{\text{arb}},\delta^{\text{arb}})}$. This implies that the energy consumption of operation $\pi(k^{\text{first}})$ in metering interval ω is the same in both schedules RS_{i_1},RS_{i_2}.

1: **procedure** CheckBaselineSchedule(BS)
2: $(k^{\text{arb}}, \delta^{\text{arb}}) \leftarrow (1, 0)$
3: **while** $k^{\text{arb}} \leq n \ \wedge \ \delta^{\text{arb}} \leq \delta^{\max}$ **do**
4: $RS \leftarrow createRealisedSchedule(BS, k^{\text{arb}}, \delta^{\text{arb}})$
5: **for all** $\omega \in \Omega$ **do**
6: **if** $energyLimitViolated(RS, \omega)$ **then**
7: **return** INFEASIBLE
8: **end if**
9: **end for**
10: **if** $\delta^{\text{arb}} < \delta^{\max}$ **then**
11: $\delta^{\text{arb}} \leftarrow \delta^{\text{arb}} + 1$
12: **else**
13: $(k^{\text{arb}}, \delta^{\text{arb}}) \leftarrow (k^{\text{arb}} + 1, 1)$
14: **end if**
15: **end while**
16: **return** FEASIBLE
17: **end procedure**

Figure 5: Algorithm for checking whether a baseline schedule violates the energy limits in any interval.

Now it remains to show that the energy consumption in metering interval ω of operations on positions $[k^{\text{first}} + 1, n]$ in RS_{i_2} is not less than the energy consumption in RS_{i_1}. A key observation is that shifting any operation on position $k \in [k^{\text{first}} + 1, n]$ to the left up to ω_{s} will not decrease the total energy consumption in metering interval ω. We show that $\forall k \in [k^{\text{first}}, n] : RS_{i_1,\pi(k)} \geq RS_{i_2,\pi(k)}$ holds. Assume that $k \in [k^{\text{first}} + 1, n]$ is the first position for which $RS_{i_1,\pi(k)} < RS_{i_2,\pi(k)}$. Then from assumption $RS_{i_1,\pi(k-1)} \geq RS_{i_2,\pi(k-1)}$ we get a contradiction

$$\begin{aligned} RS_{i_2,\pi(k)} &= \max\{BS_{\pi(k)}, RS_{i_2,\pi(k-1)} + p_{\pi(k-1)}\} \\ &\leq \max\{BS_{\pi(k)}, RS_{i_1,\pi(k-1)} + p_{\pi(k-1)}\} \\ &= RS_{i_1,\pi(k)} \ . \end{aligned} \tag{27}$$

Since $RS_{i_2,\pi(k^{\text{first}})} = RS_{i_1,\pi(k^{\text{first}})}$ and $\forall k \in [k^{\text{first}} + 1, n]$

$$\omega_{\text{s}} \leq RS_{i_2,\pi(k^{\text{first}})} + p_{\pi(k^{\text{first}})} \leq RS_{i_2,\pi(k)} \leq RS_{i_1,\pi(k)} \tag{28}$$

holds, the energy consumption in metering interval ω in realised schedule RS_{i_2} is not less than the energy consumption in RS_{i_1}. $\square$

The algorithm for checking the baseline schedule is shown in Fig. 5. The algorithm uses the idea from proof of Theorem 1 by iteratively trying all pairs from (25). From pair $(k^{\text{arb}}, \delta^{\text{arb}})$, the realised schedule for the deviation situation defined as (13) is created using function $createRealisedSchedule$. The energy consumption limits for each metering interval in realised schedule RS are checked using function $energyLimitViolated$ at the start of each iteration.

Since in the worst case the number of iterations of the loop in lines 3-15 is $n \cdot \delta^{\max}$ (this can be easily derived from the space of the explored deviation situations corresponding to the pairs (25)), the complexity of the algorithm is $\mathcal{O}(n^2 \cdot \delta^{\max} \cdot (|\Omega| + 1))$.

2) Cutting constraints: For the decomposition approach to remain exact, the only requirement on the cutting constraints generated from the subproblem is that if BS^* is an optimal schedule for the original problem, no cutting constraint is violated in BS^*.

In our subproblem, 2 types of cutting constraints are generated. The first type of cutting constraint is a *lower bound* cut. If Z' is the objective value of relaxed solution BS' that is infeasible in the original problem, then the cut has a form $Z' \leq \sum_{j \in \mathcal{J}} T_j$. The idea is to speed up the master problem by providing lower bound Z' on the original problem.

The second type of cutting constraint is a *partial solution* cut. Let RS'_i be a realised schedule corresponding to deviation situation δ_i in which the energy consumption limit is violated in some metering interval ω. Then BS' can be cut from the master problem using the following cutting constraint

$$\sum_{j \in \mathcal{J}} \sum_{t \in \{0,\dots,H-1\}: bs'_{j,t}=1} bs_{j,t} \leq n-1 \ , \tag{29}$$

which enforces that at least one operation starts at different time than in BS'. This cut can be strengthened by observing that not all operations are responsible for violating the energy limit. Let $\pi'(k^{\text{first}})$ and $\pi'(k^{\text{last}})$ be the first and the last operation that has non-zero intersection with interval ω in RS'_i, respectively. Furthermore, let $k^{\text{head}} \leq k^{\text{first}}$ be a maximum position such that $LS'_{\pi'(k^{\text{head}}-1)} + p_{\pi'(k^{\text{head}}-1)} \leq BS'_{\pi'(k^{\text{head}})}$. If no such a position exists, then $k^{\text{head}} = 1$. The meaning of k^{head} is that even if all operations on positions $k < k^{\text{head}}$ deviate by $\delta^{\max}$, the earliest start time of $\pi'(k^{\text{head}})$ will be the same as the baseline start time, i.e. the operation will not be shifted due to deviations of the previous operations. It can be shown that only operations $[\pi'(k^{\text{head}}), \pi'(k^{\text{head}}+1), \dots, \pi'(k^{\text{last}})]$ are responsible for violating the energy limit; either directly by intersecting metering interval ω or indirectly by shifting the subsequent operations. Therefore, the partial solution cut cuts out solutions in which operations $[\pi'(k^{\text{head}}), \pi'(k^{\text{head}}+1), \dots, \pi'(k^{\text{last}})]$ (i) have the same order as in π', (ii) all other operations start after $\pi'(k^{\text{last}})$ or before $\pi'(k^{\text{head}})$, and (iii) the length of intersection of the operations $[\pi'(k^{\text{head}}), \pi'(k^{\text{head}}+1), \dots, \pi'(k^{\text{last}})]$ with metering interval ω in some realised schedule is at least the length of intersection in RS'_i. Therefore, the partial solution cut can be written as

$$\left(\sum_{k \in (\mathcal{J} \setminus \{k^{\text{head}},\dots,k^{\text{last}}\})} \sum_{t=0}^{BS'_{\pi'(k^{\text{head}})}} bs_{\pi'(k),t} + \sum_{t=\bar{t}}^{BS^{\max}} bs_{\pi'(k),t} \right) + \sum_{k=k^{\text{first}}}^{k^{\text{last}}-1} \sum_{t=BS'_{\pi'(k)}}^{BS'_{\pi'(k+1)}+p_{\pi'(k+1)}-1} bs_{\pi'(k),t} + \sum_{t=BS'_{\pi'(k^{\text{last}})}}^{\bar{t}} bs_{\pi'(k^{\text{last}}),t} \leq n-1 \tag{30}$$

where $\bar{t} = \min\{RS'_{i,\pi'(k^{\text{last}})}, BS'_{\pi'(k^{\text{last}}+1)}\}$ if $k^{\text{last}}+1 \leq n$, $\bar{t} = RS'_{i,\pi'(k^{\text{last}})}$ otherwise.

IV. Experiments

The proposed decomposition approach was evaluated on randomly generated instances inspired by a real-world scenario from a glass production company. The number of operations was fixed to 15 and the length of metering interval was fixed to 15. Parameters of each operation j were sampled as following: p_j from discrete uniform distribution $\mathcal{U}\{3,7\}$, P_j from continuous uniform distribution $\mathcal{U}(10,15)$, r_j from $\mathcal{U}\{0, 0.5 \cdot \sum p_j\}$ and for generating d_j value of $d_j - (r_j + p_j)$ from $\mathcal{U}\{0, 0.75 \cdot \sum p_j\}$. Energy consumption limit $E^{\max}_\omega$ was set to $D \cdot \gamma \cdot \max_{j \in \mathcal{J}} P_j$, where γ is an *energy limit tightness* (for simplicity, one value of the energy limit is generated for all metering intervals).

We experimented with various values of $\delta^{\max}$ and γ. For each values $\delta^{\max}$ and γ, 100 instances were generated. The results of the experiment are in Tab. IV. Column $\%$ represents the percentage of the instances solved optimally within the time limit of 10 minutes. Columns $\min(\text{RT}), \text{ave}(\text{RT}), \max(\text{RT})$ represents the minimum, average and maximum solver runtime on the solved instances, respectively. Similarly, $\min(\text{IT}), \text{ave}(\text{IT}), \max(\text{IT})$ represents the minimum, average and maximum number of iterations between master problem and subproblem on the solved instances, respectively. The experiments were performed on Intel Xeon E5-2620 v2 @ 2.10 GHz processor with 64 GB of RAM. Gurobi Optimizer 6.5 was used for solving the master problem while the rest of the algorithms was programmed in C++ and compiled with GCC 4.9.3.

From the results we can see that lower value of energy limit tightness increases the complexity of the instances, while the effect of increasing the maximum deviation is less significant. However, decreasing the energy limit tightness considerably is not practical since it increases the total tardiness.

We also measured the relative execution time of the master problem and the feasibility check algorithm. In the experiment with $\gamma = 0.9$ and $\delta^{\max} = 3$, 99.4% of the total experiment execution time was spent in solving the master problem while 0.3% of the time was spent in the feasibility check algorithm.

V. Integration into Production Process - Scaling to Larger Instances

Although the number of operations in the experiments may seem to be low for the real-world scenarios, our approach can be easily scaled to larger instances using a *rolling-horizon* heuristic scheduling. First, sort the operations increasingly according to their due dates. Then, select the first 10-15 operations, solve the smaller scheduling problem with the selected operations and execute the proposed baseline schedule. After the last operation of this schedule is finished, select the next 10-15 operations and perform the same steps as above. The justification of such method is that due to possible disturbances, it could be more advantageous to perform rescheduling every k hours, therefore the advantage of solving a larger instance at once is virtually lost. Moreover, the rolling-horizon

Table IV: Experiment results

γ	$\delta^{\max}$	%	min(RT) [s]	ave(RT) [s]	max(RT) [s]	min(IT)	ave(IT)	max(IT)
0.9	1	78	0.0	13.14	360	1	11.86	216
0.9	2	66	0.0	30.82	592	1	24.26	191
0.9	3	63	0.0	34.98	558	1	28.10	299
0.88	1	56	0.0	57.12	555	1	37.98	353
0.88	2	39	0.0	28.82	255	1	25.79	236
0.88	3	37	0.0	46.76	577	1	29.24	212

approach can be easily employed in online setting where the production orders are arriving progressively through time.

VI. Conclusion

We studied a novel problem of satisfying the energy consumption limits by manufacturing companies with high energy demand under production uncertainties. Such problem often occurs in reality since the realised schedule is usually different from the proposed baseline schedule.

For solving this problem we designed a decomposition approach, in which our main contributions are (i) a pseudo-polynomial algorithm for checking whether the given baseline schedule is robust considering the uncertainties and (ii) feasibility cuts. In the experiments we showed that our approach is able to solve optimally majority of instances with 15 operations if the the energy consumption limit is not very tight. For the real-world scenarios with substantially more operations, our decomposition method can be integrated into a rolling-horizon scheduling in which the production is iteratively planned for shorter time periods.

Acknowledgment

This work was supported by the Grant Agency of the Czech Republic under the Project GACR FOREST P103-16-23509S.

References

[1] H. Aytug, M. A. Lawley, K. McKay, S. Mohan, and R. Uzsoy, "Executing production schedules in the face of uncertainties: A review and some future directions," *European Journal of Operational Research*, vol. 161, no. 1, pp. 86–110, 2005.

[2] S. A. Mansouri, E. Aktas, and U. Besikci, "Green scheduling of a two-machine flowshop: Trade-off between makespan and energy consumption," *European Journal of Operational Research*, vol. 248, no. 3, pp. 772–788, 2016.

[3] K. Nolde and M. Morari, "Electrical load tracking scheduling of a steel plant," *Computers & Chemical Engineering*, vol. 34, no. 11, pp. 1899–1903, 2010.

[4] A. Haït and C. Artigues, "On electrical load tracking scheduling for a steel plant," *Computers & Chemical Engineering*, vol. 35, no. 12, pp. 3044–3047, 2011.

[5] A. Haït and C. Artigues, "A hybrid CP/MILP method for scheduling with energy costs," *European Journal of Industrial Engineering*, vol. 5, no. 4, pp. 471–489, 2011.

[6] H. Hadera, I. Harjunkoski, G. Sand, I. E. Grossmann, and S. Engell, "Optimization of steel production scheduling with complex time-sensitive electricity cost," *Computers & Chemical Engineering*, vol. 76, pp. 117–136, 2015.

[7] K. Fang, N. A. Uhan, F. Zhao, and J. W. Sutherland, "Flow shop scheduling with peak power consumption constraints," *Annals of Operations Research*, vol. 206, no. 1, pp. 115–145, 2013.

[8] A. Bruzzone, D. Anghinolfi, M. Paolucci, and F. Tonelli, "Energy-aware scheduling for improving manufacturing process sustainability: A mathematical model for flexible flow shops," *CIRP Annals - Manufacturing Technology*, vol. 61, no. 1, pp. 459–462, 2012.

[9] W. Herroelen, "Generating robust project baseline schedules," in *Tutorials in Operations Research: OR Tools and Applications : Glimpses of Future Technologies*, ser. TutORials in Operations Research. Institute for Operations Research and the Management Sciences (INFORMS), 2007, ch. 8, pp. 124–144.

[10] O. Lambrechts, E. Demeulemeester, and W. Herroelen, "Proactive and reactive strategies for resource-constrained project scheduling with uncertain resource availabilities," *Journal of Scheduling*, vol. 11, no. 2, pp. 121–136, 2007.

[11] E. Coban, A. Heching, J. N. Hooker, and A. Scheller-Wolf, "Robust scheduling with logic-based Benders decomposition," in *Operations Research Proceedings (OR 2014)*, M. Lübbecke, A. Koster, P. Letmathe, R. Madlener, B. Peis, and G. Walther, Eds. Springer International Publishing, 2016, pp. 99–105.

[12] J. Du and J. Y. Leung, "Minimizing total tardiness on one machine is NP-hard," *Mathematics of Operations Research*, vol. 15, no. 3, pp. 483–495, 1990.

[13] J. N. Hooker, "Planning and scheduling by logic-based Benders decomposition," *Operations Research*, vol. 55, no. 3, pp. 588–602, 2007.