Integer and Constraint Programming for Batch Annealing Process Planning

Willem-Jan van Hoeve and Sridhar Tayur

Tepper School of Business, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213, USA
vanhoeve@andrew.cmu.edu, stayur@andrew.cmu.edu

Abstract. We describe an optimization application in the context of steel manufacturing, to design and schedule batches for annealing furnaces. Our solution approach uses a two-phase decomposition. The first phase groups together orders into batches using a mixed-integer linear programming model. The second phase assigns the batches to furnaces and schedules them over time, using constraint programming. Our solution has been developed for operational use in two plants of a steel manufacturer in North America.

1 Introduction

We present an application of optimization technology for a steel manufacturer in North America, that operates two plants with annealing capability. Annealing is used in the steel industry as a heat treatment to modify the structure of the metals. For example, it may remove stresses, soften the steel, or refine the grain structure. In our case, the annealing process is performed in box furnaces, which can hold a specific number of steel coils. The furnaces are a primary (bottleneck) resource of the plants, which means that related operations are scheduled subject to the furnace annealing schedules. Since the existing approach for creating and scheduling batches for the box furnaces requires substantial manual interaction, the purpose of our project was to automate and improve this process.

Optimization models for batch design and scheduling in steel plants have been proposed before. For example, in [6] a mixed-integer programming (MIP) model is proposed for annealing batch scheduling in a general industrial setting, with fixed batches. Their allocation and scheduling of batches to furnaces is similar to our setting, but the authors consider different resources such as crane movements as well. In [4] a genetic algorithm to this problem is proposed. An excellent survey of batch scheduling is presented in [5].

In [3], a decomposition approach is proposed for a more general steel production problem (not just annealing). Conceptually, we follow a similar approach. A main difference is that we utilize constraint programming (CP) for the scheduling of batches, whereas [3] uses MIP. Furthermore, because we work with a more specific application, we can streamline our batch design MIP model using additional constraints. Lastly, [2] combines the batch composition and scheduling,

O^L	set of locked orders
O^W	set of work orders
O^P	set of planned orders
O	set of all orders $O^L \cup O^W \cup O^P$
A	set of anneal cycles
F	set of furnaces
H	integer time horizon (in minutes)

Table 1. Main sets for the problem.

$c_i \in A$	anneal cycle of $i \in O$
$w_i \in \mathbb{R}^+$	the weight of one coil in $i \in O$
$n_i \in [K]$	the number of coils in $i \in O$
$r_i \in H$	release date of $i \in O$
$d_i \in H$	due date of $i \in O$
$g_i \subseteq F$	furnace group of $i \in O^L$
$\Pi \subset O \times O$	set of precedences (i, j)

Table 2. Characteristics of the orders.

as we do in this work. Their batch design is restricted to the size of the batch, however, and does not group together different orders for example.

In summary, the main novelty of our approach is the combination of a rich batch design problem (solved with MIP) with a batch scheduling problem (solved with CP). Decomposing the problem into two parts (MIP for batch design and CP for scheduling) was crucial to make the approach scalable: Our two-phase approach scales to problems with (at least) 22 furnaces and 600 orders, creating a detailed schedule for about 7 days, within 15 to 30 minutes of computing time.

2 Problem Description

The input to our problem is a set of annealing orders O. Each order consists of a number of steel coils that need to be annealed using a specific recipe (anneal cycle) in box furnaces. The set of all anneal cycles is denoted by A. The set of all furnaces is denoted by F. The orders are grouped together in batches of a fixed maximum size, depending on the furnace capacity. For our application, the furnace capacity differs per plant, but the furnaces have uniform capacity K (total number of coils that can be loaded per batch) for a given plant.

After the batches have been created, they need to be allocated and scheduled on the available furnaces, given a discrete time horizon H. Table 1 summarizes the main sets of our problem, including a partition of the orders in three types:

- Locked orders (denoted by the set O^L) have been committed in the previous planning phase. They have been grouped together in a batch, and assigned to a furnace group for execution. These batches cannot be changed and must be scheduled as soon as possible.
- Work orders (denoted by the set O^W) are partially committed in the previous planning phase. They have a fixed number of coils that cannot be changed, but their batches have not yet been decided.
- Planned orders (denoted by the set O^P) are not yet committed, but are available to be scheduled. The given number of coils of a planned order may be reduced to complete the size of a batch, but not split into separate orders.

Each order has a number of characteristics, as presented in Table 2.[1] For each order, the number of coils (n_i) is at most K. Also, for some orders the due date d_i

[1] We use the common shorthand $[n]$ to denote the set $\{1, \dots, n\}$ for an integer n.

$p_{a,f} \in \mathbb{N}$	processing time of anneal cycle $a \in A$ on furnace $f \in F$ (in minutes)
$t_{a,a',f} \in \mathbb{N}$	sequence-dependent switchover time from anneal cycle a to a' on furnace f (in minutes)
$C_{a,a'} \in \{0,1\}$	whether $a \in A$ and $a' \in A$ are compatible and can be combined
$\Gamma_{a,a'} \in \{a,a'\}$	anneal cycle that determines the processing time of the combined cycle, for $a, a' \in A$ such that $C_{a,a'} = 1$

Table 3. Characteristics of the anneal cycles.

may come before the release date r_i. These orders are identified as late, and are given high preference to be scheduled as early as possible. Only orders for which the due date is at most a given date D (for example, day two) are required to be scheduled. Lastly, there exist pairwise precedence relations (i, j) for $i, j \in O$: order j must be scheduled at least three days after order i finishes. In fact, i and j represent two annealing operations for the same set of coils.

The characteristics of the anneal cycles are given in Table 3. Some orders with different anneal cycles can be combined in one batch, if they are compatible. In that case, one of the cycles will determine the anneal recipe for the compatible batch. At most one other compatible cycle can be added to a batch. An order is not allowed to be both reduced and added as a compatible order to a batch.

Since not all anneal recipes can be performed on each furnace, we introduce a Boolean parameter $T_{f,a}$ to indicate whether furnace $f \in F$ can perform anneal cycle $a \in A$. Furthermore, each furnace can only handle a maximum of M 'heavy' coils that have a weight of at least W, within one batch. The maximum coil weight that can be processed on f is denoted by $w_f^{\max}$.

The problem is to 1) group together orders into batches (the batch design problem), and 2) allocate the batches to furnaces and sequence them over time (the batch scheduling problem). The purpose of the solution is to create a near-term schedule (about six days), in which we will operationally commit batches that are scheduled in the first two days. Therefore, the qualitative goals are:

- Minimize furnace idle time, especially in the first two to three days;
- Minimize unfilled furnace capacity (i.e., maximize the batch sizes), especially in the first two to three days;
- Minimize the number of late coils.

3 Phase 1: Batch Design

We next describe our MIP model for the batch design problem. We recall that the locked orders in O^L are already grouped in batches, so we consider here the work orders O^W and planned orders O^P. The first step is to define the possible batches that we can assign the orders to. Since orders will be grouped by anneal cycle, we create a set $\{b_{a,1}, \ldots, b_{a,N_a}\}$ for $a \in A$, representing the possible batches for anneal cycle a. The size N_a of this set depends on the total

$x_{i,a,k} \in \{0,1\}$	allocate order i to $b_{a,k}$, for $i \in O^W \cup O^P, a \in A$ such that $c_i = a$ and $k \in [N_a]$
$y_{i,a,k} \in \{0,1\}$	allocate i as *reduced* order to $b_{a,k}$, for $i \in O^P, a \in A$ such that $c_i = a$ and $k \in [N_a]$
$z_{i,a,k} \in \{0,1\}$	allocate i as *compatible* order to $b_{a,k}$, for $i \in O^W \cup O^P, a \in A$ such that $C_{c_i,a} = 1$, and $k \in [N_a]$
$y'_{i,a,k} \in [0, \frac{(K-1)}{K}]$	fraction of coils from (reduced) order i that will be used in $b_{a,k}$, for $i \in O^P$, $a \in A$ such that $c_i = a$ and $k \in [N_a]$
$u_{a,k} \in \{0,1\}$	whether $b_{a,k}$ is in use, for $a \in A$, $k \in [N_a]$
$z'_{a,k,a'} \in \{0,1\}$	whether $b_{a,k}$ is in use as compatible batch for a', for $a \in A$, $a' \in A, a \neq a', C_{a,a'} = 1, k \in [N_a]$
$e_{a,k} \in \{0,\ldots,K-1\}$	number of empty positions in $b_{a,k}$, for $a \in A$, $k \in [N_a]$
$l_{a,k} \in \{0,1\}$	whether $b_{a,k}$ contains a late order, for $a \in A$, $k \in [N_a]$
$m_{a,k} \geq 0$	maximum release date of orders in $b_{a,k}$, for $a \in A, k \in [N_a]$
$m'_{a,k} \geq 0$	release date violation of first-day batches $b_{a,k}$, for $a \in A$, $k \in [N_a], k \leq k^I_a$

Table 4. Variables used in the mixed-integer programming model.

number of coils that can be assigned to cycle a, the furnace capacity K, and the precedence constraints between orders, and is computed in advance.

Our MIP model will not keep track of time in full detail, as this will be the responsibility of the batch scheduling model. However, we do need to take timing considerations into account, for example by aiming to group together orders with the same release date. We will therefore associate an earliest release date with each batch, and our model is designed to create batches such that the release date of $b_{a,k}$ is at most the release date of $b_{a,k+1}$, if both are used.

In addition, we wish to avoid grouping first-day orders with orders that have a later release date, to create enough batches to schedule on the first day of the horizon. To that end, we identify all orders (for a given anneal cycle $a \in A$) that can start on the first day of the horizon, and determine (approximately) the number of batches that can start on the first day as $k^I_a = \lfloor (\sum_{i \in O^W \cup O^P : c_i = a, r_i \leq 1} n_i)/K \rfloor$. We will use this information to group together first-day orders (only) in as many batches as possible.

The MIP model is as follows (the variables are presented in Table 4):

$$\begin{aligned} \min \quad & \sum_{a \in A, k \in [N_a]} (\gamma^l l_{a,k} + \gamma^e_{a,k} e_{a,k}) + \sum_{\substack{i \in O^P, a \in A, k \in [N_a]: \\ c_i = a}} \gamma^y y_{i,a,k} + \\ & \sum_{\substack{i \in O^W \cup O^P, \\ a \in A, k \in [N_a]: C_{c_i,a} = 1}} \gamma^z z_{i,a,k} + \sum_{\substack{a \in A, k \in [N_a]: \\ k \leq k^I_a}} \gamma^m m'_{a,k} \end{aligned} \tag{1}$$

$$\text{s.t.} \quad \sum_{a \in A, k \in [N_a]: c_i = a} x_{i,a,k} + \sum_{a \in A, k \in [N_a]: C_{c_i,a}} z_{i,a,k} \leq 1 \qquad \forall i \in O^W \tag{2}$$

$$\sum_{a\in A,k\in[N_a]:c_i=a}(x_{i,a,k}+y_{i,a,k})+\sum_{a\in A,k\in[N_a]:C_{c_i,a}}z_{i,a,k}\le 1\quad\forall i\in O^P \tag{3}$$

$$\sum_{\substack{i\in O^P\cup O^W:\\ c_i=a,d_i<r_i}}(x_{i,a,k}+y_{i,a,k})+\sum_{\substack{i\in O^P\cup O^W:\\ C_{c_i,a}=1,d_i<r_i}}z_{i,a,k}\le Kl_{a,k}\quad\forall a\in A,k\in[N_a] \tag{4}$$

$$\sum_{i\in O:c_i=a}(n_ix_{i,a,k}+n_iy'_{i,a,k})+\sum_{i\in O:C_{c_i,a}}n_iz_{i,a,k}=Ku_{a,k}-e_{a,k}\quad\forall a\in A,k\in[N_a] \tag{5}$$

$$y'_{i,a,k}\le y_{i,a,k}(n_i-1)/n_i\quad\forall i\in O^P,a\in A,c_i=a,k\in[N_a] \tag{6}$$

$$y'_{i,a,k}\ge y_{i,a,k}/n_i\quad\forall i\in O^P,a\in A,c_i=a,k\in[N_a] \tag{7}$$

$$\sum_{i\in O^P:c_i=a}y_{i,a,k}\le 1\quad\forall a\in A,k\in[N_a] \tag{8}$$

$$\sum_{i\in O:c_i=a'}n_iz_{i,a,k}\le z'_{a,k,a'}(K-1)\quad\forall a\in A,k\in[N_a],a'\in A:C_{a,a'}=1 \tag{9}$$

$$\sum_{a'\in A:C_{a,a'}=1}z'_{a,k,a'}\le 1\quad\forall a\in A,k\in[N_a] \tag{10}$$

$$\sum_{\substack{a\in A,k\in[N_a]:\\ c_i=a,w_i\ge W}}n_i(x_{i,a,k}+y'_{i,a,k})+\sum_{\substack{a\in A,k\in[N_a]:\\ C_{c_i,a},w_i\ge W}}n_iz_{i,a,k}\le M\quad\forall a\in A,k\in[N_a] \tag{11}$$

$$\begin{aligned}\sum_{\substack{a\in A,\\ k\in[N_a]:\\ c_i=a}}k(x_{i,a,k}+y_{i,a,k})+\sum_{\substack{a\in A,\\ k\in[N_a]:\\ C_{c_i,a}=1}}kz_{i,a,k}\le\sum_{\substack{a\in A,\\ k\in[N_a]:\\ c_j=a}}k(x_{j,a,k}+y_{j,a,k})+\sum_{\substack{a\in A,\\ k\in[N_a]:\\ C_{c_j,a}=1}}kz_{j,a,k}\\ -1+(1+N_{c_i})\left(1-\sum_{\substack{a\in A,\\ k\in[N_a]:\\ c_j=a}}(x_{j,a,k}+y_{j,a,k})+\sum_{\substack{a\in A,\\ k\in[N_a]:\\ C_{c_j,a}=1}}z_{j,a,k}\right)\quad\begin{array}{c}\forall(i,j)\in\Pi:\\ c_i=c_j\vee C_{c_i,c_j}=1\end{array}\end{aligned} \tag{12}$$

$$\sum_{\substack{a\in A,\\ k\in[N_a]:\\ c_j=a}}(x_{j,a,k}+y_{j,a,k})+\sum_{\substack{a\in A,\\ k\in[N_a]:\\ C_{c_j,a}=1}}z_{j,a,k}\le\sum_{\substack{a\in A,\\ k\in[N_a]:\\ c_i=a}}(x_{i,a,k}+y_{i,a,k})+\sum_{\substack{a\in A,\\ k\in[N_a]:\\ C_{c_i,a}=1}}z_{i,a,k}\quad\forall(i,j)\in\Pi \tag{13}$$

$$\sum_{\substack{a\in A,\\ k\in[N_a]:\\ c_i=a}}x_{i,a,k}+\sum_{\substack{a\in A,\\ k\in[N_a]:\\ C_{c_i,a}}}z_{i,a,k}\ge\sum_{\substack{a\in A,\\ k\in[N_a]:\\ c_j=a}}x_{j,a,k}\sum_{\substack{a\in A,\\ k\in[N_a]:\\ C_{c_j,a}}}z_{j,a,k}\quad\begin{array}{c}\forall i,j\in O^W\cup O^P:\\ (c_i=c_j\vee C_{c_i,c_j}=1)\\ d_i<d_j\end{array} \tag{14}$$

$$\sum_{\substack{a\in A,\\ k\in[N_a]:\\ c_i=a}}y_{i,a,k}\ge\sum_{\substack{a\in A,\\ k\in[N_a]:\\ c_j=a}}y_{j,a,k}\quad\begin{array}{c}\forall i,j\in O^P:\\ (c_i=c_j\vee C_{c_i,c_j}=1)\\ d_i<d_j\end{array} \tag{15}$$

$$K\left(\sum_{\substack{i\in O^W\cup O^P:\\ c_i=a}} x_{i,a,k}+\sum_{\substack{i\in O^W\cup O^P:\\ C_{c_i=a}=1}} z_{i,a,k}\right)\geq \sum_{\substack{i\in O^W\cup O^P:\\ c_i=a}} x_{i,a,k+1}\ + \tag{16}$$

$$\sum_{\substack{i\in O^W\cup O^P:\\ C_{c_i=a}=1}} z_{i,a,k+1}\quad \forall a\in A, k\in[N_a-1]$$

$$\sum_{\substack{a\in A,k\in[N_a]:\\ c_i=a}} x_{i,a,k}+\sum_{\substack{a\in A,k\in[N_a]:\\ C_{c_i,a}}} z_{i,a,k}=1\quad \forall i\in O^W\cup O^W: d_i\leq D \tag{17}$$

$$m_{a,k}\geq r_i(x_{i,a,k}+y_{i,a,k})\quad \forall a\in A, k\in[N_a], i\in O^W\cup O^P: c_i=a \tag{18}$$

$$m_{a,k}\geq r_i z_{i,a,k}\quad \forall a\in A, k\in[N_a], i\in O^W\cup O^P: C_{c_i,a}=1 \tag{19}$$

$$m_{a,k}\leq m_{a,k+1}\quad \forall a\in A, k\in[N_a-1] \tag{20}$$

$$m_{a,k}\leq 1+m'_{a,k}\quad \forall a\in A, k\in[N_a]: k\leq k_a^I \tag{21}$$

The objective (1) is a weighted sum of penalty terms representing the number of batches with late orders, the number of empty slots, the number of reduced orders, the number of compatible orders, and the tardiness of first-day orders. Each term has its associated penalty parameter, denoted by $\gamma^l, \gamma^e_{a,k}, \gamma^y, \gamma^z$, and γ^m, respectively (listed here by decreasing emphasis). Parameter $\gamma^e_{a,k}$ is defined as $\gamma^e_{a,k}=1+0.1(N_a-k)/N_a$. It decreases for larger k, giving higher priority to filling earlier batches. The other penalty parameters are taken in $[0.01, 1.0]$.

Constraints (2) and (3) allocate the work orders and planned orders, respectively. Constraints (4) define a late batch with respect to the late orders. Constraints (5) define the capacity of the furnaces and the number of empty slots. Constraints (6) and (7) link the fraction of coils for reduced orders to the associated binary variable, while constraints (8) ensure that each planned order can be used at most once as reduced order. Constraints (9) define compatible batches with respect to compatible orders, while constraints (10) ensure that each batch can be compatible with at most one other anneal cycle. Constraints (11) limit the number of heavy coils that can be allocated to a batch. Constraints (12) and (13) represent the precedence constraints. The model ensures feasibility by allocating orders i and j, with $(i,j)\in \Pi$, to different batches. If they are allocated to different anneal cycles, this is already guaranteed. Otherwise, if $c_i=c_j$ or $C_{c_i,c_j}=1$, constraints (12) ensure that order i is placed in a batch with a lower index k than j. Constraints (13) ensure that j is not allocated if i is not. Constraints (14) and (15) state that if two orders are equivalent in terms of anneal cycle, then the one with earlier due date is allocated first, for work orders and planned orders, respectively. Constraints (16) impose that batch $b_{a,k+1}$ can only be used if $b_{a,k}$ is. Constraints (17) ensure that all near-term orders (with $d_i\leq D$) are allocated. Constraints (18) and (19) define the release dates of the batches. Constraints (20) ensure that batches are ordered by non-decreasing release date. Constraints (21) define the release violation $(m'_{a,k})$ of first-day batches.

$\bar{w}_b \in \mathbb{R}^+$	$\max_{i \in \mathcal{O}(b)} w_i$
$\bar{n}_b \in [K]$	$\sum_{i \in \mathcal{O}(b)} n_i$
$\bar{r}_b \in H$	$\max_{i \in \mathcal{O}(b)} r_i$
$\bar{d}_b \in H$	$\min_{i \in \mathcal{O}(b)} d_i$

$\bar{c}_b \in A$	if $z'_{a,k,a'} = 1$ then $\Gamma_{a,a'}$ else a, for $b = b_{a,k}$
$\bar{g}_b \subseteq F$	g_i for any $i \in \mathcal{O}(b)$ if $b \in B^L$
$\bar{\Pi} \subset B \times B$	(b, b') if $\exists i \in \mathcal{O}(b), i' \in \mathcal{O}(b') : (i, i') \in \Pi$

Table 5. Characteristics of the batches and their definition.

4 Phase 2: Batch Scheduling

For the batch scheduling problem we employ a constraint programming model, following the constraint-based scheduling concepts of activities and resources [1]. In the model syntax below, we follow the definitions of AIMMS [7]. An *activity* represents a task to be scheduled over time, by means of four implied variables: begin, length, end, and presence. When an activity is *optional*, its presence can be either 0 (absent) or 1 (present). When an activity is present, it respects the relation 'begin+length=end'. *Resources*, for example machines, represent constraints on activities. In this paper, we only consider sequential (or disjunctive) resources that limit the execution to at most one activity at a time. In addition, the sequential resource can enforce sequence-dependent setup (or switchover) times between two consecutive activities.

Our model is based on the problem definition from Section 2 and the MIP solution from Section 3. We first define the set of batches B as the union of the locked batches (denoted by B^L) and the batches $b_{a,k}$ that are used in the MIP model, i.e., for which $u_{a,k} = 1$ (for $a \in A, k \in [N_a]$). We use $\mathcal{O}(b) \subseteq O$ to denote the orders that are assigned to batch b. We define the associated parameters of a batch in Table 5. We introduce the following activities:

L_b: the execution of batch b, for $b \in B$

$L_{b,f}^{opt}$: the execution of batch b on furnace f, for $b \in B, f \in F : \bar{w}_b \leq w_f^{\max}$, $T_{f,\bar{c}_b} = 1$, and $f \in \bar{g}_b$ if $b \in B^L$

We define the schedule domain for both types of activities as $[\bar{r}_b, H]$, for $b \in B$. Activities $L_{b,f}^{opt}$ are optional, while activities L_b must always be present. The processing time for both L_b and $L_{b,f}^{opt}$ is given by $p_{\bar{c}_b,f}$.

We introduce a sequential resource R_f for each furnace $f \in F$. As arguments it receives the activities $L_{b,f}^{\text{opt}}$, as well as the sequence-dependent switchover times $t_{a,a',f}$. These resources will ensure that all activities $L_{b,f}$ that are present do not overlap in time, and respect the switchover times.

The additional constraints for our CP model are as follows:

$$\texttt{Alternative}(L_b, \{L_{b,f}^{\text{opt}}\}_{f \in F}, 1) \quad \forall b \in B \tag{22}$$

$$\texttt{EndBeforeBegin}(L_b, L_{b'}, 4320) \quad \forall (b, b') \in \bar{\Pi} \tag{23}$$

$$\texttt{EndBeforeBegin}(L_{b,f}^{\text{opt}}, L_{b',f}^{\text{opt}}) \quad \forall b, b' \in B, f \in F : b \neq b', \bar{r}_b = \bar{r}_{b'}, \bar{d}_b < \bar{d}_{b'}, \bar{n}_b = K, (b', b) \notin \bar{\Pi} \tag{24}$$

$$(L_b)\texttt{.end} \leq \bar{d}_i + v_b \quad \forall b \in B \tag{25}$$

The `Alternative` constraints (22) ensure that exactly one optional activity $L^{\text{opt}}_{b,f}$ is present for each $b \in B$, and it coincides with the execution of activity L_b. The precedence constraints are defined by constraints (23).[2] Constraints (24) are added to streamline the solutions, by ordering pairs of activities on a furnace by due date. Lastly, we introduce an integer variable v_b, for $b \in B$, that represents its due date violation (or tardiness), as defined in constraint (25). In this constraint (and in the objective function below), `.end` is used to retrieve the end variable of an activity.

We conclude with the objective function, which is a weighted sum of completion times and tardiness over different groups of activities:

$$\min \quad \bar{\gamma}^L \sum_{b \in B^L} (L_b)\texttt{.end} + \bar{\gamma}^I \sum_{b \in B^I} \bar{n}_b (L_b)\texttt{.end} + \bar{\gamma}^V \sum_{b \in B} \bar{n}_b v_b \tag{26}$$

Here, B^I refers to the batches that contain first-day orders (these refer to the batches $b_{a,k}$ with $k \leq k^I_a$). The weights $\bar{\gamma}^L$, $\bar{\gamma}^I$, and $\bar{\gamma}^V$ have decreasing value in the range $(0, 100]$, placing most emphasis on locked orders and first-day orders.

5 Implementation and Results

We implemented our MIP and CP models in the optimization modeling system AIMMS, using IBM ILOG CPLEX 12.4 for solving the MIP and CP models. We also used AIMMS to build an end-user interface for operational use.

The purpose of our tool is to plan, or revisit, the design and scheduling of annealing batches on a daily basis. A typical run may receive about 600 orders (with 100 locked orders, in 60 batches). The batch design MIP model for the remaining 500 orders has about 20,000 variables (18,000 integer) and 30,000 constraints. A feasible integer solution (with optimality gap 30%) is found within a couple seconds, while a near-optimal solution (1% optimality gap) is found within 15 minutes. The model creates about 190 batches, which together with the locked batches makes around 250 batches to be scheduled. For about 15 available furnaces, the resulting CP model has about 8,000 variables and 25,000 constraints. It finds a feasible solution instantly, and typically returns solutions in about 2 minutes that are not necessarily optimal, but are considered of good quality by the client with respect to the goals listed in Section 2.

6 Conclusion

We introduced a two-phase optimization approach to design and schedule batches for box annealing furnaces. We first allocate a given set of orders to batches using a mixed-integer programming model. We then solve the batch scheduling problem using a constraint programming model. Using this decomposition, our approach is able to compute operational schedules for a one-week planning horizon within 15 to 30 minutes of computation time.

[2] Recall that b' must be scheduled at least three days (4320 minutes) after b finishes.

References

1. P. Baptiste, C. Le Pape, and W. Nuijten. *Constraint-Based Scheduling: Applying Constraint Programming to Scheduling Problems.* Kluwer Academic Publishers, 2001.
2. P. M. Castro, M. Erdirik-Dogan, and I. E. Grossmann. Simultaneous batching and scheduling of single stage batch plants with parallel units. *AIChE Journal*, 54(1):183–193, 2008.
3. I. Harjunkoski and I. E. Grossmann. A decomposition approach for the scheduling of a steel plant production. *Computers & Chemical Engineering*, 25(11–12):1647–1660, 2001.
4. Q. Liu, W. Wanga, H. Zhanb, Z. Wanga, and R. Liua. Optimal scheduling method for a bell-type batch annealing shop and its application. *Control Engineering Practice*, 13(10):1315–1325, 2005.
5. C. A. Méndez, J. Cerdá, I. E. Grossmann, I. Harjunkoski, and M. Fahl. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers and Chemical Engineering*, 30:913–946, 2006.
6. S. Moon and A. N. Hrymak. Scheduling of the batch annealing process – deterministic case. *Computers & Chemical Engineering*, 23(9):1193–1208, 1999.
7. M. Roelofs and J. J. Bisschop. *AIMMS 4.2: The Language Reference.* AIMMS B.V., 2014.