Available online at www.sciencedirect.com
SCIENCE DIRECT®
Mathematics and Computers in Simulation 66 (2004) 181–191
MATHEMATICS AND COMPUTERS IN SIMULATION
www.elsevier.com/locate/matcom

Time-optimal scheduling for high throughput screening processes using cyclic discrete event models

E. Mayer [a,*], J. Raisch [a,b]

[a] *Fachgruppe System und Regelungstheorie, Max-Planck-Institut für Dynamik Komplexer Technischer Systeme, 39106 Magdeburg, Germany*

[b] *Lehrstuhl für Systemtheorie Technischer Prozesse, Otto-von-Guericke-Universität, 39016 Magdeburg, Germany*

Available online 15 January 2004

Abstract

A method for solving the scheduling problem for a class of cyclic systems with respect to throughput maximization is presented. A strictly cyclic mode of operation is considered, where the time offset between the start of consecutive jobs is always constant. All jobs have to follow an identical time scheme. The time scheme may be restricted by due dates or time window constraints. There are no buffers between the resources of the system.

Based on discrete events systems (DES) modeling, this job shop scheduling problem can be formulated as a mixed integer linear optimization problem. Throughput is maximized by minimization of the cycle time. The method is applied to high throughput screening (HTS) problems and illustrated by means of a small example.

Keywords: Cyclic scheduling; Throughput maximization; Mixed integer optimization; Discrete event systems; High throughput screening

1. Introduction

High throughput screening (HTS) plants are used for the analysis of chemical or biological substances. Typically, several operations have to be executed in the same specific time scheme for a large number of sample batches. For such processes, the task of throughput maximization is very important. Determining the throughput-optimal sequence and timing for all operations of a run of several hundreds of batches (‘assay’) can yield significant savings in operating time and costs. Such *scheduling problems* are known to be np-hard in most cases. For several specialized HTS plants, a number of scheduling approaches with a variety of objectives exist, e.g. [1,3]. However, as development goes towards large flexible HTS plants,

* Corresponding author.
E-mail address: eckart.mayer@mpi-magdeburg.mpg.de (E. Mayer).

0378-4754/$30.00
doi:10.1016/j.matcom.2003.11.004

a general scheduling approach is needed which can be used independently of the specific combination of machines and transport devices.

HTS scheduling differs from other scheduling problems, e.g.in chemical engineering [2,4], in the following aspects: while progressing through the operations, each single batch may pass the same machine more than once. More than one batch will be present in the system at the same time. There are no buffers between the machines. Moreover, a batch may occupy two or more machines simultaneously when being transferred from one machine to another. Additionally, there will be upper time bounds ('due dates') or time window constraints stated by the user. In many cases, due to the specific nature of the substances to be screened, operating schemes have to be strictly cyclic. This means that the time distance between two consecutive batches is required to be always constant. We show how this cyclic scheduling problem can be formulated as a mixed integer linear optimization problem. Its solution leads to a globally optimal schedule with maximum throughput, i.e. minimum cycle time, and therefore to minimum overall processing time ('makespan').

This paper is arranged as follows: first, a DES model for cyclic high throughput screening systems is discussed. The search space of the scheduling problem is reduced by simplification of the model without affecting the globally optimal solution. Subsequently, the scheduling problem for throughput maximization is derived. Its formulation as a mixed integer optimization problem is presented. Finally, a small example is treated to illustrate the basic solution approach.

2. A cyclic DES model for a HTS plant

The HTS plant is assumed to consist of m resources, i.e. incubators, readers, transport devices, etc. Substances are aggregated in batches, which are realized by so called microplates holding up to 1536 substances. They pass the plant following a time scheme given by the user. As a basic principle, the time scheme is identical for each batch (called *single-batch time scheme*). It consists of $i_{\max}$ activities. Each activity i allocates one of the resources, which is denoted by $J_i \in \{1 \dots m\}$. For each activity i, $i = 1 \dots i_{\max}$, the respective resource is allocated starting from time o_i (starting event) until time r_i, where $r_i > o_i$ (release event). As all resources have capacity one, no other activity can use the same resource during this period. The sequence of activities on each single resource is assumed to be fixed:

$$o_{i2} \geq r_{i1} \quad \forall \ \{(i1, i2), \ \ i2 > i1, J_{i1} = J_{i2}\}. \tag{1}$$

Note that the time instants where activities start and end are not necessarily identical to the time instants at which the corresponding plates enter respectively leave the resources, as there may be additional pre- or post-processing. Transfers of plates between the resources are modeled by transfer events, which connect activities on different resources.

One possibility of modeling temporal interdependency between the events of the single-batch time scheme is the use of a timed graph. A short explanation of such a model will be given in the following. The nodes of the graph represent the events within the single-batch time scheme. As discussed above, there are three types of events:

- starting events: start of activity (time o_i),
- release events: end of activity (time r_i), and

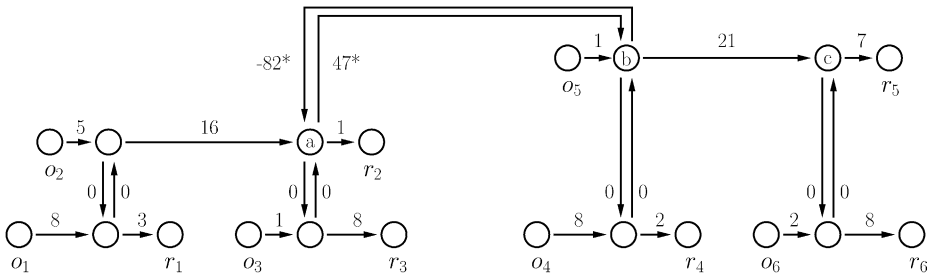

Fig. 1. Graph model for the single-batch time scheme.

- transfer events: transfer of a plate between two resources. A transfer event always takes place simultaneously to at least one corresponding transfer event associated with another resource.

The (directed) arcs of the graph represent temporal interdependencies. A label is assigned to every arc, describing the minimum time which has to elapse between the two events. An arc leading from node a to node b, labeled with time $t_\mathrm{l} \in \mathbb{R}$ implies that at least time t_l has to pass between occurrence of the event represented by a and occurrence of the event represented by b. It is also possible to model upper bounds for the time interval between two events. For example if the time interval between a and b is less or equal to t_u, the graph contains an arc from b to a labeled by $-t_\mathrm{u}$.

Temporal interdependencies may originate from the following requirements:

- Operations (i.e. activities or parts thereof) require a certain minimum time interval for processing, e.g. a plate can never leave a machine before work is finished. In HTS, these constraints are normally investigated by test runs on the real plant.
- Starting of one operation depends on completion of another operation, e.g. an operation for a plate cannot start before the preceding operation has been finished.
- Due to chemical requirements, the time that is allowed to elapse between two (not necessarily consecutive) operations is restricted by lower and/or upper bounds. Such requirements will be prescribed by the user.

An example for such a graph defining temporal interdependencies for the events of a single-batch time scheme is given in Fig. 1.

The graph representation could, of course, be directly translated into mathematical inequality constraints arc by arc. In this case, one would need one variable for each event describing the time when this event occurs. The number of variables would therefore equal the number of nodes and each arc would provide one inequality restriction, implying that the mathematical formulation becomes unnecessarily complex and often unmanageable.

Often, the mathematical form can be substantially simplified by suitably parameterizing the time instants o_i and r_i.[1] We represent the o_i and r_i as linear combinations of K time variables $\theta_k \in \mathbb{R}_0^+, k = 1 \ldots K$, where normally $K \ll 2i_\mathrm{max}$:

$$\begin{aligned} o_i &= \chi_{i,0} + \textstyle\sum_{k=1}^{K} (\chi_{i,k} \cdot \theta_k) \quad \ldots \quad \text{Time, when activity } i \text{ starts,} \\ r_i &= \psi_{i,0} + \textstyle\sum_{k=1}^{K} (\psi_{i,k} \cdot \theta_k) \quad \ldots \quad \text{Time, when activity } i \text{ ends.} \end{aligned} \tag{2}$$

[1] The time instants for the *transfer events* are not needed to formulate the constraints for the scheduling problem.

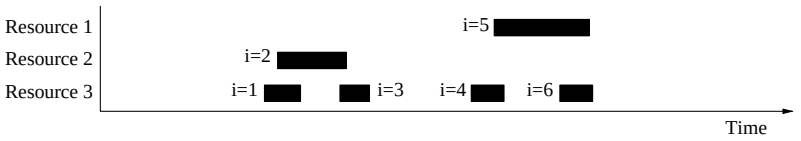

Fig. 2. Example for a single-batch time scheme (Gantt chart).

This means that the single-batch time scheme is described by fixed parameters $\chi_{i,0}$, $\psi_{i,0}$, $\chi_{i,k}$, and $\psi_{i,k}$, $i = 1 \dots i_{\max}$, and yet unknown variables θ_k, $k = 1 \dots K$.

The constraints on the variables o_i and r_i are represented by bounds for the time variables θ_k:

$$0 \leq \theta_k \leq \theta_{k,\max}, \quad k = 1 \dots K \tag{3}$$

and, if necessary, by additional linear constraints of the form

$$\sum_{k=1}^{K} (\kappa_{p,k} \cdot \theta_k) \leq \vartheta_p, \quad p = 1 \dots P. \tag{4}$$

Fixing the event times for the single-batch time scheme defines all event times within all batches, because all batches follow each other in a strictly cyclic manner. This means that the time instants for start and release events for activity i of batch ρ, $\rho \in \mathbb{Z}$ are

$$o_i^{(\rho)} = o_i + \rho \cdot T \quad \text{and} \quad r_i^{(\rho)} = r_i + \rho \cdot T \tag{5}$$

respectively, where T is called period or cycle time, i.e. T is the time which elapses between the same event within two consecutive batches. Normally, several batches are present in the plant simultaneously.

An example for a single-batch time scheme, pictured as a Gantt chart, is given in Fig. 2. An extract of a corresponding cyclic schedule is given in Fig. 3.

An additional condition for the cycle time T is that it can never be smaller than the sum of activity durations on any resource during the single-batch time scheme:

$$T \geq \sum_{i=1}^{i_{\max}} (r_i - o_i)\delta_{J_i j}, \quad j = 1 \dots m, \tag{6}$$

$$\text{where } \delta_{J_i j} = \begin{cases} 0 & \text{for } J_i \neq j \\ 1 & \text{for } J_i = j. \end{cases}$$

Having defined a mathematical model for cyclic operating sequences, we can now formulize the cyclic scheduling problem and its solution.

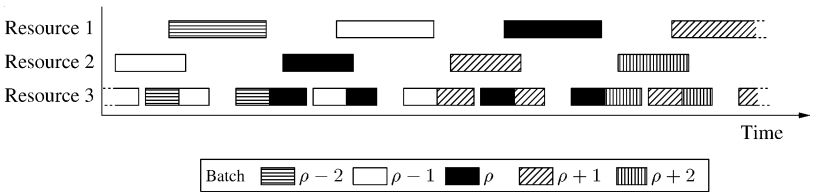

Fig. 3. Extract from cyclic schedule for the single-batch time scheme from Fig. 2.

3. The cyclic scheduling problem

Throughput maximization for a cyclic system is equivalent to the following scheduling problem:

Find $\theta_k, k = 1 \dots K$ such that the cycle time T is minimized subject to constraints (3), (4), (6) and the additional constraint that no allocation conflicts arise.

As all resources have capacity one, a resource can never be allocated by two activities simultaneously[2]. This has not only to hold for the single-batch time scheme but also with regard to all batches that are present in the plant at the same time. This requirement leads to mathematical conditions, which are often called *disjunctive constraints*. For two activities $(i1, i2)$ of the same resource $J_{i1} = J_{i2}$ belonging to batch $\rho 1$ and $\rho 2$, respectively, the following condition ensures exclusion of overlapping:[3]

$$o_{i1}^{(\rho 2)} \geq r_{i2}^{(\rho 1)} \ \text{OR} \ o_{i2}^{(\rho 1)} \geq r_{i1}^{(\rho 2)}. \tag{7}$$

This condition has to be met for each pair of activities using the same resource, including activities within the same batch and including the same activity within different batches.

Condition (6) ensures (7) for $i1 = i2$. It is also assumed that the constraints for the single-batch time scheme as defined in Section 2 ensure Condition (7) for all pairs of activities within the same batch, i.e. for $\rho 1 = \rho 2$.

For the remaining cases ($i1 \neq i2$, $\rho 1 \neq \rho 2$), due to assumption (1), it is sufficient to ensure (7) for $i1 < i2$ and $\rho 1 < \rho 2$. Moreover, as cyclically repeated sequences are considered, this can be narrowed down to $\rho 1 = 0$ and $\rho 2 =: \rho$, $(\rho = 1, 2, \dots)$. Eq. (7) can then be reformulated using (5):

$$o_{i1} + \rho \cdot T \geq r_{i2}, \tag{8a}$$

OR

$$o_{i2} \geq r_{i1} + \rho \cdot T \quad \rho = 1, 2, \dots . \tag{8b}$$

We now derive a necessary and sufficient condition for (8). As (8) has to hold for any ρ, either (8a) or (8b) has to hold for

$$\rho = \left\lfloor \frac{o_{i2} - r_{i1}}{T} \right\rfloor + 1, \tag{9}$$

where $\lfloor x \rfloor$ denotes the floor function, i.e. the largest integer less or equal to x. For (9), Constraint (8b) cannot be satisfied. Hence, Constraint (8a) has to hold. Substituting (9) into (8a) gives a necessary condition for (8):

$$\left(\left\lfloor \frac{o_{i2} - r_{i1}}{T} \right\rfloor + 1 \right) \cdot T - (r_{i2} - o_{i1}) \geq 0. \tag{10}$$

[2] However, there are no switching times, i.e. two allocations of the same resource can immediately follow each other.

[3] In (7), OR and XOR are equivalent as only one part of Condition (7) can be satisfied.

We now show that Condition (10) is also sufficient for (8). In order to prove sufficiency, we will consider two cases:

- For $\rho \geq \left\lfloor \frac{o_{i2} - r_{i1}}{T} \right\rfloor + 1$, Condition (10) implies

$$\rho \cdot T - (r_{i2} - o_{i1}) \geq 0,$$

and therefore (8a).

- For $\rho < \left\lfloor \frac{o_{i2}-r_{i1}}{T} \right\rfloor + 1$, we get $\rho \leq \left\lfloor \frac{o_{i2}-r_{i1}}{T} \right\rfloor$, because ρ is defined to be an integer number. Hence,

$$\rho \cdot T - (o_{i2} - r_{i1}) \leq 0,$$

and therefore (8b).

Condition (10) is now formulated in a more useful way by introducing a new integer variable $z_{(i1,i2)}$:

$$z_{(i1,i2)} := \left\lfloor \frac{o_{i2} - r_{i1}}{T} \right\rfloor. \tag{11}$$

Obviously, Condition (10) is equivalent to

$$(z_{(i1,i2)} + 1) \cdot T - (r_{i2} - o_{i1}) \geq 0, \tag{12}$$

with the additional restrictions

$$z_{(i1,i2)} \in \mathbb{Z}_0^+, \tag{13}$$

$$z_{(i1,i2)} > \frac{o_{i2} - r_{i1}}{T} - 1, \tag{14}$$

$$z_{(i1,i2)} \leq \frac{o_{i2} - r_{i1}}{T}. \tag{15}$$

In fact, as $r_{i1} > o_{i1}$ and $r_{i2} > o_{i2}$, (14) is implied by (12).

Hence, equations (12), (13) and (15) constitute a necessary and sufficient condition for (8). To sum up, exclusion of allocation conflicts is guaranteed if

$$z_{(i1,i2)} \cdot T - (o_{i2} - r_{i1}) \leq 0, \tag{16}$$

$$(z_{(i1,i2)} + 1) \cdot T - (r_{i2} - o_{i1}) \geq 0, \tag{17}$$

holds for each pair of activities $(i1, i2), i1 < i2, J_{i1} = J_{i2}$, where $z_{(i1,i2)}$ is a non-negative integer: $z_{(i1,i2)} \in \mathbb{Z}_0^+$.

4. Formulation of the mixed integer optimization problem

In order to formulate the scheduling problem as an optimization problem, we take cycle time T as the objective function to be minimized subject to the constraints given by equations (16) and (17) as well as (3), (4), and (6). The search space for the optimization problem is defined by the variables $T \in \mathbb{R}^+$, $\theta_k \in \mathbb{R}_0^+$, and $z_{(i1,i2)} \in \mathbb{Z}_0^+$. Using (2), the HTS scheduling problem can therefore be written as the following mixed

integer nonlinear optimization program (MINLP), i.e. as an optimization problem involving both, real and integer variables:

Min T s. th.

$$z_{(i1,i2)} \cdot T - \left(\chi_{i2,0} + \sum_{k=1}^{K} (\chi_{i2,k} \cdot \theta_k) - \psi_{i1,0} - \sum_{k=1}^{K} (\psi_{i1,k} \cdot \theta_k) \right) \leq 0$$

$$(z_{(i1,i2)} + 1) \cdot T - \left(\psi_{i2,0} + \sum_{k=1}^{K} (\psi_{i2,k} \cdot \theta_k) - \chi_{i1,0} - \sum_{k=1}^{K} (\chi_{i1,k} \cdot \theta_k) \right) \geq 0$$

$$\text{for } \{(i1, i2), \quad i2 > i1, \quad J_{i1} = J_{i2}, \quad i1, i2 \in \{1 \dots i_{\max}\}\} \tag{18}$$

$$\theta_k \leq \theta_{k,\max} \quad \text{for } k = 1 \dots K \tag{19}$$

$$\sum_{k=1}^{K} (\kappa_{p,k} \cdot \theta_k) \leq \vartheta_p \quad \text{for } p = 1 \dots P \tag{20}$$

$$T \geq \sum_{i=1}^{i_{\max}} \left(\psi_{i,0} + \sum_{k=1}^{K} (\psi_{i,k} \cdot \theta_k) - \chi_{i,0} - \sum_{k=1}^{K} (\chi_{i,k} \cdot \theta_k) \right) \delta_{J_i j} \quad \text{for} \quad j = 1 \dots m,$$

$$\delta_{J_i j} = \begin{cases} 0 & \text{for } J_i \neq j \\ 1 & \text{for } J_i = j \end{cases} \tag{21}$$

Several software packages are available to address mixed integer nonlinear problems[4]. However, a straightforward application of such software packages to the above optimization problem can result in long computation times even for small problems. Moreover, in many cases, globally optimal solutions will not be found.

Fortunately, it is possible to transform (18)–(21) to linear form using

$$\bar{T} := \frac{1}{T}, \qquad \bar{\theta}_k := \frac{\theta_k}{T}, \quad k = 1 \dots K. \tag{22}$$

The HTS optimization problem can then be reformulated as follows:

Max $\bar{T}$ s. th.

$$z_{(i1,i2)} - \left(\chi_{i2,0} \cdot \bar{T} + \sum_{k=1}^{K} (\chi_{i2,k} \cdot \bar{\theta}_k) - \psi_{i1,0} \cdot \bar{T} - \sum_{k=1}^{K} (\psi_{i1,k} \cdot \bar{\theta}_k) \right) \leq 0 \tag{23}$$

$$(z_{(i1,i2)} + 1) - \left(\psi_{i2,0} \cdot \bar{T} + \sum_{k=1}^{K} (\psi_{i2,k} \cdot \bar{\theta}_k) - \chi_{i1,0} \cdot \bar{T} - \sum_{k=1}^{K} (\chi_{i1,k} \cdot \bar{\theta}_k) \right) \geq 0$$

$$\text{for} \{(i1, i2), \quad i2 > i1, \quad J_{i1} = J_{i2}, \quad i1, i2 \in \{1 \dots i_{\max}\}\} \tag{24}$$

$$\bar{\theta}_k - \theta_{k,max} \cdot \bar{T} \leq 0 \quad \text{for} \quad k = 1 \dots K \tag{25}$$

[4] For example GAMS/SBB (http://www.gams.com).

$$\sum_{k=1}^{K}(\kappa_{p,k}\cdot\bar{\theta}_k)-\vartheta_p\cdot\bar{T}\le 0\quad\text{for } p=1\dots P \tag{26}$$

$$\sum_{i=1}^{i_{\max}}\left(\psi_{i,0}\bar{T}+\sum_{k=1}^{K}(\psi_{i,k}\cdot\bar{\theta}_k)-\chi_{i,0}\bar{T}-\sum_{k=1}^{K}(\chi_{i,k}\cdot\bar{\theta}_k)\right)\delta_{J_i j}-1\le 0$$

$$\text{for } j=1\dots m,\ \delta_{J_i j}=\begin{cases}0 & \text{for } J_i\neq j\\ 1 & \text{for } J_i=j\end{cases} \tag{27}$$

This problem can be solved using advanced branch and bound techniques, e.g. the CPLEX library (http://www.ilog.com/products/cplex). Attention has to be paid to numerical aspects due to the fact that now the reciprocal of the original objective function is used.

5. A simplified example

In this section, the proposed method will be illustrated by a simple example. A cyclic schedule has to be derived for the single-batch time scheme pictured in Fig. 2. It consists of $i_{\max}=6$ activities. One activity takes place on both, resource 1 and 2: $J_2=1$, $J_5=2$. Four activities take place on resource 3: $J_1=J_3=J_4=J_6=3$. The corresponding graph is given in Fig. 1. Arcs are labeled with minimum time distances as described in Section 2. The events for those pairs of nodes that are connected with arcs labeled with time 0 in both directions need to happen simultaneously. These events represent the transfer of a plate between resources. The arcs labeled by * are constraints stated by the user: the transfer event associated with node b occurs at least 47 and at most 82 time units after the event associated with node a. Hence, the relative timing of both events can vary by up to 35 time units. Only three variables θ_k are needed to describe the degrees of freedom that have to be preserved in order to get a globally optimal solution. The linear combination (2) is chosen in such a way that the time variables $\theta_1\dots\theta_3$ can be interpreted as delays, which are inserted in the sequence of activities at the following positions:

- Variable θ_1 describes a delay during activity $i=2$, just before the transfer event associated with node a,
- variable θ_2 describes an extension of the time interval between activity $i=3$ and activity $i=4$ (No resource is allocated by the plate at that time),
- variable θ_3 describes a delay during activity $i=5$, just before the transfer event associated with node c.

Thus, the parameters $\chi_{i,0}$, $\psi_{i,0}$, $\chi_{i,k}$, and $\psi_{i,k}$ take the following values:

$$\begin{array}{llllllll}
\chi_{1,0}=0 & \chi_{1,1}=0 & \chi_{1,2}=0 & \chi_{1,3}=0 & \psi_{1,0}=11 & \psi_{1,1}=0 & \psi_{1,2}=0 & \psi_{1,3}=0\\
\chi_{2,0}=3 & \chi_{2,1}=0 & \chi_{2,2}=0 & \chi_{2,3}=0 & \psi_{2,0}=25 & \psi_{2,1}=1 & \psi_{2,2}=0 & \psi_{2,3}=0\\
\chi_{3,0}=23 & \chi_{3,1}=1 & \chi_{3,2}=0 & \chi_{3,3}=0 & \psi_{3,0}=32 & \psi_{3,1}=1 & \psi_{3,2}=0 & \psi_{3,3}=0\\
\chi_{4,0}=63 & \chi_{4,1}=1 & \chi_{4,2}=1 & \chi_{4,3}=0 & \psi_{4,0}=73 & \psi_{4,1}=1 & \psi_{4,2}=1 & \psi_{4,3}=0\\
\chi_{5,0}=70 & \chi_{5,1}=1 & \chi_{5,2}=1 & \chi_{5,3}=0 & \psi_{5,0}=99 & \psi_{5,1}=1 & \psi_{5,2}=1 & \psi_{5,3}=1\\
\chi_{6,0}=90 & \chi_{6,1}=1 & \chi_{6,2}=1 & \chi_{6,3}=1 & \psi_{6,0}=100 & \psi_{6,1}=1 & \psi_{6,2}=1 & \psi_{6,3}=1
\end{array} \tag{28}$$

Note that we could immediately find two *suboptimal* solutions to the scheduling problem:

- One (suboptimal) solution is to start each batch at the time when the previous batch is finished. The single-batch time scheme is chosen in such a way that the batch is finished as fast as possible. This obviously yields a cycle time of $T_1 = \psi_{6,0} - \chi_{1,0} = 100$.
- To find another suboptimal solution, we remove degrees of freedom by setting $\theta_1 = \theta_2 = \theta_3 = 0$. Hence, the only remaining degree of freedom is T and we determine the smallest T for which activities on the same resource do not overlap. This solution can be found by simple algorithms in polynomial time. It is pictured in Fig. 3 and has a cycle time of $T_2 = 50$.

In order to find a *globally optimal* solution to the scheduling problem, the search space is now enlarged by allowing variables $\theta_k, k = 1 \ldots 3$ to be greater than zero.

In order to include the condition represented by the arc labeled by -87^*, we need to introduce an upper bound for θ_2:

$$\theta_2 \leq \theta_{2,\max} = 35. \tag{29}$$

The values (28) and (29) are now substituted into the equations for the mixed integer nonlinear program (18)–(21).

There are six pairs of activities $(i1, i2)$, $i2 > i1$, $J_{i1} = J_{i2}$ for which constraints (18) need to be considered:

$$(i1, i2) \in \{(1, 3), (1, 4), (1, 6), (3, 4), (3, 6), (4, 6)\}.$$

This means we need to introduce the integer variables $z_{(1,3)}$, $z_{(1,4)}$, $z_{(1,6)}$, $z_{(3,4)}$, $z_{(3,6)}$, and $z_{(4,6)}$ together with six pairs of constraints of the form (18):

$$\begin{aligned}
z_{(1,3)} \cdot T - (12 + \theta_1) &\leq 0 \\
(z_{(1,3)} + 1) \cdot T - (32 + \theta_1) &\geq 0 \\
z_{(1,4)} \cdot T - (52 + \theta_1 + \theta_2) &\leq 0 \\
(z_{(1,4)} + 1) \cdot T - (73 + \theta_1 + \theta_2) &\geq 0 \\
z_{(1,6)} \cdot T - (79 + \theta_1 + \theta_2 + \theta_3) &\leq 0 \\
(z_{(1,6)} + 1) \cdot T - (100 + \theta_1 + \theta_2 + \theta_3) &\geq 0 \\
z_{(3,4)} \cdot T - (31 + \theta_2) &\leq 0 \\
(z_{(3,4)} + 1) \cdot T - (50 + \theta_2) &\geq 0 \\
z_{(3,6)} \cdot T - (58 + \theta_2 + \theta_3) &\leq 0 \\
(z_{(3,6)} + 1) \cdot T - (77 + \theta_2 + \theta_3) &\geq 0 \\
z_{(4,6)} \cdot T - (17 + \theta_3) &\leq 0 \\
(z_{(4,6)} + 1) \cdot T - (37 + \theta_3) &\geq 0
\end{aligned} \tag{30}$$

With (29), Condition (19) gives

$$\theta_2 \leq 35. \tag{31}$$

With values (28), Condition (21) results in

$$T \geq 29 + \theta_3, \qquad T \geq 22 + \theta_1, \qquad T \geq 40. \tag{32}$$

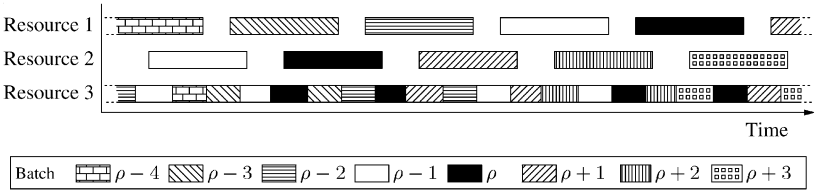

Fig. 4. Cyclic schedule for the single-batch time scheme from Fig. 2 with modified timing.

Hence, the mixed integer nonlinear program, which has to be solved in order to obtain a globally optimal solution to the scheduling problem is the following:

Min T over
$(T \in \mathbb{R}^+, \theta_k \in \mathbb{R}_0^+, z_{(i1,i2)} \in \mathbb{Z}_0^+, k \in \{1,2,3\}, (i1,i2) \in \{(1,3),(1,4),(1,6),(3,4),(3,6),(4,6)\})$
such that conditions (30), (31) and (32) hold.

The solution to this problem can be found using a MINLP solver or, of course, by transforming the problem into a mixed integer linear program using (22). A globally optimal solution is

$$T = 40, \quad \theta_1 = 8, \quad \theta_2 = 30, \quad \theta_3 = 3,$$
$$z_{(1,3)} = 0, \quad z_{(1,4)} = 2, \quad z_{(1,6)} = 3, \quad z_{(3,4)} = 1, \quad z_{(3,6)} = 2, \quad z_{(4,6)} = 0.$$

A graphical representation of the resulting cyclic schedule is given in Fig. 4. As additional degrees of freedom have been added, it is no surprise that this solution is better than the suboptimal solutions $T_1 = 100$ and $T_2 = 50$.

This example illustrated the main solution procedure of the proposed method for a simple problem. However, the approach has also been applied to real-world problems as they arise for modern, fully flexible HTS systems. Globally optimal solutions have been determined for assays involving up to 10 resources and 150 activities per single batch using GAMS/CPLEX (http://www.gams.com).

When solving mixed integer optimization problems, it is generally impossible to provide guaranteed bounds for computation times, as small changes to the problem structure may have significant impact on the computation time of the solver algorithm. Nevertheless, for all problems we considered computation of the globally optimal solution took only a few seconds and is therefore highly acceptable for the user.

6. Conclusion

The problem of finding a throughput-optimal schedule for the cyclic operation of High-Throughput-Screening plants has been treated. It has been shown that the constraints of this scheduling problem can be formulated using discrete event modeling. The problem can be cast into a mixed integer linear program and a globally optimal solution can be determined. The method has been illustrated by means of a basic example. Problem instances for real HTS plants result in optimization problems of reasonable size which can be solved within short computation time. However, the method is not limited to HTS applications. It may also be extended to similar problems in chemical engineering, flexible manufacturing or transportation systems.

Acknowledgements

The authors gratefully acknowledge funding by CyBio AG, Jena, Germany and the German Ministry of Economics and Technology via the PRO INNO project ‘Just-In-Time Optimierung verteilter Prozessabläufe im HTS-Labor’.

References

[1] A. Donzel, J. Carmona, L.A. Corkan, Perspectives on scheduling, in: J.P. Devlin (Ed.), High throughput screening, Marcel Dekker, New York, 1997, 525–544.
[2] M.G. Ierapetritou, C.A. Floudas, Effective Continuous-Time Formulation for Short-Term Scheduling. 1. Multipurpose Batch Processes, Ind. Eng. Chem. Res. 37 (1998) 4341–4359.
[3] C. Murray, C. Anderson, Scheduling Software for High-Throughput Screening, Laboratory Robotics & Automation 8 (5) (1996) 295–305.
[4] G. Schilling, C.C. Pantelides, Optimal periodic scheduling of multipurpose plants, Computers and Chemical Engineering 23 (1999) 635–655.