Discrete Event Dyn Syst (2008) 18:355–383
DOI 10.1007/s10626-008-0038-3

Throughput-Optimal Sequences for Cyclically Operated Plants

Eckart Mayer · Utz-Uwe Haus · Jörg Raisch · Robert Weismantel

Received: 28 November 2005 / Accepted: 24 January 2008 /
Published online: 15 March 2008

Abstract In this paper, we present a method to determine globally optimal schedules for cyclically operated plants where activities have to be scheduled on limited resources. In cyclic operation, a large number of entities is processed in an identical time scheme. For strictly cyclic operation, where the time offset between entities is also identical for all entities, the objective of maximizing throughput is equivalent to the minimization of the cycle time. The resulting scheduling problem is solved by deriving a mixed integer optimization problem from a discrete event model. The model includes timing constraints as well as open sequence decisions for the activities on the resources. In an extension, hierarchical nesting of cycles is considered, which often allows for schedules with improved throughput. The method is motivated by the application to high throughput screening plants, where a specific combination of requirements has to be obeyed (e.g. revisited resources, absence of buffers, or time window constraints).

Keywords Cyclic scheduling · High throughput screening · Integer programming

E. Mayer · J. Raisch
Max-Planck-Institut für Dynamik komplexer technischer Systeme,
39106 Magdeburg, Germany

U.-U. Haus · R. Weismantel
Institut für Mathematische Optimierung, Otto-von-Guericke-Universität Magdeburg,
39106 Magdeburg, Germany

J. Raisch (✉)
Fachgebiet Regelungssysteme, Technische Universität Berlin,
10587 Berlin, Germany
e-mail: Raisch@control.TU-Berlin.de

1 Introduction

Cyclic operation is characterized by the repetition of worksteps in a *periodic scheme*. This contribution considers cyclically operated plants, where the sequence and timing of worksteps is identical for all cycles. A usually large number of entities (*batches*) is processed successively in an identical time scheme. This can be advantageous or even compulsory for robot production lines or chemical batch processes as well as for transportation systems and in many other areas. A new field of application for cyclic operation methods are so-called *screening plants*, where a large number of substances are analyzed with respect to their benefit for chemical or pharmaceutical purposes. Determining the sequence and timing of the worksteps for a cyclic screening run is a scheduling problem (Brucker 2004; Pinedo 1995; Parker 1995) that is characterized by a specific combination of requirements:

- *Cyclic operation:* all batches pass the plant in an identical time scheme. For *Strictly cyclic operation*, the time offset between the start of consecutive batches is always constant. For screening plants, the requirement of identical time schemes for all batches is mandatory in order to receive comparable analysis results.
- *Due dates:* the time scheme for the batch may be restricted, involving lower as well as upper bounds (time window constraints).
- *General precedence network structure:* the sequence of worksteps for the single batch includes points of split-up and synchronization as well as parallel branches.
- *Revisited resources:* along the process, the same resource may be visited several times by the same batch.
- *Overtaking:* batches will overlap in time. Worksteps for batch ρ may take place prior to other worksteps for previous batches, even on the same resource.
- *Deterministic workstep durations:* the time needed for each workstep is predefined and deterministic. Batch sizes and workstep recipies are fixed. Nevertheless, entities may keep resources allocated during additional waiting intervals that succeed the worksteps.
- *No buffers:* after a workstep on one resource is finished, the resource will not be released before the resource for the next workstep is allocated (*blocking*). Thus, a batch will normally allocate two resources simultaneously while being transferred. This may apply for some or all of the system's resources.
- *No preemption:* worksteps cannot be interrupted by other worksteps on the same resource.
- *Globally optimal solution:* we are interested in systematic approaches such that a globally optimal solution can be guaranteed.

The set of worksteps which is necessary for a screening task as well as the sequence of worksteps is specific for the substances and the tests to be performed. Thus, a new instance of the scheduling problem has to be solved for every screening task. This is performed in advance before the start of the screening run. The general objective of scheduling is to maximize throughput, i.e. to process as many batches per time as possible (*high throughput screening*) or, more generally, to finish a run of a fixed number of batches as fast as possible.

A large variety of cyclic scheduling problems have been considered in the literature. Major applications include manufacturing systems (e.g. Levner et al. 1997; Crama et al. 2000; Seo and Lee 2002; Hall et al. 2002; Karimi et al. 2004) traffic and transportation (e.g. Odijk 1996) or chemical batch plants (e.g. Pinto and Grossmann 1998; Alle et al. 2004; Pinto and Grossmann 1994; Shah et al. 1993). For the latter, additional degrees of freedom arise from recipies and material balances. For screening processes, material balancing does not play a role since materials are carried in the wells of microplates, which are always handled as one elementary unit.

Solution methods for cyclic scheduling are problem specific. If the sequence of all worksteps (*activities*) is fixed, the minimum possible cycle time (i.e. a schedule with maximum throughput) can be found using max-plus algebra (e.g. Lee 2000; Seo and Lee 2002) or algorithms with polynomial complexity (e.g. Lee and Posner 1997; Levner and Kats 1998). If batches do not overlap on any single resource, i.e. all activities of a batch are finished on a resource prior to the start of the first activity of the following batch (no overtaking), the problem of scheduling is reduced to fixing the optimal sequence and timing such that the single batch can be processed and repeated as fast as possible (e.g. Hall et al. 2002; Lee and Posner 1997; Levner and Kats 1998). Many of the cyclic scheduling methods in the literature address cycle shop problems (Middendorf and Timkovsky 2002), which do not cover the case of a general precedence network structure. Often, this is restricted further if resources are only visited once by each batch (flow shop) or revisiting of resources is limited to a single robot (e.g. Chen et al. 1998; Crama et al. 2000; Levner et al. 1997). The absence of buffers may reduce the complexity of the scheduling problem (Crama et al. 2000; Chen et al. 1998; Levner et al. 1997), but this is an additional requirement if buffers exist for some of the resources but not for all of them, as it is the case for screening plants. Another additional requirement is the demand for due dates resp. time window constraints (e.g. Crama et al. 2000; Levner and Kats 1998; Chen et al. 1998), which makes it necessary to set up additional constraints in the scheduling problem.

For many of these cyclic scheduling problems, an algorithm with polynomial complexity has not been found. Such problems may nevertheless be efficiently solved by formulating them as (mixed integer) optimization problems. Acceptable calculation times for globally optimal solutions can then be achieved by use of problem-specific formulations and branch-and-bound techniques (e.g. Chen et al. 1998; Roundy 1992; Seo and Lee 2002).

Although many of the approaches from the literature can be transferred to other applications, they do not meet all the requirements listed above. For strictly cyclic operation under these requirements, the maximum throughput problem has been solved in Mayer and Raisch (2004) for problem instances from the pharmaceutical industries. Several extensions, such as sequence dependent switching times or resources with multiple capacity, have also been studied (Mayer and Raisch 2003). The method presented there is not limited to screening plants but can be applied to any cyclically operated system with the same (or less) requirements.

This contribution provides a thorough discussion of the modeling and solution steps for the strictly cyclic maximum throughput problem. It then addresses the extension of the strictly cyclic case to a more general *hierarchical cyclic structure*, where cycles are nested in two levels. For specific task structures, this allows for considerably increased throughput.

This paper is arranged as follows: In Section 2, we present a timed discrete event model for cyclic processes with activities and resources. The constraints for the scheduling problem are derived. In Section 3, it is shown that the scheduling problem can be cast into a mixed integer linear program (MILP). A number of additional constraints are provided which allow to reduce computation time of the globally optimal solution.

In Section 4, the method is extended from the basic strictly cyclic case to the hierarchical cyclic case. It is shown that, by an appropriate modeling approach, the hierarchical cyclic problem can be formulated as a nonlinear mixed-integer program. A transformation to a mixed-integer linear program is presented, which again allows effective solution of the scheduling problem. In Section 5, both, the strictly cyclic and the hierarchical cyclic method, are illustrated by a small illustrating example and finally in Section 6, both methods are applied to a standard problem instance from high throughput screening.

2 Discrete event modeling of cyclic systems for scheduling

2.1 General setup

The general modeling approach for cyclic processes is described in Mayer and Raisch (2004), Mayer (2007) and will therefore only be outlined in this paper.

Using a system with m resources of capacity 1, a set of n worksteps (a *batch*) is executed a large number of times. The single batch represents one job or a fixed group of jobs. The worksteps (*activities*) for the batch are given in the batch definition, together with sequence and timing constraints. The duration needed for the workstep within any activity is predefined and deterministic. Also, the resource allocated by the activity during execution is well defined.

For the basic cyclic scheduling problem, a strictly cyclic operating scheme is applied. Batches are started repeatedly with a fixed time offset, called cycle time T. The activities of the batch as well as their sequence and timing are identical for all batches. Usually, the overall processing time of the single batch exceeds the time distance between the start of consecutive batches. Therefore batches may overlap, i.e. at any instant of time, several batches may be in process simultaneously.

The model for the cyclic process therefore consists of two ingredients: the *cycle time* T and the *single batch time scheme,* which is valid for all batches:

$$\begin{aligned} o_i &\in \mathbb{R} \quad \ldots \quad \text{time, when activity } i \text{ starts.} \\ r_i &\in \mathbb{R} \quad \ldots \quad \text{time, when activity } i \text{ ends,} \\ r_i &> o_i \\ i &= 1 \ldots n \end{aligned} \tag{1}$$

Each activity allocates precisely one resource, denoted by J_i.

$$\begin{aligned} J_i \quad &\ldots \quad \text{resource allocated by activity } i, \\ J_i &\in \{1 \ldots m\}. \end{aligned}$$

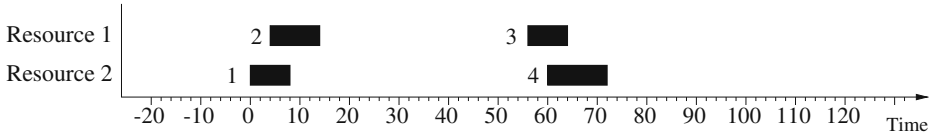


Fig. 1 Example for a single batch time scheme (Gantt chart)

Of course, any resource may be allocated subsequently by different activities. Within the single batch time scheme, the number of activities allocating resource j is

$$n_j = \sum_{i=1}^{n} \delta_{J_i j}\,, \quad j = 1 \ldots m\,, \tag{2}$$

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

Together with the cycle time T, the single batch time scheme defines the timing for the entire cyclic schedule: the times for the ρ-th batch are given by:

$$\begin{aligned} o_i^{(\rho)} &= o_i + \rho \cdot T \\ r_i^{(\rho)} &= r_i + \rho \cdot T, \quad \rho \in \mathbb{Z}, \quad i = 1 \ldots n. \end{aligned} \tag{3}$$

Note that the cyclic model does not account for the overall number of batches: in principle, the cyclic process could be repeated infinitely often. Figure 1 shows an example for a simple single batch time scheme involving $n = 4$ activities on $m = 2$ resources. The optimal strictly cyclic schedule for this single batch time scheme is pictured as a Gantt chart in Fig. 2.

The *batch definition* given by the user identifies the set of activities together with their resources J_i. If the values for the variables o_i and r_i, $i = 1 \ldots n$, were predetermined, the cycle time T would remain as the only degree of freedom. The problem would then be reduced to the problem of determining the optimal value for T and could be solved by simple algorithms in polynomial time. The optimal value for T could for example be found by successively excluding all forbidden intervals from the range of possible values for T. These intervals can be directly derived from the fact that two activities cannot allocate the same resource simultaneously. The number of forbidden intervals for n activities is $O(n^2)$.

However, in most cases the user will not fix the entire batch time scheme but will only provide a number of (usually) linear constraints for the set of possible values for

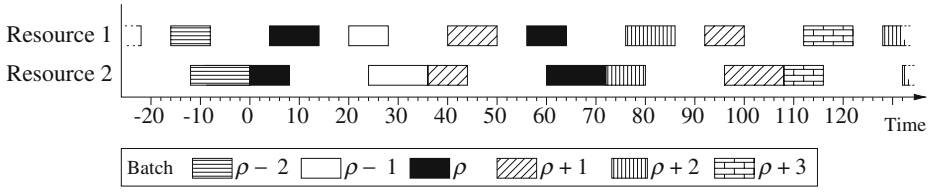


Fig. 2 Optimal cyclic schedule for the single batch time scheme from Fig. 1

the variables o_i and r_i. These constraints, together with a number of problem-intrinsic constraints, will be discussed in Section 2.2.

Often, the mathematical formulation can be substantially simplified by suitably parameterizing the time instants o_i and r_i. We represent the o_i and r_i as affine functions of K time variables $\theta_k \in \mathbb{R}_0^+$, $k = 1 \dots K$, where normally $K << 2n$:

$$\begin{aligned} o_i &= \chi_{i,0} + \sum_{k=1}^{K} (\chi_{i,k} \cdot \theta_k) \quad \dots \quad \text{time, when activity } i \text{ starts ,} \\ r_i &= \psi_{i,0} + \sum_{k=1}^{K} (\psi_{i,k} \cdot \theta_k) \quad \dots \quad \text{time, when activity } i \text{ ends.} \end{aligned} \tag{4}$$

Thus, the single batch time scheme is defined by the fixed parameters $\chi_{i,0}$, $\psi_{i,0}$, $\chi_{i,k}$, and $\psi_{i,k}$, $i = 1 \dots n$, $k = 1 \dots K$ and yet unknown variables θ_k, $k = 1 \dots K$.

The reparametrization (4) allows to reduce the number of degrees of freedom in the timing of the single batch. Often, a parametrization can be found such that the variables θ_k can simply be interpreted as an artificial delay that is either inserted between two activities of the single batch time scheme or within an activity just before the entity is transferred to a subsequent resource. The number K of variables θ_k depends on the sequence structure of the activities in the batch definition. It is always possible to find a parametrization (4) with $K < 2n$. A formal method to derive the parameters $\chi_{i,0}$, $\psi_{i,0}$, $\chi_{i,k}$ and $\psi_{i,k}$ from a raw graph description of the batch definition can be found in Mayer (2007).

2.2 Constraints

There are three types of constraints that have to be met by the cyclic schedule.

First, a number of user defined constraints for the single batch time scheme, i.e. the variables o_i and r_i, $i = 1 \dots n$, ensure

1. That activities last long enough to allow finishing of the required operations
2. That certain sequence constraints hold (for example, a batch may not be allowed to allocate the next resource before the activity in the previous resource is completed)
3. That certain time window constraints (e.g. maximum admissible durations for chemical reactions) hold.

After applying parametrization (4), these constraints for the variables o_i and r_i are represented by upper bounds for the time variables θ_k:

$$\theta_k \le \theta_{k,\max} \ , \ k = 1 \dots K \ , \ \theta_k \in \mathbb{R}_0^+ \tag{5}$$

and by P additional linear constraints of the form

$$\sum_{k=1}^{K} (\kappa_{p,k} \cdot \theta_k) \le \vartheta_p \ , \ p = 1 \dots P. \tag{6}$$

Second, there is the problem intrinsic constraint that the cycle time T can never be smaller than the sum of activity durations on any resource during the single batch time scheme:

$$T \geq \sum_{i=1}^{n} (r_i - o_i)\delta_{J_i j}\,, \quad j = 1 \ldots m. \tag{7}$$

Substituting Eq. 4 into Eq. 7 results in

$$\gamma_{j,0} + \sum_{k=1}^{K} (\gamma_{j,k} \cdot \theta_k) - T \leq 0\,, \quad j = 1 \ldots m\,, \tag{8}$$

where

$$\gamma_{j,0} = \sum_{i=1}^{n} (\psi_{i,0} - \chi_{i,0})\delta_{J_i j}$$

$$\gamma_{j,k} = \sum_{i=1}^{n} (\psi_{i,k} - \chi_{i,k})\delta_{J_i j}.$$

Third, there is the requirement that no two activities are allowed to allocate the same resource simultaneously (*disjunctive constraints*).

This requirement is met if the mutual exclusion condition

$$o_{i_1}^{(\rho_2)} \geq r_{i_2}^{(\rho_1)} \;\; \text{XOR} \;\; o_{i_2}^{(\rho_1)} \geq r_{i_1}^{(\rho_2)} \tag{9}$$

holds for any pair of activities using the same resource, i.e. for all $i_1, i_2 \in \{1 \ldots n\}$, $\rho_1, \rho_2 \in \mathbb{Z}, i_1 < i_2, J_{i_1} = J_{i_2}$. Due to symmetry in Eq. 9, it is sufficient to demand Eq. 9 for $i_1 < i_2$. The special case $i_1 = i_2, \rho_1 \neq \rho_2$ is already ensured by Eq. 7.

For strictly cyclic operation, it has been shown in Mayer and Raisch (2004) that the following condition correctly models the constraint (9):

$$\forall (i_1, i_2),\; i_1 < i_2,\; J_{i_1} = J_{i_2} \;\; \exists\, z_{(i_1,i_2)} \in \mathbb{Z} \text{ s.t.}$$

$$z_{(i_1,i_2)} \cdot T - (o_{i_2} - r_{i_1}) \leq 0 \tag{10}$$

$$\left(z_{(i_1,i_2)} + 1 \right) \cdot T - (r_{i_2} - o_{i_1}) \geq 0. \tag{11}$$

Thus, an integer variable $z_{(i_1,i_2)}$ is introduced for each pair of activities within the single batch time scheme that use the same resource. For $o_{i_2} > o_{i_1}$ and $z_{(i_1,i_2)} \geq 0$, this allows for the following physical interpretation for the integer variable $z_{(i_1,i_2)}$:

between activity i_1 for a batch ρ_1 and activity i_2 for the same batch, the resource is used for exactly $z_{(i_1,i_2)}$ activities of type i_1 of subsequent batches, i.e. for batches $\rho_2 \in \{\rho_1 + 1 \ldots \rho_1 + z_{(i_1,i_2)}\}$.

By introducing the integer variables, the infinite number of XOR conditions (9) is replaced by a finite number of requirements of the form (10), (11).

In order to formulate the scheduling problem as a mixed integer optimization problem with compact notation, some abbreviations are introduced.

Each possible pair of indices (i_1, i_2), $i_2 > i_1$, $J_{i_1} = J_{i_2}$, is denoted by a single number ι,

$$\iota = 1 \ldots \iota_{\max}\ , \quad \iota_{\max} = \sum_{j=1}^{m} \frac{n_j(n_j - 1)}{2}\ . \tag{12}$$

Hence, each value for ι signifies a pair of activities (within the single batch time scheme) using the same resource.

Substituting Eq. 4 into Eqs. 10 and 11 results in

$$z_\iota \cdot T - \ v_{\iota,0} - \sum_{k=1}^{K} (v_{\iota,k} \cdot \theta_k) \leq 0 \tag{13}$$

$$(z_\iota + 1) \cdot T - w_{\iota,0} - \sum_{k=1}^{K} (w_{\iota,k} \cdot \theta_k) \geq 0 \tag{14}$$

with the following abbreviations:

$$v_{\iota,k} = \chi_{i_2,k} - \psi_{i_1,k} \tag{15}$$

$$w_{\iota,k} = \psi_{i_2,k} - \chi_{i_1,k} \tag{16}$$

$$v_{\iota,0} = \chi_{i_2,0} - \psi_{i_1,0} \tag{17}$$

$$w_{\iota,0} = \psi_{i_2,0} - \chi_{i_1,0}. \tag{18}$$

Constraints (13) and (14) have to hold for all $\iota = 1 \ldots \iota_{\max}$.

3 Optimization problem

The objective of the scheduling problem is to maximize throughput. For strictly cyclic processes, this is equivalent to minimizing the cycle time T. Formulating the scheduling problem as an optimization problem, we have to take the cycle time T as the objective function to be minimized under the constraints given by Eqs. 13 and 14 as well as Eqs. 5, 6, and 8. The search space for the optimization problem is defined by the following variables:

- Cycle time $T \in \mathbb{R}^+$,
- Time variables $\theta_k \in \mathbb{R}_0^+$,
- Integer variables $z_\iota \in \mathbb{Z}$.

Hence, the basic cyclic scheduling problem can be written as the following mixed integer nonlinear program:

Min T subject to

$$z_\iota \cdot T - \ v_{\iota,0} - \sum_{k=1}^{K} (v_{\iota,k} \cdot \theta_k) \leq 0 \qquad \text{for } \iota = 1 \ldots \iota_{\max} \tag{19a}$$

$$(z_\iota + 1) \cdot T - w_{\iota,0} - \sum_{k=1}^{K} (w_{\iota,k} \cdot \theta_k) \geq 0 \qquad \text{for } \iota = 1 \ldots \iota_{\max} \tag{19b}$$

$$\theta_k \leq \theta_{k,\max} \qquad \text{for } k = 1 \ldots K \tag{19c}$$

$$\sum_{k=1}^{K} (\kappa_{p,k} \cdot \theta_k) \leq \vartheta_p \qquad \text{for } p = 1 \ldots P \tag{19d}$$

$$\gamma_{j,0} + \sum_{k=1}^{K} (\gamma_{j,k} \cdot \theta_k) - T \leq 0 \qquad \text{for } j = 1 \ldots m \tag{19e}$$

In general, problem (19a) to (19e) has more than one unique globally optimal solution. Of course, any of its globally optimal solutions constitutes a throughput-optimal cyclic schedule for the underlying problem instance. However, non-convex mixed-integer optimization problems, especially of this size, are in general very difficult to solve in a globally optimal way. Fortunately, it turns out that Eqs. 19a to 19e can be reformulated in mixed-integer linear form. Along the way, stricter bounds for some variables are derived, which reduces computation time.

3.1 Strengthening the formulation

For batches, where some resources have to be shared by a large number of activities, the optimization problem (19a) to (19e) may become very large (for n_j activities on a resource j, the number of pairs and therefore the number $\iota_{\max}$ of integer variables z is $\frac{n_j(n_j-1)}{2}$, see Eq. 12). Therefore it can be helpful to add additional bounds for the variables in Eqs. 19a to 19e, hence reducing the computation time for the optimization algorithm.

A lower bound for the cycle time T can be derived from the fact that if each single activity is finished as fast as possible and the busiest resource is allocated non-stop, no further reduction of cycle time is possible:

$$T \geq T_{\min} = \max_j \left(\min_{\theta_1 \ldots \theta_K} \sum_{i=1}^{n} (r_i - o_i) \delta_{J_i j} \right). \tag{20}$$

An upper bound $T_{\max}$ for the cycle time T can be prescribed by the user. Alternatively, such a bound can be deduced from the trivial case in which no batch is started before the previous batch is finished:

$$T \leq T_{\max} = \max_{\theta_1 \ldots \theta_K} \left(\max_i r_i - \min_i o_i \right) \quad . \tag{21}$$

A tighter bound can be derived if, in addition to Eq. 21, the single batch is required to be finished as fast as possible. This corresponds to solving the optimization problem (19a) to (19e) with $z_\iota \in \{-1, 0\}$, which usually can be solved significantly faster than the optimization problem with nominally unbounded variables $z \in \mathbb{Z}$ and always has a solution if Eqs. 19a to 19e have a solution. Note that this upper bound for T reduces the feasible region. Nevertheless, the feasible region does not become empty. Since the objective is to minimize T, at least one globally optimal solution is preserved in the formulation.

In addition to the bounds for T, lower and upper bounds for the integer variables z_ι can be introduced as follows:

$$z_{\iota,\min} \leq z_\iota \leq z_{\iota,\max}, \quad \iota = 1 \ldots \iota_{\max} \ , \tag{22}$$

where $z_{\iota,\min}$ and $z_{\iota,\max}$ are retrieved from solving the relaxation of Eqs. 19a to 19e, together with Eqs. 20 and 21 with the objective of minimizing respectively maximizing z_ι (the relaxation of an optimization problem is obtained by allowing the integer variables to take any value from $\mathbb{R}$). A faster, but less strict way to find lower and upper bounds $z_{\iota,\min}$, $z_{\iota,\max}$ is as follows:

$$z_{\iota,\min} := \begin{cases} \lceil \frac{\underline{W}_\iota}{T_{\min}} \rceil - 1 & \text{for } \underline{W}_\iota < 0 \\ \lceil \frac{\underline{W}_\iota}{T_{\max}} \rceil - 1 & \text{for } \underline{W}_\iota \geq 0 \end{cases}$$

$$z_{\iota,\max} := \begin{cases} \lfloor \frac{\bar{V}_\iota}{T_{\max}} \rfloor & \text{for } \bar{V}_\iota \leq 0 \\ \lfloor \frac{\bar{V}_\iota}{T_{\min}} \rfloor & \text{for } \bar{V}_\iota > 0 \end{cases}$$

$$\underline{W}_\iota = \min_{\theta_1 \ldots \theta_K} \left(w_{\iota,0} + \sum_{k=1}^{K} w_{\iota,k} \cdot \theta_k \right)$$

$$\bar{V}_\iota = \max_{\theta_1 \ldots \theta_K} \left(v_{\iota,0} + \sum_{k=1}^{K} v_{\iota,k} \cdot \theta_k \right).$$

($\lfloor x \rfloor$ denotes the floor-function, i.e. the largest integer number that is less or equal to x. $\lceil x \rceil$ denotes the ceil-function, i.e. the smallest integer number that is greater or equal to x). These conditions can be derived directly from Eq. 19a resp. Eq. 19b, together with Eqs. 20 and 21.

Conditions (22) can be interpreted as Gomory cuts (Bertsimas and Weismantel 2005) in the original system. Introducing such cuts does not reduce the feasible region of the optimization problem. In other words, the introduction of the additional

bounds only removes values for the integer variables, for which at least one of conditions (19a) to (19e) is not satisfied.

3.2 Transformation to MILP

The nonlinear mixed-integer optimization problem (19a) to (19e), together with the bounds derived in Section 3.1, can be transformed to a linear problem: the feasible region is reparameterized by introducing

$$\bar{T} := \frac{1}{T} \quad , \quad \bar{\theta}_k := \frac{\theta_k}{T}, \; k = 1 \ldots K. \tag{23}$$

This is an exact reformulation of the problem. The set of globally optimal solutions remains unchanged. Note that the property of nonlinearity has been eliminated without paying the cost of additional variables. The resulting linear reformulation of the optimization problem reads as follows:

Max $\bar{T}$ subject to

$$z_\iota - \; v_{\iota,0} \cdot \bar{T} - \sum_{k=1}^{K} \left(v_{\iota,k} \cdot \bar{\theta}_k \right) \leq 0 \qquad \text{for } \iota = 1 \ldots \iota_{\max} \tag{24a}$$

$$z_\iota + 1 - w_{\iota,0} \cdot \bar{T} - \sum_{k=1}^{K} \left(w_{\iota,k} \cdot \bar{\theta}_k \right) \geq 0 \qquad \text{for } \iota = 1 \ldots \iota_{\max} \tag{24b}$$

$$z_{\iota,\min} \leq z_\iota \leq z_{\iota,\max} \qquad \text{for } \iota = 1 \ldots \iota_{\max} \tag{24c}$$

$$\bar{\theta}_k \leq \theta_{k,\max} \cdot \bar{T} \qquad \text{for } k = 1 \ldots K \tag{24d}$$

$$\sum_{k=1}^{K} \left(\kappa_{p,k} \cdot \bar{\theta}_k \right) \leq \vartheta_p \cdot \bar{T} \qquad \text{for } p = 1 \ldots P \tag{24e}$$

$$\frac{1}{T_{\max}} \leq \bar{T} \leq \frac{1}{T_{\min}} \tag{24f}$$

$$\gamma_{j,0} \cdot \bar{T} + \sum_{k=1}^{K} \left(\gamma_{j,k} \cdot \bar{\theta}_k \right) - 1 \leq 0 \qquad \text{for } j = 1 \ldots m \tag{24g}$$

Equations 24a to 24g state a mixed integer linear program (MILP). This optimization problem can be efficiently solved using standard methods of mathematical programming (e.g. Branch and Cut). For a comprehensive discussion of solution techniques for MILPs see Bertsimas and Weismantel (2005).

4 Extension: hierarchical cyclic structure

4.1 General framework

The scheduling problem for strictly cyclic operation can be solved in a globally optimal way by solving the mixed-integer linear optimization problem presented in Section 3. However, a strictly cyclic timetable, i.e. a strictly cyclic timetable with a constant time offset between all consecutive entities is often unnecessarily restrictive and a cyclic requirement may be sufficient. For a specific class of high throughput screening tasks, a two-level hierarchical nesting of cycles allows for a throughput rate higher than in the strictly cyclic case.

Again, the task is to process a (theoretically infinite) number of entities, which all need the same set of worksteps (activities). In the hierarchical cyclic framework, we use the term 'job' for such an entity. Together with their timing, the corresponding set of activities is called the 'job time scheme'.

For a hierarchical cyclic structure, a finite number of such jobs are grouped together to form one batch. Within a batch, these individual jobs are also processed in a cyclic scheme ('inner cycles'). The batches itself are, again, periodically repeated in a strictly cyclic scheme ('outer cycles'). For the outer cyclic scheme the rules are the same as for the strictly cyclic case described in Section 2:

- The time distance between the start of consecutive batches is always constant.
- The processing time scheme is identical for all batches.
- The cyclic time scheme does allow for infinite repetition.

For the inner cyclic scheme the following rules apply:

- The time distance between the start of consecutive jobs is constant.
- All individual jobs have to be processed in the same time scheme ('job time scheme').

Figure 3 shows a schedule with hierarchical cyclic structure for the single batch time scheme from Fig. 1. The schedule shown in Fig. 3 is said to be four-periodic, i.e. the time offset between an activity for entity i and its counterpart for entity $i+4$ is constant for all entities and all activities. Compared to the strictly cyclic solution from Fig. 2, this four-periodic schedule obviously allows for an improved throughput rate.

If the number of jobs in the inner cycle, i.e. the number of jobs for one batch is fixed a-priori, the problem is, again, reduced to the strictly cyclic scheduling problem

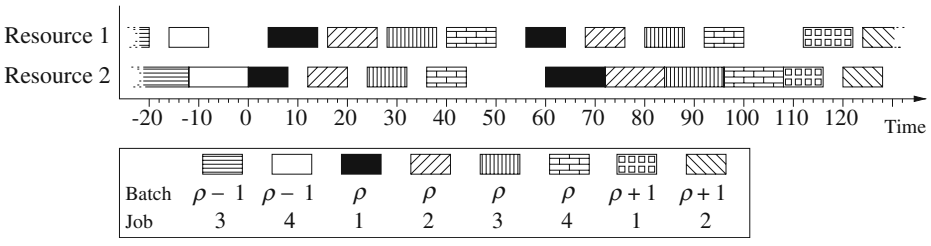


Fig. 3 Schedule with hierarchical cyclic structure (four-periodic)

described in Section 2: the jobs in the inner cycle can be grouped into one batch, where the parametrization (4), together with the linear constraints (5) and (6) ensure identical time distance between the start of the jobs and identical processing time schemes for all jobs.

However, the optimal number of jobs per batch is usually not known a-priori, but has to be determined simultaneously with the optimal values for the other variables (i.e. the single batch time scheme and the cycle time T). This is done by formulating the scheduling problem for the hierarchical cyclic structure as an optimization problem which, in addition to the timing of the single batch time scheme and the cycle time for the outer cycle, also includes the number of jobs within the single batch time scheme as a variable $Y \in \mathbb{N}$.

4.2 Batch time scheme

In our hierarchical setting, the set of activities in the single batch time scheme is not fixed but depends on the value of the decision variable Y.

The time scheme for the individual jobs within the single batch is identical for all jobs. It consists of n^* activities, which are described by their starting times and end times, parameterized by K^* time variables $\theta_k \in \mathbb{R}_0^+$:

$$\begin{aligned} o_i^* &= \chi_{i,0}^* + \sum_{k=1}^{K^*} \left(\chi_{i,k}^* \cdot \theta_k\right) \quad \ldots \quad \text{time, when activity } i \text{ starts ,} \\ r_i^* &= \psi_{i,0}^* + \sum_{k=1}^{K^*} \left(\psi_{i,k}^* \cdot \theta_k\right) \quad \ldots \quad \text{time, when activity } i \text{ ends.} \end{aligned} \tag{25}$$

$$\begin{aligned} r_i^* &> o_i^* \\ i &= 1 \ldots n^*. \end{aligned} \tag{26}$$

Each activity i allocates one resource $J_i^* \in \{1 \ldots m\}$.

Again, user-defined constraints ensure that the activity durations allow for finishing the required operations and that the sequence and time window constraints for the individual jobs within the job time scheme hold. These constraints are represented by upper bounds for the time variables θ_k

$$\theta_k \leq \theta_{k,\max}^* \,, \quad k = 1 \ldots K^* \,, \quad \theta_k \in \mathbb{R}_0^+ \tag{27}$$

and by P additional linear constraints of the form

$$\sum_{k=1}^{K^*} \left(\kappa_{p,k}^* \cdot \theta_k\right) \leq \vartheta_p^* \,, \quad p = 1 \ldots P. \tag{28}$$

It is natural to assume that an upper bound $Y_{\max}$ for the variable Y is prescribed by the user.

Thus, the single batch time scheme consists of at most $Y_{\max}$ jobs and $n = Y_{\max} \cdot n^*$ activities. Within the batch, jobs are started cyclically with cycle time T^*. As all jobs

follow the time scheme given in Eq. 25, the activity start and end times for job $h \in \{1 \dots Y_{\max}\}$ are

$$o_i^{*(h)} = \chi_{i,0}^* + \sum_{k=1}^{K^*} \left(\chi_{i,k}^* \cdot \theta_k\right) + (h-1) \cdot T^* \quad \text{for } i = 1 \dots n^*$$

$$r_i^{*(h)} = \psi_{i,0}^* + \sum_{k=1}^{K^*} \left(\psi_{i,k}^* \cdot \theta_k\right) + (h-1) \cdot T^* \quad \text{for } i = 1 \dots n^* \tag{29}$$

As the cycle time of the inner cycle T^* is a decision variable that is part of the single batch time scheme, notation $\theta_K := T^*$, $K = K^* + 1$ will be used. Therefore the constraints for the variables $\theta_k \in \mathbb{R}_0^+, k = 1 \dots K$, are

$$\theta_k \le \theta_{k,\max} \ , \ k = 1 \dots K \ , \ \theta_k \tag{30}$$

and

$$\sum_{k=1}^{K} (\kappa_{p,k} \cdot \theta_k) \le \vartheta_p \ , \ p = 1 \dots P \ , \tag{31}$$

where

$$\theta_{k,\max} = \theta_{k,\max}^* \ , \quad k = 1 \dots K^* \tag{32}$$

$$\theta_{k,\max} = \infty \ , \quad k = K \tag{33}$$

$$\kappa_{p,k} = \kappa_{p,k}^* \ , \quad p = 1 \dots P \ , \ k = 1 \dots K^* \tag{34}$$

$$\kappa_{p,k} = 0 \ , \quad p = 1 \dots P \ , \ k = K \tag{35}$$

$$\vartheta_p = \vartheta_p^* \ , \quad p = 1 \dots P. \tag{36}$$

To simplify notation, the activities of all $Y_{\max}$ jobs are subsumed in one set and are serially numbered by a common index i. Thus, the single batch time scheme for the hierarchical cyclic structure is defined as follows:

$$o_i = \chi_{i,0} + \sum_{k=1}^{K} (\chi_{i,k} \cdot \theta_k) \ , \quad i = 1 \dots n$$

$$r_i = \psi_{i,0} + \sum_{k=1}^{K} (\psi_{i,k} \cdot \theta_k) \ , \quad i = 1 \dots n \tag{37}$$

$$\psi_{i,0} = \psi^*_{i^*,0}\ , \quad i = 1 \ldots n$$

$$\chi_{i,0} = \chi^*_{i^*,0}\ , \quad i = 1 \ldots n$$

$$\psi_{i,k} = \psi^*_{i^*,k}\ , \quad i = 1 \ldots n\ , \quad k = 1 \ldots K^*$$

$$\psi_{i,k} = \lfloor (i-1)\,/\,n^* \rfloor\ , \quad i = 1 \ldots n\ , \quad k = K = K^* + 1$$

$$\chi_{i,k} = \chi^*_{i^*,k}\ , \quad i = 1 \ldots n\ , \quad k = 1 \ldots K^*$$

$$\chi_{i,k} = \lfloor (i-1)\,/\,n^* \rfloor\ , \quad i = 1 \ldots n\ , \quad k = K = K^* + 1$$

where $n = n^* \cdot Y_{\max}$ and $i^* = (i-1) \bmod n^* + 1$.

The resources allocated by the activities are

$$J_i = J^*_{i^*}\ , \quad i = 1 \ldots n. \tag{38}$$

As only the first Y jobs are part of the single batch time scheme, only activities i, $y_i = \lfloor (i-1)\,/\,n^* \rfloor + 1 \leq Y$ take place:

$$Y \geq y_i \ \ldots \text{ activity } i \text{ is part of the single batch time scheme (='effective')} \tag{39}$$

$$Y < y_i \ \ldots \text{ activity } i \text{ is not part of the single batch time scheme (='ineffective').} \tag{40}$$

4.3 Constraints

Since the outer cyclic structure is identical to the standard strictly cyclic case described in Section 2, the constraints for the optimization problem are derived in the same way. However, the set of activities in the single batch time scheme is not fixed but depends on the number of jobs per batch (Y). The set of constraints for the optimization problem therefore depends on the value of the decision variable Y.

First, a lower bound for the cycle time T can be set up equivalently to Constraint (7). For the hierarchical cyclic case, this constraint needs to account for the number of effective activities in the single batch time scheme: for a certain value of Y, only activities i that are effective (i.e. $y_i \leq Y$) contribute to the sum of activity durations in the single batch time scheme. Hence, a lower bound for the cycle time T is

$$T \geq \sum_{i=1}^{n} (r_i - o_i)\delta_{J_i j}\Delta_{y_i Y}\ , \quad j = 1 \ldots m \tag{41}$$

$$\text{where } \Delta_{y_i Y} = \begin{cases} 0 & \text{for } y_i > Y \\ 1 & \text{for } y_i \leq Y. \end{cases}$$

Substituting Eq. 4 into Eq. 41 results in

$$\gamma_{j,0,Y} + \sum_{k=1}^{K} (\gamma_{j,k,Y} \cdot \theta_k) - T \leq 0\ , \quad j = 1 \ldots m\ , \tag{42}$$

where

$$\gamma_{j,0,Y} = \sum_{i=1}^{n} (\psi_{i,0} - \chi_{i,0}) \delta_{J_i j} \Delta_{y_i Y}$$

$$\gamma_{j,k,Y} = \sum_{i=1}^{n} (\psi_{i,k} - \chi_{i,k}) \delta_{J_i j} \Delta_{y_i Y}. \tag{43}$$

This set of constraints depends on the value of the decision variable Y. In order to have a fixed number of constraints, Eq. 42 can be transformed to the following set of 'OR' statements:

$$\gamma_{j,0,Y'} + \sum_{k=1}^{K} (\gamma_{j,k,Y'} \cdot \theta_k) - T \leq 0 \quad \text{OR} \quad Y \neq Y' \tag{44}$$

$$\text{for } j = 1 \ldots m \, , \, Y' = 1 \ldots Y_{\max}.$$

Due to the definition of $\gamma_{j,0,Y}$ and $\gamma_{j,k,Y}$, Eqs. 43, 37 together with Eq. 26, the following always holds for $Y1 \geq Y2$:

$$\gamma_{j,0,Y1} + \sum_{k=1}^{K} (\gamma_{j,k,Y1} \cdot \theta_k) \geq \gamma_{j,0,Y2} + \sum_{k=1}^{K} (\gamma_{j,k,Y2} \cdot \theta_k). \tag{45}$$

Therefore, Eq. 44 can be finally given as follows:

$$\gamma_{j,0,Y'} + \sum_{k=1}^{K} (\gamma_{j,k,Y'} \cdot \theta_k) - T \leq 0 \quad \text{OR} \quad Y \leq Y' - 1 \tag{46}$$

$$\text{for } j = 1 \ldots m \, , \, Y' = 1 \ldots Y_{\max}.$$

The second set of constraints for the optimization problem is given by the linear constraints on the variables θ_k, $k = 1 \ldots K$, given in Eqs. 30 and 31.

The third type of constraints are the disjunctive constraints (9) that have to be met for all pairs of effective activities belonging to the single batch time scheme. Therefore, Eqs. 10 and 11 have to hold for all pairs (i_1, i_2) for which

$$y_{i_1} \leq Y \;\text{ and }\; y_{i_2} \leq Y. \tag{47}$$

Accordingly, Constraints (13), (14) have to come into effect if Eq. 47 holds for the indices i_1, i_2 associated to the index ι. A variable g_ι is used to formulate this condition: constraints (13), (14) have to hold if

$$Y > g_\iota \, , \tag{48}$$

$$\text{with } g_\iota = \max(y_{i_1}, y_{i_2}) - 1. \tag{49}$$

In other words,

$$z_\iota \cdot T - \ v_{\iota,0} - \sum_{k=1}^{K} (v_{\iota,k} \cdot \theta_k) \leq 0 \quad \text{OR} \quad Y \leq g_\iota \ , \quad \iota = 1 \ldots \iota_{\max} \tag{50}$$

$$(z_\iota + 1) \cdot T - w_{\iota,0} - \sum_{k=1}^{K} (w_{\iota,k} \cdot \theta_k) \geq 0 \quad \text{OR} \quad Y \leq g_\iota \ , \quad \iota = 1 \ldots \iota_{\max} \ , \tag{51}$$

where $\iota_{\max}$ has to be determined for the case $Y = Y_{\max}$.

4.4 Optimization problem

The objective is to process a maximum number of jobs per time. As Y jobs are grouped into one batch, the objective function for throughput maximization is to minimize the cycle time of the outer cycles divided by the number of jobs per batch:

$$\text{minimize } \frac{T}{Y}.$$

This term represents the inverse of the job throughput, i.e. the mean time offset between consecutive jobs.

This function has to be minimized subject to constraints (30), (31), (46), (50), and (51).

Lower and upper bounds for the cycle time T as well as for the integer variables z_ι can be included in the optimization problem as detailed for the model in Section 3.1.

Additionally, a lower and an upper bound can be prescribed for the objective function term:

$$T_{\text{lo}} \leq \frac{T}{Y} \leq T_{\text{up}}. \tag{52}$$

The lower bound T_{lo} can be determined by evaluating Eq. 20 for $Y = 1$, i.e. for one job:

$$T_{\text{lo}} = \max_j \left(\min_{\theta_1 \ldots \theta_{K^*}} \sum_{i=1}^{n^*} \left(r_i^* - o_i^* \right) \delta_{J_i^* j} \right). \tag{53}$$

An upper bound T_{up} can be derived by solving the strictly cyclic scheduling problem with one single job per batch.

To sum up, the following optimization problem describes the cyclic scheduling problem with hierarchical cyclic structure:

$$(T \in \mathbb{R}^+, \ \theta_k \in \mathbb{R}_0^+, \ z_\iota \in \mathbb{Z}, \ Y \in \mathbb{N}, Y > 0)$$

Min $\frac{T}{Y}$ subject to

$$z_\iota \cdot T - v_{\iota,0} - \sum_{k=1}^{K}(v_{\iota,k} \cdot \theta_k) \leq 0 \quad \text{OR} \quad Y \leq g_\iota \qquad \text{for } \iota = 1 \ldots \iota_{\max} \tag{54a}$$

$$(z_\iota + 1) \cdot T - w_{\iota,0} - \sum_{k=1}^{K}(w_{\iota,k} \cdot \theta_k) \geq 0 \quad \text{OR} \quad Y \leq g_\iota \qquad \text{for } \iota = 1 \ldots \iota_{\max} \tag{54b}$$

$$z_{\iota,\min} \leq z_\iota \leq z_{\iota,\max} \qquad \text{for } \iota = 1 \ldots \iota_{\max} \tag{54c}$$

$$\theta_k \leq \theta_{k,\max} \qquad \text{for } k = 1 \ldots K \tag{54d}$$

$$\sum_{k=1}^{K}(\kappa_{p,k} \cdot \theta_k) \leq \vartheta_p \qquad \text{for } p = 1 \ldots P \tag{54e}$$

$$T_{\min} \leq T \leq T_{\max} \tag{54f}$$

$$T_{\text{lo}} \leq \frac{T}{Y} \leq T_{\text{up}} \tag{54g}$$

$$\gamma_{j,0,Y'} + \sum_{k=1}^{K}(\gamma_{j,k,Y'} \cdot \theta_k) - T \leq 0 \quad \text{OR} \quad Y \leq Y' - 1 \qquad \text{for } j = 1 \ldots m\,, \quad Y' = 1 \ldots Y_{\max} \tag{54h}$$

$$Y \leq Y_{\max} \tag{54i}$$

This problem can once again be transformed into a linear one, albeit we need to expend some more effort because of the nonlinear objective function; it should also be noted that an increase in the number of values allowed for Y influences the size of the resulting linear problem and makes it worse from a complexity theoretic point of view. However, in the typically occurring instances, Y has a small range.

The linear reformulation can be split into 5 steps.

1. Reformulation of the objective function
   Instead of minimizing T/Y we can maximize Y/T, since $T, Y > 0$. This obviously changes the objective function value, but any optimal solution for the minimization problem will be optimal for the maximization problem, and vice versa.
2. Reparametrization of T
   As in the previous model we will reparameterize T by substituting $\bar{T} = \frac{1}{T}$, and $\bar{\theta}_k = \frac{\theta_k}{T}$ for $k = 1, \ldots, K$. This leads to a set of constraints that is of the same form as in Eqs. 24a to 24g, except for the disjunctions in Eqs. 54a, 54b and 54h.
3. Removal of Eq. 54h
   The inequalities (54h) can be subsumed under Eqs. 54a and 54b, by introducing $m \cdot Y_{\max}$ new indices $\iota_1, \ldots, \iota_{m \cdot Y_{\max}}$ after $\iota_{\max}$, fixing the associated variables z_{ι_i} to 0, setting the new coefficients $v_{\iota_i} = 0$ and $w_{\iota_i,\{0,k\}} = \gamma_{[(i-1) \bmod m]+1,\{0,k\},\lfloor (i-1)/m \rfloor + 1}$, and limiting Y by $g_{\iota_i} = (i-1) \bmod m$ for $Y' = 1, \ldots, Y_{\max}$. The associated inequalities of type (54a) are then no restriction, while those of type (54b) take the place of Eq. 54h.

4. Modeling of the disjunctions (54a) and (54b)
The disjunctions (54a) and (54b) with enlarged $\iota_{\max}$, i.e. including Eq. 54h from the application of step 3, can be modeled by introducing decision variables $x_\iota \in \{0, 1\}$ for $\iota = 1, \ldots, \iota_{\max}$, which are to be equal to 1 whenever $Y \le g_\iota$, and 0 otherwise. We can then model the disjunctions (54a) and (54b), already rewritten according to the reparametrization step above, by a family of 3 inequalities for each $\iota = 1, \ldots, \iota_{\max}$:

$$Y - g_\iota \le U_\iota^Y \cdot x_\iota \quad \text{with constant} \quad U_\iota^Y \ge Y_{\max} - g_\iota,$$

$$z_\iota - v_{\iota,0} \cdot \bar{T} - \sum_{k=1}^{K} v_{\iota,k} \cdot \bar{\theta_k} \le (1 - x_\iota) \cdot U_\iota^z$$

with constant

$$U_\iota^z \ge z_{\iota,\max} + |v_{\iota,0}| \frac{1}{T_{\min}} + \sum_{k=1}^{K} |v_{\iota,k}| \cdot \theta_{k,\max} \frac{1}{T_{\min}},$$

and finally

$$z_\iota + 1 - w_{\iota,0} \cdot \bar{T} - \sum_{k=1}^{K} w_{\iota,k} \cdot \bar{\theta_k} \ge (1 - x_\iota) \cdot L_\iota^z$$

with constant

$$L_\iota^z \le z_{\iota,\min} + 1 - |w_{\iota,0}| \frac{1}{T_{\min}} - \sum_{k=1}^{K} |w_{\iota,k}| \cdot \theta_{k,\max} \frac{1}{T_{\min}}.$$

In this formulation the choice of the three constants U_ι^Y, U_ι^z, and L_ι^z ensures that the respective inequality is not a restriction whenever the decision variable x_ι is not forcing the right hand side to 0.

5. Linear formulation of the objective $\max Y \cdot \bar{T}$
For each possible value q of Y in the range of $1, \ldots, Y_{\max}$ we introduce a binary variable $Y_{\neq q}$, which is 1 whenever Y does not attain the value q:

$$Y_{\neq q} \ge (Y - q)/(Y_{\max} - 1) \qquad q = 1, \ldots, Y_{\max}$$

$$Y_{\neq q} \ge (q - Y)/(Y_{\max} - 1) \qquad q = 1, \ldots, Y_{\max}$$

$$\sum_{q=1}^{Y_{\max}} Y_{\neq q} = Y_{\max} - 1.$$

Then we can replace the objective $\max Y \cdot \bar{T}$ by $\max \gamma$, where γ is a variable modeling the supremum of $Y \cdot \bar{T}$ over the feasible region:

$$\gamma \le q \cdot \bar{T} + M \cdot Y_{\neq q} \qquad q = 1, \ldots, Y_{\max}$$

where the constant $M \ge Y_{\max} \cdot \frac{1}{T_{\min}}$, ensures that only one of these restrictions is binding.

After applying steps 1 to 5, we obtain a linear reformulation of the problem that has the same set of globally optimal solutions. Nevertheless, this time, the advantage of linearity comes at the cost of $\iota_{\max}$ additional binary variables x_ι, $Y_{\max}$ additional binary variables $Y_{\neq q}$, and the additional variable $\gamma \in \mathbb{R}^+$. The linear reformulation finally reads as follows:

Max γ subject to

$$\gamma \leq q \cdot \bar{T} + M \cdot Y_{\neq q} \qquad \text{for } q = 1 \ldots Y_{\max} \tag{55a}$$

$$Y_{\neq q} \geq (Y - q)/(Y_{\max} - 1) \qquad \text{for } q = 1 \ldots Y_{\max} \tag{55b}$$

$$Y_{\neq q} \geq (q - Y)/(Y_{\max} - 1) \qquad \text{for } q = 1 \ldots Y_{\max} \tag{55c}$$

$$\sum_{q=1}^{Y_{\max}} Y_{\neq q} = Y_{\max} - 1 \tag{55d}$$

$$Y - g_\iota \leq U_\iota^Y \cdot x_\iota \qquad \text{for } \iota = 1 \ldots \iota_{\max} \tag{55e}$$

$$z_\iota - v_{\iota,0} \cdot \bar{T} - \sum_{k=1}^{K} v_{\iota,k} \cdot \bar{\theta}_k \leq (1 - x_\iota) \cdot U_\iota^z \qquad \text{for } \iota = 1 \ldots \iota_{\max} \tag{55f}$$

$$z_\iota + 1 - w_{\iota,0} \cdot \bar{T} - \sum_{k=1}^{K} w_{\iota,k} \cdot \bar{\theta}_k \geq (1 - x_\iota) \cdot L_\iota^z \qquad \text{for } \iota = 1 \ldots \iota_{\max} \tag{55g}$$

$$z_{\iota,\min} \leq z_\iota \leq z_{\iota,\max} \qquad \text{for } \iota = 1 \ldots \iota_{\max} \tag{55h}$$

$$\bar{\theta}_k \leq \theta_{k,\max} \cdot \bar{T} \qquad \text{for } k = 1 \ldots K \tag{55i}$$

$$\sum_{k=1}^{K} (\kappa_{p,k} \cdot \bar{\theta}_k) \leq \vartheta_p \cdot \bar{T} \qquad \text{for } p = 1 \ldots P \tag{55j}$$

$$\frac{1}{T_{\max}} \leq \bar{T} \leq \frac{1}{T_{\min}} \tag{55k}$$

$$Y \leq Y_{\max} \tag{55l}$$

5 Illustrating example

In order to illustrate the modeling and solution procedure proposed in Sections 2 to 4, we describe all necessary steps for the strictly cyclic case as well as for the hierarchical

cyclic structure, both for the very simple example from Fig. 1. The batch definition consists of $n = 4$ activities on $m = 2$ resources. Each single batch passes through the following sequence:

– Workstep on resource 2 (4 time units)
– Transfer to resource 1 (4 time units)
– Workstep on resource 1 (6 time units)
} activity 1, $J_1 = 2$
– Interval during which the batch is not present in the system but, e.g. stored in an infinite buffer (minimum 40 time units, maximum 46 time units)
} activity 2, $J_2 = 1$
– Workstep on resource 1 (4 time units)
– Transfer to resource 2 (4 time units)
– Workstep on resource 2 (8 time units)
} activity 3, $J_3 = 1$
} activity 4, $J = 2$

Inserting any artificial waiting times between the worksteps and the transfers will not allow for increased throughput. The interval between activity 2 and 3 therefore constitutes the only degree of freedom within the single batch time scheme. We assume that the batch definition allows this interval to vary by a maximum of 6 time units.

In this case, the parametrization (4) can be found straightforward. It incorporates one variable θ_1 (i.e. $K = 1$) and is chosen such that this variable represents the additional waiting time between activity 2 and 3, up to a maximum of 6 time units. This results in the following values for the parameters $\chi_{i,0}$, $\psi_{i,0}$, $\chi_{i,k}$, and $\psi_{i,k}$:

$$\begin{aligned} \chi_{1,0} &= 0 & \chi_{1,1} &= 0 & \psi_{1,0} &= 8 & \psi_{1,1} &= 0 \\ \chi_{2,0} &= 4 & \chi_{2,1} &= 0 & \psi_{2,0} &= 14 & \psi_{2,1} &= 0 \\ \chi_{3,0} &= 56 & \chi_{3,1} &= 1 & \psi_{3,0} &= 64 & \psi_{3,1} &= 1 \\ \chi_{4,0} &= 60 & \chi_{4,1} &= 1 & \psi_{4,0} &= 72 & \psi_{4,1} &= 1 \end{aligned} \tag{56}$$

Figure 1 illustrates the single batch time scheme for $\theta_1 = 0$.

Condition (5) gives

$$0 \le \theta_1 \le \theta_{1,\max} = 6. \tag{57}$$

Since there is only one degree of freedom in the single batch time scheme, there are no constraints of form (6) necessary for this example.

With values (56), Condition (8) results in

$$T \ge 18, \qquad T \ge 20. \tag{58}$$

There are two pairs of activities (i_1, i_2), $i_2 > i_1$, $J_{i_1} = J_{i_2}$ for which constraints (10) and (11) need to be considered:

$$(i_1, i_2) \in \{(2, 3), (1, 4)\} \ .$$

This means we need to introduce the integer variables $z_1 = z_{(2,3)}$ and $z_2 = z_{(1,4)}$. The abbreviations (15) to (18) provide the following notation:

$$\begin{aligned} v_{1,0} &= 42 & v_{1,1} &= 1 \\ w_{1,0} &= 60 & w_{1,1} &= 1 \\ v_{2,0} &= 52 & v_{2,1} &= 1 \\ w_{2,0} &= 72 & w_{2,1} &= 1 \end{aligned} \tag{59}$$

Hence, we get two pairs of constraints of the form (13) resp. (14):

$$\begin{aligned} z_1 \cdot T - (\,42 + \theta_1\,) &\leq 0 \\ (z_1+1)\cdot T - (\,60 + \theta_1\,) &\geq 0 \\ z_2 \cdot T - (\,52 + \theta_1\,) &\leq 0 \\ (z_2+1)\cdot T - (\,72 + \theta_1\,) &\geq 0 \end{aligned} \tag{60}$$

To sum up, 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 $\left(\; T \in \mathbb{R}^+,\;\; \theta_1 \in \mathbb{R}_0^+,\;\; z_\iota \in \mathbb{Z}, \iota \in \{1,2\}\;\right)$
such that conditions (60), as well as conditions (57) and (58) 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 Eq. 23. A globally optimal solution is

$$T = 36,\;\; \theta_1 = 0,\;\; z_1 = 1,\;\; z_2 = 1.$$

A graphical representation of the resulting strictly cyclic schedule can be found in Fig. 2.

To illustrate the extension from Section 4, we now allow for a hierarchical cyclic structure with a maximum number of $Y_{\max} = 5$ identical jobs in one batch. The job time scheme matches the batch time scheme from the strictly cyclic case and involves $n^* = 4$ activities and the time variable θ_1 with $0 \leq \theta_1 \leq \theta_{1,\max} = 6$. The batch time scheme involves up to 5 such jobs and therefore $n = 5 \cdot 4 = 20$ activities. An additional time variable θ_2 is introduced to represent the cycle time of the inner cycle T^*. This results in the following values for the parameters $\chi_{i,0}$, $\psi_{i,0}$, $\chi_{i,k}$, and $\psi_{i,k}$ in the parametrization (37):

$$\begin{array}{llllll}
\chi_{1,0} = 0 & \chi_{1,1} = 0 & \chi_{1,2} = 0 & \psi_{1,0} = 8 & \psi_{1,1} = 0 & \psi_{1,2} = 0 \\
\chi_{2,0} = 4 & \chi_{2,1} = 0 & \chi_{2,2} = 0 & \psi_{2,0} = 14 & \psi_{2,1} = 0 & \psi_{2,2} = 0 \\
\chi_{3,0} = 56 & \chi_{3,1} = 1 & \chi_{3,2} = 0 & \psi_{3,0} = 64 & \psi_{3,1} = 1 & \psi_{3,2} = 0 \\
\chi_{4,0} = 60 & \chi_{4,1} = 1 & \chi_{4,2} = 0 & \psi_{4,0} = 72 & \psi_{4,1} = 1 & \psi_{4,2} = 0 \\
\chi_{5,0} = 0 & \chi_{1,1} = 0 & \chi_{1,2} = 1 & \psi_{5,0} = 8 & \psi_{1,1} = 0 & \psi_{1,2} = 1 \\
\chi_{6,0} = 4 & \chi_{2,1} = 0 & \chi_{2,2} = 1 & \psi_{6,0} = 14 & \psi_{2,1} = 0 & \psi_{2,2} = 1 \\
\chi_{7,0} = 56 & \chi_{3,1} = 1 & \chi_{3,2} = 1 & \psi_{7,0} = 64 & \psi_{3,1} = 1 & \psi_{3,2} = 1 \\
\chi_{8,0} = 60 & \chi_{4,1} = 1 & \chi_{4,2} = 1 & \psi_{8,0} = 72 & \psi_{4,1} = 1 & \psi_{4,2} = 1 \\
 & & & \vdots & & \\
\chi_{17,0} = 0 & \chi_{1,1} = 0 & \chi_{1,2} = 4 & \psi_{17,0} = 8 & \psi_{1,1} = 0 & \psi_{1,2} = 4 \\
\chi_{18,0} = 4 & \chi_{2,1} = 0 & \chi_{2,2} = 4 & \psi_{18,0} = 14 & \psi_{2,1} = 0 & \psi_{2,2} = 4 \\
\chi_{19,0} = 56 & \chi_{3,1} = 1 & \chi_{3,2} = 4 & \psi_{19,0} = 64 & \psi_{3,1} = 1 & \psi_{3,2} = 4 \\
\chi_{20,0} = 60 & \chi_{4,1} = 1 & \chi_{4,2} = 4 & \psi_{20,0} = 72 & \psi_{4,1} = 1 & \psi_{4,2} = 4
\end{array} \tag{61}$$

together with

$$0 \leq \theta_1 \leq \theta_{1,\max} = 6. \tag{62}$$

Constraints (46) read as follows (constraints for $j = 1$ are redundant and therefore omitted):

$$\begin{array}{lll} 20 - T \leq 0 & \text{OR} & Y \leq 0 \\ 40 - T \leq 0 & \text{OR} & Y \leq 1 \\ \vdots & & \\ 100 - T \leq 0 & \text{OR} & Y \leq 4. \end{array} \tag{63}$$

The batch time scheme involves $n_j = 10$ activities for each of two resources, thus resulting in $\iota_{\max} = 90$ integer variables z_ι (Eq. 12). Constraints (50), (51) for 68 out of 90 variables z_ι turn out to be redundant. Omitting the corresponding 136 redundant lines, constraints (50) and (51) result in

$$\begin{array}{rrrrcll} z_1 \cdot T - (& 42 & +\theta_1 &) \leq 0 & \text{OR} & Y \leq 0 \\ (z_1 + 1) \cdot T - (& 60 & +\theta_1 &) \geq 0 & \text{OR} & Y \leq 0 \\ z_2 \cdot T - (& 52 & +\theta_1 &) \leq 0 & \text{OR} & Y \leq 0 \\ (z_2 + 1) \cdot T - (& 72 & +\theta_1 &) \geq 0 & \text{OR} & Y \leq 0 \\ z_3 \cdot T - (& 42 & +\theta_1 & +\theta_2) \leq 0 & \text{OR} & Y \leq 1 \\ (z_3 + 1) \cdot T - (& 60 & +\theta_1 & +\theta_2) \geq 0 & \text{OR} & Y \leq 1 \\ z_4 \cdot T - (& 52 & +\theta_1 & +\theta_2) \leq 0 & \text{OR} & Y \leq 1 \\ (z_4 + 1) \cdot T - (& 72 & +\theta_1 & +\theta_2) \geq 0 & \text{OR} & Y \leq 1 \\ & & \vdots & & & \\ z_9 \cdot T - (& 42 & +\theta_1 & +4\theta_2) \leq 0 & \text{OR} & Y \leq 4 \\ (z_9 + 1) \cdot T - (& 60 & +\theta_1 & +4\theta_2) \geq 0 & \text{OR} & Y \leq 4 \\ z_{10} \cdot T - (& 52 & +\theta_1 & +4\theta_2) \leq 0 & \text{OR} & Y \leq 4 \\ (z_{10} + 1) \cdot T - (& 72 & +\theta_1 & +4\theta_2) \geq 0 & \text{OR} & Y \leq 4 \\ z_{11} \cdot T - (& -60 & -\theta_1 & +\theta_2) \leq 0 & \text{OR} & Y \leq 1 \\ (z_{11} + 1) \cdot T - (& -42 & -\theta_1 & +\theta_2) \geq 0 & \text{OR} & Y \leq 1 \\ z_{12} \cdot T - (& -72 & -\theta_1 & +\theta_2) \leq 0 & \text{OR} & Y \leq 1 \\ (z_{12} + 1) \cdot T - (& -52 & -\theta_1 & +\theta_2) \geq 0 & \text{OR} & Y \leq 1 \\ z_{13} \cdot T - (& -60 & -\theta_1 & +2\theta_2) \leq 0 & \text{OR} & Y \leq 2 \\ (z_{13} + 1) \cdot T - (& -42 & -\theta_1 & +2\theta_2) \geq 0 & \text{OR} & Y \leq 2 \\ z_{14} \cdot T - (& -72 & -\theta_1 & +2\theta_2) \leq 0 & \text{OR} & Y \leq 2 \\ (z_{14} + 1) \cdot T - (& -52 & -\theta_1 & +2\theta_2) \geq 0 & \text{OR} & Y \leq 2 \\ z_{15} \cdot T - (& -60 & -\theta_1 & +3\theta_2) \leq 0 & \text{OR} & Y \leq 3 \\ (z_{15} + 1) \cdot T - (& -42 & -\theta_1 & +3\theta_2) \geq 0 & \text{OR} & Y \leq 3 \\ z_{16} \cdot T - (& -72 & -\theta_1 & +3\theta_2) \leq 0 & \text{OR} & Y \leq 3 \\ (z_{16} + 1) \cdot T - (& -52 & -\theta_1 & +3\theta_2) \geq 0 & \text{OR} & Y \leq 3 \\ z_{17} \cdot T - (& -60 & -\theta_1 & +4\theta_2) \leq 0 & \text{OR} & Y \leq 4 \\ (z_{17} + 1) \cdot T - (& -42 & -\theta_1 & +4\theta_2) \geq 0 & \text{OR} & Y \leq 4 \\ z_{18} \cdot T - (& -72 & -\theta_1 & +4\theta_2) \leq 0 & \text{OR} & Y \leq 4 \\ (z_{18} + 1) \cdot T - (& -52 & -\theta_1 & +4\theta_2) \geq 0 & \text{OR} & Y \leq 4 \\ z_{19} \cdot T - (& -12 & & +\theta_2) \leq 0 & \text{OR} & Y \leq 1 \\ (z_{19} + 1) \cdot T - (& 12 & & +\theta_2) \geq 0 & \text{OR} & Y \leq 1 \\ z_{20} \cdot T - (& -12 & & +2\theta_2) \leq 0 & \text{OR} & Y \leq 2 \\ (z_{20} + 1) \cdot T - (& 12 & & +2\theta_2) \geq 0 & \text{OR} & Y \leq 2 \\ z_{21} \cdot T - (& -12 & & +3\theta_2) \leq 0 & \text{OR} & Y \leq 3 \\ (z_{21} + 1) \cdot T - (& 12 & & +3\theta_2) \geq 0 & \text{OR} & Y \leq 3 \\ z_{22} \cdot T - (& -12 & & +4\theta_2) \leq 0 & \text{OR} & Y \leq 4 \\ (z_{22} + 1) \cdot T - (& 12 & & +4\theta_2) \geq 0 & \text{OR} & Y \leq 4. \end{array} \tag{64}$$

The following lower and upper bounds for the mean job cycle time T/Y respectively the batch cycle time T can be added to the optimization problem:

$$T_{\text{lo}} = T_{\text{min},Y=1} = 20 \tag{65}$$

$$T_{\text{up}} = T_{\text{opt}}\ (\text{strictly cyclic}) = 36 \tag{66}$$

$$T_{\text{min}} = T_{\text{lo}} = 20 \tag{67}$$

$$T_{\text{max}} = Y_{\text{max}} \cdot T_{\text{max}}\ (\text{strictly cyclic}) = 360. \tag{68}$$

Constraints (62) to (68) build up the problem formulations (54a) to (54i), which can be reformulated to linear form according to Eqs. 55a to 55l, following steps 1 to 5 in Section 4.4.

The globally optimal solution results in

$$Y = 4\,, \quad T = 72\,, \quad \theta_1 = 0\,, \quad \theta_2 = T^* = 12$$

and is pictured in Fig. 3. The mean job cycle time for this four-periodic schedule is $T/Y = 18$. Compared to the strictly cyclic solution, throughput is improved by 100%.

6 Application example

As a second example, we apply the proposed method to the ELISA test, which is a typical problem instance from high throughput screening. The example is adapted from Murray and Anderson (1996). It involves all requirements listed in Section 1. A compact illustration of the example is provided by the Gantt chart for the single batch time scheme in Fig. 4, which consists of four distinct parts. The resources of this example include not only machines for liquid handling, incubation and reading etc. but also transport units: three robots are used to move the entities (microplates) between the machine resources. The time an empty robot needs to move to the next resource after having transferred a plate to its destination is neglected.

The time distances between the four parts are restricted by lower and upper bounds (time window constraints). Apart from these restrictions, all activities may be arbitrarily prolonged by inserting an idle time interval before the end of the activity,

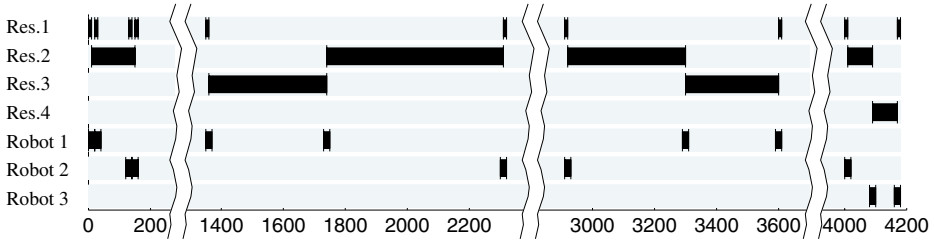


Fig. 4 Job time scheme for the ELISA assay (Gantt chart). Note that the problem specification also includes lower and upper bounds for time intervals, which cannot be visualized here

causing the follow-up activity on the next resource to start later. Altogether, these degrees of freedom are used to find a globally optimal timing for the activities, hence allowing for a schedule with maximum throughput.

Following the modeling process of Sections 2 and 3, a mathematical program can be established for the strictly cyclic case. The basic MILP consists of 28 continuous and 78 integer variables and includes 195 constraints (apart from lower and upper bounds for variables). The solution of the mathematical program provides a strictly cyclic schedule with globally optimal throughput. During the modeling process, an upper bound $T_{\max} = 2350$ for T can be found by solving the problem for the fixed single batch time scheme from Fig. 4, i.e. without allowing for any change in the timing. A lower bound $T_{\min} = 1150$ can be established by solving the relaxed MILP.

After adding bounding constraints for the variables and cuts as described in Section 3.1, a globally optimal solution to the mathematical program is found within less than 1 second on a 2 GHz Linux PC. The optimal cycle time turns out to be $T_{\text{opt}} = 2260$ and therefore allows for an increase in throughput of 4% compared to the originally known solution $T_{\max} = 2350$. Information about the model size and the size of the mathematical program as well as computation times are given in Table 1 below.

We now drop the prerequisite of strictly cyclic operation and allow for hierarchical nesting of cycles as described in the extension in Section 4: while all entities still have to be processed within the same time scheme, the distance between the start of consecutive entities is not required to be constant any more. A maximum number of $Y_{\max} = 4$ entities form one batch and are processed in an inner cyclic scheme. Following the modeling steps of Section 4, the scheduling problem is cast into a mixed integer linear program of the form (55a) to (55l) with 30 continuous, 528 integer variables, and 1079 constraints.

The resulting mixed-integer linear programs can be solved using standard integer programming software like CPLEX (ILOG 2006). Since the range of Y is small, it is worth to require branching on Y first. This can be achieved by supplying suitable priority-order files to the small CPLEX solver, and allows significant problem compactification in the Branch-and-Cut tree on subsequent nodes.

Optimization results in a globally optimal solution with $Y_{\text{opt}} = 3$ and $T_{\text{opt}} = 4150$: three jobs have to be grouped to one batch in the inner cycle; a new batch is started every 4150 time units. Hence, the mean cycle time results in $4150/3 = 1383\frac{1}{3}$, which is significantly better than the best possible cycle time 2260 for the strictly cyclic case.

Table 1 provides data about the basic ELISA problem (strictly cyclic case) and the extended version (hierarchical cyclic structure). For both problems, the table

Table 1 ELISA problem instances

m	n	$Y_{\max}$	n^*	#var	#constr	#integer	obj.function	t_{solve}
ELISA: strictly cyclic								
7	30	1	30	106	195	78	$T_{\text{opt}} = 2260$	1 s$^{(*)}$
ELISA: hierarchical cyclic structure								
7	120	4	30	558	1079	528	$T_{\text{opt}}/Y_{\text{opt}} = 1383\frac{1}{3}$	1105 s$^{(**)}$

Table 2 Problem instances from high throughput screening

m	n	$Y_{\max}$	n^*	#var	#constr	#integer	t_{solve}
Example from Section 5							
2	4	1	4	4	7	2	0.01 s(*)
Normal-sized problem							
18	57	1	57	183	1051	131	0.65 s(*)
Large sized problem							
18	87	1	87	399	2402	321	122 s(*)

gives the size of the underlying scheduling problem as well as the following data about the MILP:

- The number of variables, i.e. the number of columns in the MILP (#var),
- The number of constraints, i.e. the number of rows in the MILP (#constr), and
- The number of integer variables in the MILP (#integer).

Additionally, the globally optimal objective function value is given, which is equivalent to the mean cycle time, i.e. the inverse of the best possible throughput. Finally, the last column of Table 1 shows the time needed to compute the globally optimal solution of the MILP on a 2 GHz Linux PC (*) resp. a Sun-Fire-V440 with 1.28 GHz Ultrasparc-IIIi processors (**).

To show the applicability of the strictly cyclic scheduling method to problems of different size, Table 2 provides analogue figures for three other problem instances from high throughput screening. Globally optimal strictly cyclic schedules have been calculated for all problems within reasonable computation time. For the hierarchical cyclic approach, the computation time furthermore depends on the value for $Y_{\max}$. From Table 1, it can be seen that the computation of the globally optimal hierarchical cyclic schedule takes much more time then the computation of the globally optimal strictly cyclic schedule. Nevertheless, due to large numbers of batches, this will normally be worthwhile since throughput is considerably increased.

7 Conclusion

In this paper, we have addressed the problem of finding throughput-optimal sequences for a class of cyclic discrete event systems. We are interested in systematic approaches that guarantee globally optimal solutions. For the case of strictly cyclic operation, it has been shown that globally optimal solutions can be found by casting the cyclic scheduling problem into a mixed integer linear program (MILP). The method has been very successful in computing strictly cyclic schedules for a large number of high throughput screening problems from the pharmaceutical industries.

Additionally, a new extension towards a more general case of cyclicity has been presented: for some systems, hierarchical nesting of cycles allows for schedules with improved throughput rates. For this extension, it is still possible to solve the problem in a globally optimal way by casting it into an MILP. However, nesting of cycles

demands for a more advanced modeling approach and thus results in more complex mathematical programs.

The extended method has been applied to an example from high throughput screening, showing its applicability to real-sized problem instances. Clearly, the computation time for the solution of the resulting mathematical program suffers from combinatoric explosion, as the size of the problem is increased. Further mathematical programming methods are needed to improve computation times and to solve large problem instances. Such methods are currently under investigation.

Acknowledgement The authors gratefully acknowledge funding by "Deutsche Forschungsgemeinschaft (DFG)" via the DFG-Forschergruppe 468 "Methods from Discrete Mathematics for the Synthesis and Control of Chemical Processes".

References

Alle A, Papageorgiou L, Pinto J (2004) A mathematical programming approach for cyclic production and cleaning scheduling of multistage continuous plants. Comput Chem Eng 28(1–2):3–15

Bertsimas D, Weismantel R (2005) Optimization over integers. Dynamic Ideas, Belmont

Brucker P (2004) Scheduling algorithms. Springer

Chen H, Chu C, Proth J-M (1998) Cyclic scheduling of a hoist with time window constraints. IEEE Trans Robot Autom 14(1):144–152

Crama Y, Kats V, van de Klundert J, Levner E (2000) Cyclic scheduling in robotic flowshops. Ann Oper Res 96:97–124

Hall N, Lee T-E, Posner M (2002) The complexity of cyclic shop scheduling problems. J Sched 5(4):307–327

ILOG (2006) ILOG CPLEX. http://www.ilog.com/products/cplex/

Karimi I, Tan Z, Bhushan S (2004) An improved formulation for scheduling an automated wet-etch station. Comput Chem Eng 29(1):217–224

Lee T-E (2000) Stable earliest starting schedules for cyclic job shops: a linear system approach. Int J Flex Manuf Syst 12(1):59–80

Lee T-E, Posner M (1997) Performance measures and schedules in periodic job shops. Oper Res 45(1):72–91

Levner E, Kats V (1998) A parametric critical path problem and an application for cyclic scheduling. Discrete Appl Math 87(1–3):149–158

Levner E, Kats V, Levit V (1997) An improved algorithm for cyclic flowshop scheduling in a robotic cell. Eur J Oper Res 97(3):500–508

Mayer E (2007) Globally optimal schedules for cyclic systems with non-blocking specification and time window constraints. PhD Thesis, Fachgebiet Regelungssysteme, Technische Universität Berlin, Germany.

Mayer E, Raisch J (2003) Throughput-optimal scheduling for cyclically repeated processes. MMAR2003—9th IEEE international conference on methods and models in automation and robotics. Miedzyzdroje, Poland, pp 871–876

Mayer E, Raisch J (2004) Time-optimal scheduling for high throughput screening processes using cyclic discrete event models. MATCOM—Math Comput Simul 66(2–3):181–191

Middendorf M, Timkovsky VG (2002) On scheduling cycle shops: classification, complexity and approximation. J Sched 5(2):135–169

Murray C, Anderson C (1996) Scheduling software for high-throughput screening. Lab Robot Autom 8(5):295–305

Odijk M (1996) A constraint generation algorithm for the construction of periodic railway timetables. Transp Res Part B—Methodol 30B(6):455–464

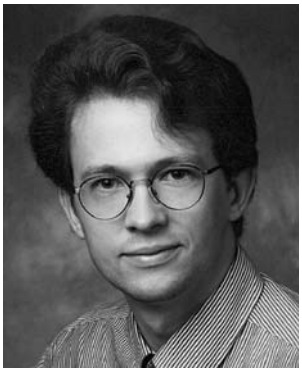
Parker R (1995) Deterministic scheduling theory. Chapman & Hall

Pinedo M (1995) Scheduling: theory, algorithms, and systems. Prentice Hall

Pinto J, Grossmann I (1994) Optimal cyclic scheduling of multistage continuous multiproduct plants. Comput Chem Eng 18(9):797–816

Pinto J, Grossmann I (1998) Assignment and sequencing models for the scheduling of process systems. Ann Oper Res 81:433–466
Roundy R (1992) Cyclic schedules for job shops with identical jobs. Math Oper Res 17(4):842–865
Seo J-W, Lee T-E (2002) Steady-state analysis and scheduling of cyclic job shops with overtaking. Int J Flex Manuf Syst 14(4):291–318
Shah N, Pantelides C, Sargent R (1993) Optimal periodic scheduling of multipurpose batch plants. Ann Oper Res 42(1–4):193–228

Eckart Mayer studied Engineering Cybernetics and Control Engineering at Stuttgart University and at University of Exeter, UK. At the time of the work presented in this paper, he was a member of the Systems and Control Theory Group at the Max Planck Institute for Dynamics of Complex Technical Systems in Magdeburg, Germany. He received his Ph.D in Electrical Engineering from Technische Universität Berlin in 2007 and is now working for Robert Bosch GmbH in Stuttgart, Germany.



Utz-Uwe Haus is post-doc at the University of Magdeburg, Germany, where he heads a Junior Research Group at the Research Center "Dynamic Systems in Biomedicine and Process Engineering". He studied Mathematics at the Technical University in Berlin, Germany, where he received the Diploma in 1999. He completed his doctoral studies in 2004 at the Institute for Mathematical Optimization of the University of Magdeburg. His research interests are applications of integer linear and nonlinear programming in biological and engineering applications, as well as logical models of signal transduction networks in biomedicine.

Jörg Raisch is a professor at Technische Universität Berlin, where he heads the Control Systems Group within the Department of Electrical Engineering and Computer Science. He is also head of the Systems and Control Theory Group at the Max Planck Institute for Dynamics of Complex Technical Systems in Magdeburg, Germany. He studied Engineering Cybernetics and Control Systems at Stuttgart University and UMIST, Manchester. He got his Ph.D and "Habilitation", both from Stuttgart University, in 1991 and 1998, respectively. From 1991–1993 he was a postdoc in the Systems Control Group at the University of Toronto. From 2000–2006 he was a professor at the Otto-von-Guericke University Magdeburg, where he headed the automatic control lab. His research interests are in hybrid systems and hierarchical control and include biomedical control and chemical process control applications.

Robert Weismantel is the chair professor for Mathematical Optimization at the Faculty of Mathematics of the University of Magdeburg, Germany. He is a spokesman of the Research Center "Dynamic Systems in Biomedicine and Process Engineering". He studied Mathematics at the University of Augsburg, Germany, where he received the Diploma in 1988. He completed his doctoral studies in 1992 and his habilitation in 1995 at the Technical University in Berlin. He was an associate head of the department "Combinatorial Optimization" at ZIB Berlin, a post-doc and a visiting professor at various places in Europe and the US before he joined the University of Magdeburg in 1998. His research interests are the theory of algorithmic discrete mathematics and their application to life sciences and engineering.