Computers and Chemical Engineering 33 (2009) 1644–1660

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

A novel network-based continuous-time representation for process scheduling: Part II. General framework

Diego M. Giménez[a], Gabriela P. Henning[a,*], Christos T. Maravelias[b]

[a] INTEC (Universidad Nacional del Litoral - CONICET), Güemes 3450, S3000GLN Santa Fe, Argentina
[b] Department of Chemical and Biological Engineering, University of Wisconsin, Madison, WI 53706, USA

ARTICLE INFO

Article history:
Received 8 November 2008
Received in revised form 22 March 2009
Accepted 28 April 2009
Available online 9 May 2009

Keywords:
Process scheduling
Network-based continuous-time representation
Mixed-integer linear programming

ABSTRACT

In the first part of this series of papers we presented a new network-based continuous-time representation for the short-term scheduling of batch processes, which overcomes numerous shortcomings of existing approaches. In this second part, we discuss how this representation can be extended to address aspects such as: (i) preventive maintenance activities on unary resources (e.g., processing and storage units) that were planned ahead of time; (ii) resource-constrained changeover activities on processing and shared storage units; (iii) non-instantaneous resource-constrained material transfer activities; (iv) intermediate deliveries of raw materials and shipments of finished products at predefined times; and (v) scenarios where part of the schedule is fixed because it has been programmed in the previous scheduling horizon. The proposed integrated framework can be used to address a wide variety of process scheduling problems, many of which are intractable with existing tools.

1. Introduction

Most existing approaches to the scheduling of general batch processes are based on the state-task network (STN) or the resource-task network (RTN) representations proposed by Kondili, Pantelides, and Sargent (1993) and Pantelides (1994), respectively. These representations have been used as a basis for the development of a variety of discrete- (Shah, Pantelides, & Sargent, 1993), continuous- (Castro, Barbosa-Póvoa, & Matos, 2001; Ierapetritou & Floudas, 1998; Maravelias & Grossmann, 2003a; Mockus & Reklaitis, 1999; Schilling & Pantelides, 1996; Sundaramoorthy & Karimi, 2005; Zhang & Sargent, 1996), as well as mixed-time (Maravelias, 2005) network-based mixed-integer linear programming (MILP) formulations. In most of these formulations it is implicitly assumed that:

(a) All processing units are connected to all the vessels that are used for the storage of the corresponding input and output materials, as well as to all upstream/downstream processing units. Thus, material transfer between units is always possible.
(b) All input (output) materials consumed (produced) by a task are transferred simultaneously to (from) the processing unit when the task starts (ends). This further implies that inventory changes are always caused by the beginning and end of processing tasks, and not by the execution of specific material transfer activities.
(c) Stable output materials can be temporarily stored in a processing unit after a task is completed, but stable input materials cannot be temporarily stocked up before a task actually starts, i.e. in continuous-time representations the beginning of a task must coincide with a time point; besides, the storage of stable output materials is always bounded by the time point representing the end of the task.
(d) Material transfers are viewed as instantaneous activities which are executed to change the location of material resources and have no resource requirements.

However, these assumptions do not always hold in industrial environments. Regarding the first assumption, the actual topologies of most batch plants impose connectivity constraints that prevent material transfers between certain pieces of equipment. In the same way, according to some industrial recipes, the materials that are fed to or drained from a processing unit are not necessarily transferred at the same time; thus, violating the second assumption. For example, in certain chemical reactions reactants are fed before the beginning of the task, which actually occurs when the catalyst is added. In this way, the reactor can also be used as a temporary storage tank. This example presents a situation in which the third assumption is also violated because input materials are maintained within the processing unit before the task actually starts. On the other hand, a certain input (output) material may be fed (discharged) into (from) a processing unit by resorting to multiple

* Corresponding author. Tel.: +54 342 4559175; fax: +54 342 4550944.
E-mail address: ghenning@intec.unl.edu.ar (G.P. Henning).

0098-1354/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compchemeng.2009.04.013

Nomenclature

Sets/Indices

$\mathbf{N}/n, n'$ global time points/intervals
$\mathbf{I}/i, i'$ tasks
$\mathbf{I}^P/\mathbf{I}^C/\mathbf{I}^M/\mathbf{I}^T/\mathbf{I}^{PH}$ processing/cleaning/maintenance/transfer/previous horizon tasks
$\mathbf{I}^{CZW}/\mathbf{I}^{PZW}$ processing tasks $i \in \mathbf{I}^P$ that consume/produce an unstable material
$\mathbf{I}^{TZW}$ transfer tasks $i \in \mathbf{I}^T$ associated with an unstable material
$\mathbf{J}/j, j'$ units
$\mathbf{J}^P/\mathbf{J}^S/\mathbf{J}^C/\mathbf{J}^T$ processing/storage/cleaning/transfer units
$\mathbf{J}^{DS}/\mathbf{J}^{SS}$ dedicated/shared storage units
$\mathbf{M}/m$ materials
$\mathbf{M}^D/\mathbf{M}^P$ sold/purchased materials
$\mathbf{M}^V$ intermediate materials with commercial value
$\mathbf{M}^{ZW}$ unstable materials (they must be handled under a zero-wait policy)
$\mathbf{M}^{NIS}$ materials for which no storage unit is available
$\mathbf{M}^S$ materials that can be stored in storage units
$\mathbf{L}/l$ material shipments
$\mathbf{R}/r$ non-unary resources
$\mathbf{C}/c$ unit modes
$\mathbf{V}/v$ transfer devices
$\mathbf{I}_m^C/\mathbf{I}_m^P$ processing tasks $i \in \mathbf{I}^P$ that consume/produce material m
$\mathbf{I}_j$ tasks associated with unit j
$\mathbf{I}_r$ tasks that require non-unary resource r
$\mathbf{I}_c^P$ tasks that produce mode c when end
$\mathbf{J}_j$ units connected to unit j
$\mathbf{J}_i$ units associated with execution of task i
$\mathbf{J}_m^S$ storage units that can store material m
$\mathbf{J}_m^I/\mathbf{J}_m^O$ processing units $j \in \mathbf{J}^P$ that can store material m as an input/output material
$\mathbf{J}_v^T$ transfer units $j \in \mathbf{J}^T$ in which transfer device v is a component
$i(m, j)$ transfer task $i \in \mathbf{I}^T$ associated with the transfer of material m through transfer unit $j \in \mathbf{J}^T$
$j(i)$ unit j on which task i is scheduled to take place
$j(j', j'')$ transfer unit $j \in \mathbf{J}^T$ that connects the outlet of unit $j' \in (\mathbf{J}^P \cup \mathbf{J}^S)$ with the inlet of unit $j'' \in (\mathbf{J}^P \cup \mathbf{J}^S)$
$\mathbf{M}_j^S$ materials that can be stored in unit $j \in \mathbf{J}^{SS}$
$\mathbf{M}_j^I/\mathbf{M}_j^O$ materials that can be stored in processing unit $j \in \mathbf{J}^P$ as input/output materials
$m(l)$ material corresponding to shipment l
$m(c)$ material whose storage during a time interval produces mode c
$\mathbf{C}_j$ modes defined for unit $j \in (\mathbf{J}^P \cup \mathbf{J}^{SS})$
$\mathbf{C}_i^C/\mathbf{C}_m^C$ modes that can be consumed by the beginning of task i/storage of material m

Parameters

$A_{c,j}^0$ 1 if mode $c \in \mathbf{C}_j$ is initially available in unit $j' \in (\mathbf{J}^P \cup \mathbf{J}^{SS})$ at the beginning of the horizon; otherwise, it must be equal to zero.
H time horizon
$a_{i,j}/b_{i,j}$ fixed/variable duration of task i in unit j
$B_{i,j}^0$ batch size of previous horizon processing task $i \in (\mathbf{I}^P \cap \mathbf{I}^{PH})$ executed in unit $j \in \mathbf{J}^P$
$f_{i,j,r}/g_{i,j,r}$ fixed/variable amount of resource r required by task i in unit j
$I_{m,j}^0$ initial amount of material m in storage unit $j \in \mathbf{J}^S$
q_l amount associated with shipment l
$\beta_j^{MAX}/\beta_J^{MIN}$ maximum/minimum capacity for unit j
$\gamma_{i,m}$ mass fraction for the consumption (−)/production (+) of material m by processing task $i \in \mathbf{I}^P$
π_m price (value) of material m
ρ_r^{MAX} maximum availability of resource r
ς_j^{MAX} maximum storage capacity for storage unit $j \in \mathbf{J}^S$
σ_i cleaning cost associated with cleaning task $i \in \mathbf{I}^C$
τ_l time corresponding to shipment l
τ_i^{MB}/τ_i^{ME} fixed start/end time of maintenance task $i \in \mathbf{I}^M$
$\tau_i^{MEB}/\tau_i^{MLE}$ earliest start time/latest end time for maintenance task $i \in \mathbf{I}^M$

Binary variables

$O_{l,j,n}$ 1 if shipment l occurs to/from storage unit $j \in \mathbf{J}^S$ at time point n
$S_{j,n}^I/S_{j,n}^O$ 1 if in processing unit $j \in \mathbf{J}^P$ input/output materials are stored during time interval n
$S_{m,j,n}^S$ 1 if material $m \in \mathbf{M}_j^S$ is stored in shared storage unit $j \in \mathbf{J}^{SS}$ during time interval n
$X_{i,j,n}/Y_{i,j,n}$ 1 if task $i \in \mathbf{I}_j$ formally starts/ends in unit j at T_n

Continuous (non-negative) variables

$A_{c,j,n}$ 1 if mode $c \in \mathbf{C}_j$ is available in unit $j \in (\mathbf{J}^P \cup \mathbf{J}^{SS})$ at time point n
$E_{j,n}$ 1 if unit j is formally performing/receiving a task during time interval n
$K_{c,j,n}^C/K_{c,j,n}^P$ 1 if mode $c \in \mathbf{C}_j$ is consumed/produced in unit $j \in (\mathbf{J}^P \cup \mathbf{J}^{SS})$ at time point n
$S_{j,n}$ 1 if unit $j \in (\mathbf{J}^P \cup \mathbf{J}^S)$ is storing materials during time interval n
$W_{j,n}$ 1 if unit j is available during time interval n
$Z_{j,n}$ 1 if unit j is performing/receiving at time point n a task started in a previous time point
$B_{i,j,n}^S/B_{i,j,n}^E$ batch size of task i which formally starts/ends at T_n in unit j
$B_{i,j,n}^P$ batch size of task i being executed at T_n in/by unit j
$F_{m,j',j,n}$ instantaneous transfer of material m from unit $j' \in (\mathbf{J}^P \cup \mathbf{J}^S)$ to $j \in ((\mathbf{J}^P \cup \mathbf{J}^S) \cap \mathbf{J}_{j'})$ at time point n
$I_{m,j,n}^I/I_{m,j,n}^O$ amount of input/output material m stored in processing unit $j \in \mathbf{J}^P$ during time interval n
$I_{m,j,n}^S$ amount of material m stored in storage unit $j \in \mathbf{J}^S$ during time interval n
$P_{m,j,n}$ amount of output material m in unit $j \in \mathbf{J}^P$ that becomes an input material for the same unit at T_n
$Q_{r,n}$ total amount of non-unary resource r required during time interval n
T_n time corresponding to global point n
$\bar{T}_l^E/\bar{T}_l^T$ earliness/tardiness for shipment l
$\bar{T}_{j,n}^{LB}/\bar{T}_{j,n}^{EE}$ number of time units the actual beginning/end of a task is delayed/anticipated in unit j with respect to its formal beginning/end at T_n
$\bar{T}_{j,n}^S/\bar{T}_{j,n}^W$ length of the storage/idle interval n taking place in unit j from T_n to T_{n+1}

transfers of the same material ("partial" transfer), instead of making a unique one. Finally, any transfer task takes time and such time must be considered in the schedule. By assuming that transfer tasks are instantaneous, it is implicitly assumed that resources employed in such activities (pipes, pumps, valves, etc.) are not shared and, thus, not limiting, which is not true in practice. In addition, while a transfer occurs the source and destination units cannot be used to carry out processing tasks. Interestingly, despite the large number of methods recently proposed to tackle process scheduling, there are very few attempts to address these limitations in a comprehensive manner.

In the first paper of this series (Giménez, Henning, & Maravelias, 2009) we presented a novel continuous-time network-based representation that overcomes the shortcomings of previous approaches due to the first three aforementioned assumptions. We introduced five new modeling concepts that resulted in an effective MILP formulation, which explicitly accounts for material transfers and material inventory in processing units. In the present contribution we extend this framework to address five additional rather important aspects in batch scheduling that have received little attention in the literature. In particular, we consider:

(a) Preventive maintenance activities in the course of the scheduling horizon that restrain the usage of certain processing or storage units during a given period of time. In the industrial practice it is pretty common that certain time intervals associated with specific units are held in reserve to carry out maintenance operations. Another frequent situation is that maintenance activities are considered as other tasks to be scheduled within certain time windows. Situations like these ones are taken into account in many commercial scheduling packages.
(b) Sequence dependent or independent changeover activities (in processing units and shared vessels) having a duration that cannot be disregarded and that may also involve the use of common resources such as Cleaning-In-Place (CIP) and Sterilization-In-Place (SIP) stations, personnel, etc. CIP and SIP are important to many industries such as food, dairy, beverage, nutraceutical, biotechnology, pharmaceutical, cosmetic, etc., in which the processing must take place in a hygienic and aseptic environment. Thus, when switching between different products, or even after one or several batches of the same product, units must usually be cleaned and/or sterilized.
(c) Non-instantaneous material transfer activities that may require shared resources like connections between units, ancillary piping devices, such as pumps, valves, etc. These transfer tasks not only demand time and resources to be carried out, but also prevent the use of the source and destination units while the material movement takes place. The proposed representation allows to seamlessly address problems where transfer operations are the major tasks (e.g., crude oil transfer operations).
(d) Intermediate deliveries of raw materials and shipments of finished products that must be done at predefined times during the scheduling horizon. When the scheduling horizon is not too short the hypothesis that all the raw materials are available at its launch does not always hold. Similarly, certain finished products may be demanded before the end of the scheduling horizon.
(e) A rolling scheduling horizon in which certain tasks that were programmed in the previous scheduling period are being continued in the current one, thus competing for resources. Note that this situation appears routinely in practice during rescheduling. Thus, our approach can be readily used to address reactive and dynamic scheduling problems.

It should be noticed that these five aspects have not been fully addressed in the literature. In fact, very few contributions have tackled these issues in a comprehensive way. In the last years, the trend has been to improve representations from a computational point of view (Janak & Floudas, 2008) and not from the perspective of the model expressive power. Nevertheless, some relevant works are discussed below.

Regarding plant structure, unit connectivity and material transfers, Crooks (1992) introduced a complete plant network representation that considers connections between processing units and relevant transfer operations. This representation was then extended to unambiguously represent information about recipes, flowsheets and material transfers in a combined and unified manner. Crooks (1992) was also the first one to bring in the notion of unit state to explicitly model the status of a processing unit (e.g., dirty, clean, etc.) as opposed to that of the material being processed.

Barbosa-Póvoa and Macchietto (1994a) extended the ideas behind Crooks' representation to address the detailed design of multipurpose batch plants as well as the retrofit design of a multipurpose pilot plant facility while accounting for Cleaning-In-Place integration (Barbosa-Póvoa & Macchietto, 1994b). To address the later, they explicitly considered connections among units, adopted the notions of unit and connection states, as well as transfer tasks. Regarding transfer activities they acknowledged both normal processing and cleaning related material transfers. More recently, Castro, Barbosa-Póvoa, and Novais (2005) also considered connection resources when they addressed the simultaneous design and scheduling of a multipurpose batch plant. They introduced a rich RTN representation that allowed designing connections between equipment units.

In the last decade, the assumptions of negligible transfer times and full connectivity among processing and storage units were generally accepted in most contributions. For instance, Castro et al. (2005) explicitly accounted for transfer tasks, which were assumed to be instantaneous, in order to address the synthesis of the plant pipeline network mentioned above. They considered the case where several transfer tasks regarding different input materials occur at distinct time points (i.e. non-simultaneous material transfers); however, each material was associated with only one transfer. Moreover, transfer tasks were also employed to model the transfer of material within the same equipment unit, to distinguish the case where the material is being produced from that where the material is being consumed.

In fact, the few works that explicitly addressed non-instantaneous transfer times have focused on multistage batch facilities. For the more general and complex multipurpose case, transfer time management have been neglected or assumed to be lumped into the batch processing time (Sundaramoorthy & Karimi, 2005). Recently, Ferrer-Nadal, Capón-García, Méndez, and Puigjaner (2008) pointed out the problems that may appear when transfer tasks are ignored. As it is shown later in this contribution, non-instantaneous transfer activities require proper synchronization between the processing/storage units supplying and receiving the material; e.g., it becomes compulsory that no other task is simultaneously performed in both units. To take into account material transfer operations, Ferrer-Nadal et al. (2008) extended a previous general precedence-based MILP model to account for non-zero transfer times. They also presented two alternative methods to avoid generating infeasible schedules.

Preventive maintenance tasks have not been treated in a consistent manner either. Sanmartí, Espuña, and Puigjaner (1997) addressed them in the framework of the scheduling problem of multipurpose batch plants in which equipment failure uncertainty is considered. According to this proposal, the execution and timing of maintenance activities are decision variables: preventive maintenance tasks are scheduled along with production batches to increase unit reliability (the more maintenance activities are carried out, the higher the schedule robustness). More recently, Harjunkoski and Sand (2008) addressed maintenance and production scheduling simultaneously. Based on information about which equipment to maintain, maintenance duration and the corresponding earliest and latest start-times (time window), maintenance requests are treated as jobs with fixed equipment assignments.

A few contributions (Ierapetritou, Hené, & Floudas, 1999; Janak, Floudas, Kallrath, & Vormbrock, 2006a; Janak, Lin, & Floudas, 2004; Maravelias & Grossmann, 2003b) have dealt with intermediate demand/order due-dates, which are pretty common in nowadays just-in-time industrial environments. Janak et al. (2006a) presented a more general proposal in which an order, having a given due-date, can be satisfied by means of one or more tasks, which are allowed to finish earlier or later than the deadline. Thus, the model is flexible and accounts for the early and late production of orders. In addition to due-times, Maravelias and Grossmann (2003b) accounted also for intermediate delivery of raw materials, which are generally assumed to be available at the beginning of the horizon in other works.

Finally, rolling horizon approaches, which are usually employed in the industrial practice, have been considered by the academic community to address hard scheduling or integrated production planning-scheduling problems (Dimitriadis, Shah, & Pantelides, 1997; Erdirik-Dogan & Grossmann, 2006; Janak et al., 2006a). The rolling horizon notion is also used in reactive and dynamic scheduling problems, where those tasks that started prior to the rescheduling point, and finish after it, need to be taken into account as frozen activities that still consume resources (Janak, Floudas, Kallrath, & Vormbrock, 2006b).

The *general* batch-scheduling problem considered in this paper is defined as follows:

Given are:

(i) A set of materials $m \in \mathbf{M}$; $\mathbf{M}^{NIS}$ and $\mathbf{M}^{ZW}$ are the subsets of materials with no intermediate storage and zero-wait policy, respectively; $\mathbf{M}^S = \mathbf{M} \backslash \{\mathbf{M}^{NIS} \cup \mathbf{M}^{ZW}\}$ is the subset of materials that can be stored in vessels.

(ii) A set of storage units (vessels) $j \in \mathbf{J}^S$ with capacity ζ_j^{MAX}; $\mathbf{J}^{DS}$ ($\mathbf{J}^{SS}$) is the subset of dedicated (shared) storage vessels; M_j^S is the subset of materials that can be stored in $j \in \mathbf{J}^{SS}$ and $\mathbf{J}_m^S$ is the subset of storage units where material m can be stored in.

(iii) A set of processing units $j \in \mathbf{J}^P$ with minimum β_j^{MIN} and maximum β_j^{MAX} processing capacity; $\mathbf{J}_m^I$ ($\mathbf{J}_m^O$) is the subset of processing units that can store material m as input (output); $\mathbf{M}_j^I$ ($\mathbf{M}_j^O$) is the subset of materials that can be stored in processing unit j as input (output) materials.

(iv) The subset of units (processing or storage) physically connected to each unit j, denoted by $\mathbf{J}_j$.

(v) A set of processing tasks $i \in \mathbf{I}^P$; $\mathbf{I}_j$ is the subset of tasks that can be carried out in unit j; the subset of processing tasks consuming (producing) material $m \in \mathbf{M}$ is denoted by $\mathbf{I}_m^C$ ($\mathbf{I}_m^P$); the subset of processing tasks consuming (producing) materials $m \in \mathbf{M}^{ZW}$ is denoted by $\mathbf{I}^{CZW}$ ($\mathbf{I}^{PZW}$).

(vi) A set of non-unary resources (e.g., utilities) $r \in \mathbf{R}$ with maximum availability ρ_r^{MAX}; $\mathbf{I}_r$ is the subset of tasks requiring resource r.

It is also assumed that the following processing data is given:

(vii) The mass fraction for the consumption of input and production of output materials by task i is denoted by γ_{im}, where $\gamma_{im} > 0$ for output and $\gamma_{im} < 0$ for input materials.

(viii) The processing time of task i in unit j is equal to a fixed term a_{ij} and a term that is proportional to the batch size of task i, with proportionality constant b_{ij}.

(ix) Similarly, the requirement of task i in unit j for resource r is described by constants f_{ijr} (fixed term) and g_{ijr} (variable term).

In addition, we consider the following issues:

(x) Maintenance operations can be carried out during the scheduling horizon. A maintenance activity concerns a specific unit and can have either predefined start and finish times or a time window within which it must be carried out.

(xi) A sequence dependent or independent changeover may be necessary between batches carried out consecutively in the same unit or between storage of different materials on the same shared vessel; a changeover activity may be performed by "unary" cleaning/sterilizing stations (e.g., CIP and SIP devices) and require additional non-unary resources (e.g., manpower, utilities, etc.).

(xii) Material transfer is not instantaneous and requires transfer devices and/or utilities.

(xiii) Raw materials may become available within the scheduling horizon through a set of deliveries and demand for finished products may be satisfied at intermediate due-times. The combined set of shipments (raw material deliveries and demands for finished products) is denoted by $\mathbf{L}$; the amount shipped is q_l, ($q_l > 0$ for deliveries of raw materials and $q_l < 0$ for finished product demand); a shipment $l \in \mathbf{L}$ occurs at time τ_l and corresponds to material $m(l)$.

(xiv) Since scheduling is carried out repeatedly in a rolling horizon fashion, at any point in time there may be previously scheduled tasks that will finish within the current scheduling horizon.

The paper is structured as follows. In Section 2, the basic concepts of the representation proposed by Giménez et al. (2009) are reviewed and in Section 3 their mixed-integer linear programming (MILP) formulation is presented in compact form. In Section 4, we discuss how their framework can be extended to address the general batch-scheduling problem and present the general MILP formulation. We close in Section 5 with three example problems that illustrate the modeling capabilities of the integrated framework.

2. Basic concepts

In this section, the five basic concepts of the approach presented by Giménez et al. (2009) are outlined.

2.1. Time representation

A new global continuous-time representation was introduced: a set of global time points $n \in \mathbf{N} = \{1, 2, \ldots, N\}$ spans the scheduling horizon from 0 to H delimiting a set of $N-1$ time intervals of unknown length, where interval n starts at time point n and ends at $n+1$. The timing of time point n is denoted by T_n. The novelty of this new representation is that tasks that do not consume (produce) an unstable material are not enforced to start (finish) exactly at a time point. In other words, a task assigned to start on a unit at time point n can actually start any time within interval n (a situation referred to as *late beginning*) as shown in Fig. 1a. Similarly, a task that is assigned to end at time point n can actually finish at any time within interval $n-1$ (a situation called *early end*). This is achieved via the introduction of "slack" variables that model the mismatch between time points and the actual beginning or end of a task (see

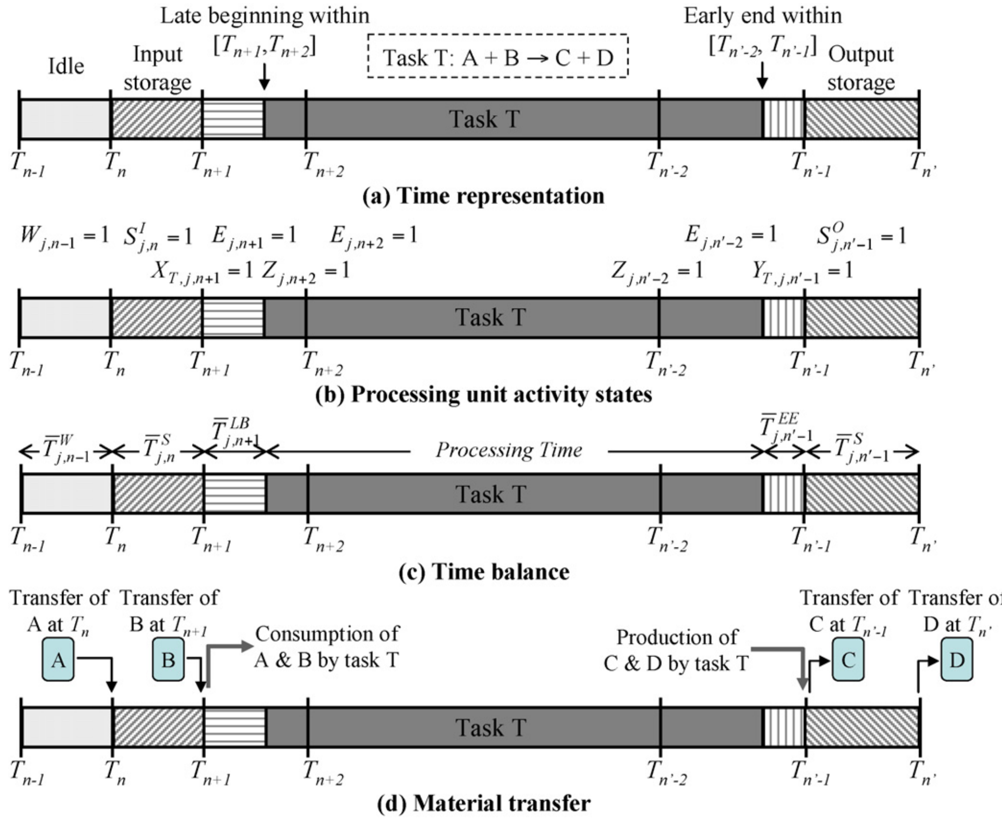


Fig. 1. Key concepts of the representation of Giménez et al. (2009).

Section 2.3). Thus, the new representation can potentially lead to MILP formulations with fewer time points.

2.2. Processing unit activity states

Unlike previous network-based models, a processing unit can be used for both carrying out processing tasks and storing input/output materials before/after the beginning/end of a task. Hence, a unit $j \in \mathbf{J}^P$ can be in three different activity states during time interval n (Fig. 1b): (a) idle state ($W_{j,n} = 1$); (b) storage state ($S_{j,n} = 1$); and (c) execution state ($E_{j,n} = 1$). If used for storage, then it can either be used for input ($S^I_{j,n} = 1$) or output ($S^O_{j,n} = 1$) materials. Regarding the execution state, a set of three execution variables is introduced to identify such a state. If a task $i \in \mathbf{I}_j$ is assigned to start in unit j within interval n (at or after time point n) then variable $X_{i,j,n} = 1$. If a task is assigned to end within interval $n-1$ (at or before time point n) then variable $Y_{i,j,n} = 1$. In addition, variable $Z_{j,n} = 1$ if a task which started before time point n is still being carried out in unit j at such point (n).

2.3. Time variables and time balance

To accurately account for a late beginning (early end) of a task in unit j after (before) time point n, two new "slack" variables were defined: (a) $\bar{T}^{LB}_{j,n}$ that denotes the lateness within interval n in starting a task, and (b) $\bar{T}^{EE}_{j,n}$ that denotes the earliness within interval $n-1$ in finishing a task. The length of time interval n during which unit j is in the storage or idle state is represented by variables $\bar{T}^S_{j,n}$ and $\bar{T}^W_{j,n}$, respectively (Fig. 1c). Note that the bar is used throughout this paper to denote duration. These new variables were used in three types of time balance constraints that are sufficient to correctly enforce the timing of the grid time points and the matching between time points and events.

2.4. Material transfer

In the first part of this series, the transfer of material m from unit j' to $j \in \mathbf{J}_{j'}$ at time point n was explicitly represented via "flow" variable $F_{m,j',j,n}$. In such a context, the concept of flow represents an instantaneous material transfer from a storage/processing unit to another physically connected storage/processing unit. The introduction of flow variables allowed to explicitly account for the connections between physical units, therefore enabling detailed modeling of complex process networks. In this paper, the treatment of transfer activities is extended to deal with non-instantaneous material transfers requiring resources such as connections and pumps. Furthermore, input/output materials do not have to be simultaneously transferred to/from a unit (Fig. 1d).

2.5. Material storage

Material storage constraints were considered for both storage and processing units. In storage units, only one material can be stored in any time interval and material balance constraints include only incoming and outgoing flows. The amount stored in a storage vessel j during time interval n is denoted by $I^S_{m,j,n}$. On the other hand, in processing units, multiple input/output materials can be simultaneously stored before/after a task starts/ends. The corresponding balances include the incoming and outgoing flows as well as production and consumption terms that correspond to the transformation of materials by processing tasks. The amount of input/output material m stored in processing unit j during time interval n is denoted by $I^I_{m,j,n}/I^O_{m,j,n}$.

3. Basic mathematical formulation

In this section, the MILP proposed by Giménez et al. (2009) is presented. In addition to the variables defined in the previous section, we also introduce: (a) continuous variables $B^S_{i,j,n}$, $B^P_{i,j,n}$ and $B^E_{i,j,n}$ to denote the batch size of task i that formally starts, is being processed and formally ends, respectively, in unit j at time point n; and (b) variable $Q_{r,n}$ to represent the total amount of non-unary resource r required during time interval n.

3.1. Execution-state constraints

Eq. (1) forces each processing unit to be in only one activity state at each time interval. Eq. (2) defines the state variable $E_{j,n}$ in terms of variables $Z_{j,n}$ and $X_{i,j,n}$, and Eq. (3) relates task beginnings and ends through variable $Z_{j,n}$. In turn, Eq. (4) expresses the state variable $S_{j,n}$ in terms of variables $S^I_{j,n}$ and $S^O_{j,n}$.

$$E_{j,n} + S_{j,n} + W_{j,n} = 1, \forall j \in \mathbf{J}^P,\ n < N \tag{1}$$

$$E_{j,n} = Z_{j,n} + \sum_{i \in \mathbf{I}_j} X_{i,j,n}, \forall j \in \mathbf{J}^P,\ n < N \tag{2}$$

$$Z_{j,n} = Z_{j,n-1} + \sum_{i \in \mathbf{I}_j} X_{i,j,n-1} - \sum_{i \in \mathbf{I}_j} Y_{i,j,n}, \forall j \in \mathbf{J}^P,\ n > 1 \tag{3}$$

$$S_{j,n} = S^I_{j,n} + S^O_{j,n}, \forall j \in \mathbf{J}^P,\ n < N \tag{4}$$

3.2. Slack time constraints

Inequality (5) allows $\bar{T}^{LB}_{j,n}$ to be positive if a task consuming stable materials starts at T_n. Similarly, inequality (6) allows $\bar{T}^{EE}_{j,n}$ to be positive if a task producing stable materials ends at T_n:

$$\bar{T}^{LB}_{j,n} \le H \sum_{i \in \mathbf{I}_j \setminus \mathbf{I}^{CZW}} X_{i,j,n}, \forall j \in \mathbf{J}^P,\ n < N \tag{5}$$

$$\bar{T}^{EE}_{j,n} \le H \sum_{i \in \mathbf{I}_j \setminus \mathbf{I}^{PZW}} Y_{i,j,n}, \forall j \in \mathbf{J}^P,\ n > 1 \tag{6}$$

3.3. Storage and idle periods constraints

Inequalities (7) and (8) relate variables representing storage and idle time intervals to their corresponding state variables:

$$\bar{T}^{S}_{j,n} \le H(S_{j,n}), \forall j \in \mathbf{J}^P,\ n < N \tag{7}$$

$$\bar{T}^{W}_{j,n} \le H(W_{j,n}), \forall j \in \mathbf{J}^p,\ n < N \tag{8}$$

In turn, the inequalities in expression (9) fix the lengths of both storage and idle time intervals.

$$T_{n+1} - T_n - H(1 - S_{j,n} - W_{j,n}) \le \bar{T}^S_{j,n} + \bar{T}^W_{j,n} \le T_{n+1} - T_n, \forall j \in \mathbf{J}^P,\ n < N \tag{9}$$

3.4. Time balance constraints

The matching between time points and task beginnings/ends is achieved via the time balance constraints (10)–(12). For a given time point n, inequality (10) expresses the partial balance of those time elements (i.e. processing times, slack times as well as storage and idle intervals) taking place before such point in unit j. Similarly, expression (11) models the balance of those time elements that occur after T_n in unit j. In turn, Eq. (12) represents the global time balance for each unit.

$$T_n \ge \sum_{1<n' \le n} \sum_{i \in \mathbf{I}_j} (a_{i,j} Y_{i,j,n'} + b_{i,j} B^E_{i,j,n'}) + \sum_{1<n' \le n} \bar{T}^{EE}_{j,n'} + \sum_{n'<n} (\bar{T}^{LB}_{j,n'} + \bar{T}^{S}_{j,n'} + \bar{T}^{W}_{j,n'}), \forall j \in \mathbf{J}^P,\ n > 1 \tag{10}$$

$$H - T_n \ge \sum_{n \le n' < N} \sum_{i \in \mathbf{I}_j} (a_{i,j} X_{i,j,n'} + b_{i,j} B^S_{i,j,n'}) + \sum_{n'>n} \bar{T}^{EE}_{j,n'} + \sum_{n \le n' < N} (\bar{T}^{LB}_{j,n'} + \bar{T}^{S}_{j,n'} + \bar{T}^{W}_{j,n'}), \forall j \in \mathbf{J}^P,\ n < N \tag{11}$$

$$\sum_{n>1} \bar{T}^{EE}_{j,n} + \sum_{n<N} (\bar{T}^{LB}_{j,n} + \bar{T}^{S}_{j,n} + \bar{T}^{W}_{j,n}) + \sum_{n>1} \sum_{i \in \mathbf{I}_j} (a_{i,j} Y_{i,j,n} + b_{i,j} B^E_{i,j,n}) = H, \forall j \in \mathbf{J}^P \tag{12}$$

Note that the matching between tasks and time points is achieved without resorting to big-M constraints.

3.5. Batch size constraints

Expressions (13)–(15) introduce bounds on the three different batch size variables. Eq. (16) ensures that they are equal for those time intervals associated with the formal execution of the same batch:

$$\beta^{MIN}_j Y_{i,j,n} \le B^E_{i,j,n} \le \beta^{MAX}_j Y_{i,j,n}, \forall i \in \mathbf{I}^P, j \in \mathbf{J}_i,\ n > 1 \tag{13}$$

$$B^S_{i,j,n} \le \beta^{MAX}_j X_{i,j,n}, \forall i \in \mathbf{I}^P, j \in \mathbf{J}_i,\ n < N \tag{14}$$

$$\sum_{i \in \mathbf{I}_j} B^P_{i,j,n} \le \beta^{MAX}_j Z_{j,n}, \forall j \in \mathbf{J}^P,\ n \tag{15}$$

$$B^S_{i,j,n} + B^P_{i,j,n} = B^P_{i,j,n+1} + B^E_{i,j,n+1}, \forall i \in \mathbf{I}^P, j \in \mathbf{J}_i,\ n < N \tag{16}$$

3.6. Material storage in storage units

Expression (17) represents the material balance for each storage vessel at each time point, where material transfers to/from a storage vessel are explicitly considered.

$$I^S_{m,j,n} = I^S_{m,j,n-1} - \sum_{j' \in (\mathbf{J}_j \cap (\mathbf{J}^S_m \cup \mathbf{J}^I_m))} F_{m,j,j',n} + \sum_{j' \in (\mathbf{J}_j \cap (\mathbf{J}^S_m \cup \mathbf{J}^O_m))} F_{m,j',j,n} \le \varsigma^{MAX}_{m,j}, \forall m \in \mathbf{M}^S, j \in \mathbf{J}^S_m,\ n \tag{17}$$

where $I^S_{m,j,0} = I^0_{m,j}$ is the given initial inventory.

For shared storage vessels, a binary variable $S^S_{m,j,n}$ was adopted to identify which specific material is being stored is being stored during each time interval during each time interval. Constraint (18) enforces that at most one material is stored at any time interval and constraint (19) bounds accordingly the inventory level.

$$\sum_{m \in \mathbf{M}^S_j} S^S_{m,j,n} \le 1, \forall j \in \mathbf{J}^{SS},\ n > 1 \tag{18}$$

$$I^S_{m,j,n} \le \varsigma^{MAX}_{m,j} S^S_{m,j,n}, \forall m \in \mathbf{M}^S_j,\ j \in \mathbf{J}^{SS},\ n > 1 \tag{19}$$

3.7. Material storage in processing units

The amount of input material m stored in processing unit j during time interval n, before being consumed by a task, is given by Eq. (20). The amount of an output material m stored in a processing unit j during time interval n, after being generated by an already finished task, is given by Eq. (21).

$$I^I_{m,j,n} = I^I_{m,j,n-1} + \sum_{j' \in (\mathbf{J}_j \cap (\mathbf{J}^S_m \cup \mathbf{J}^O_m))} F_{m,j',j,n} + \sum_{i \in (\mathbf{I}_j \cap \mathbf{I}^C_m)} \gamma_{i,m} B^S_{i,j,n}, \forall m \in \mathbf{M}, j \in \mathbf{J}^I_m, n \quad (20)$$

$$I^O_{m,j,n} = I^O_{m,j,n-1} + \sum_{i \in (\mathbf{I}_j \cap \mathbf{I}^P_m)} \gamma_{i,m} B^E_{i,j,n} - \sum_{j' \in (\mathbf{J}_j \cap (\mathbf{J}^S_m \cup \mathbf{J}^I_m))} F_{m,j,j',n}, \forall m \in \mathbf{M}, j \in \mathbf{J}^O_m, n > 1 \quad (21)$$

Inequalities (22) and (23) ensure that input and output materials, respectively, can be temporally stored in a processing unit only if the unit remains in the corresponding "storage" state during such time interval:

$$\sum_{m \in \mathbf{M}^I_j} I^I_{m,j,n} \le \beta^{MAX}_j S^I_{j,n}, \forall j \in \mathbf{J}^P, n < N \quad (22)$$

$$\sum_{m \in \mathbf{M}^O_j} I^O_{m,j,n} \le \beta^{MAX}_j S^O_{j,n}, \forall j \in \mathbf{J}^P, n < N \quad (23)$$

3.8. Utility constraints

The total amount of utility r consumed is calculated and bounded:

$$Q_{r,n} = Q_{r,n-1} + \sum_{i \in \mathbf{I}_r} \sum_{j \in \mathbf{J}_i} [f_{i,j,r}(X_{i,j,n} - Y_{i,j,n}) + g_{i,j,r}(B^S_{i,j,n} - B^E_{i,j,n})] \le \rho^{MAX}_r, \forall r \in \mathbf{R}, n \quad (24)$$

3.9. Continuous relaxation of some execution and state variables

As discussed in Part I of this series, despite execution and state variables being binary in nature, $Z_{j,n}$, $E_{j,n}$, $S_{j,n}$ and $W_{j,n}$ can be defined as non-negative continuous variables because they are forced to render binary values by Eqs. (1)–(4). Variable $Z_{j,n}$ is uniquely defined in Eq. (3), and since $X_{i,j,n}$ and $Y_{i,j,n}$ are defined as binary variables, $Z_{j,n}$ can only be integral at every feasible solution. Similarly, variable $E_{j,n}$ is uniquely defined in Eq. (2), and since variable $X_{i,j,n}$ is binary and variable $Z_{j,n}$ can only assume integral values, $E_{j,n}$ will also assume integral values. In turn, variable $S_{j,n}$ is uniquely defined in Eq. (4), and since $S^I_{j,n}$ and $S^O_{j,n}$ are strictly defined as binary variables, $S_{j,n}$ can only assume integral values (i.e. 0, 1 or 2) at every feasible solution. Furthermore, since the left-hand side in Eq. (1) must always be equal to 1, $E_{j,n}$, $S_{j,n}$ and $W_{j,n}$ are defined as non-negative continuous variables, and $E_{j,n}$ and $S_{j,n}$ can only assume integral values; then, variables $E_{j,n}$, $S_{j,n}$ and $W_{j,n}$, and, therefore, $Z_{j,n}$, will always assume binary values at every feasible solution.

4. General framework

4.1. Generalized tasks

In the context of process scheduling, non-productive activities that require resources need to be taken into account. For example, cleaning and maintenance activities usually take place within the scheduling horizon, while material transfer tasks occur routinely. All these activities can be viewed as tasks that require unary and non-unary resources and are characterized by start and end times. We propose to treat these activities as tasks in a uniform manner. We extend the set of tasks $\mathbf{I}$ to include the subsets $\mathbf{I}^C$ (cleaning tasks), $\mathbf{I}^M$ (maintenance tasks), $\mathbf{I}^T$ (transfer tasks), as well as the subset $\mathbf{I}^{PH}$ of those tasks (i.e. processing, cleaning, maintenance, and transfer tasks) that have been initiated in a previous scheduling horizon and that are still being executed at the beginning of the current one.

Likewise, the concept of *unit* is generalized to consider unary resources other than processing and storage units. Consequently, set $\mathbf{J}$ is extended by incorporating the following subsets: $\mathbf{J}^C$ (cleaning "units"), which includes unary cleaning resources such as cleaning-in-place or sterilization-in-place stations that carry out the cleaning/sterilization of other units; and $\mathbf{J}^T$ (transfer "units"), that includes unary transfer resources, each of which is physically composed by a set of transfer devices (i.e. pipes, pumps, valves, manifolds, etc.) and allows the flow of material between two processing/storage units. Below it is explained how these new concepts are integrated within our original approach.

4.1.1. Maintenance tasks

Basically, maintenance activities on unary resources introduce *downtime* periods, during which the use of processing, storage, cleaning, or transfer units is forbidden. Therefore, a maintenance task $i \in \mathbf{I}^M$ is assumed to take place once during the scheduling horizon on the unit $j(i)$. In this case, the task duration is fixed and equal to the planned maintenance time, using only the fixed term ($a_{i,j(i)}$) of the "processing" time of the corresponding task. A maintenance activity can have either predefined start and finish times (see Fig. 2a) or an allowed time window during which it can take place (see Fig. 2b). The formal beginning and end of maintenance tasks are assigned to two time points using the same approach followed for processing tasks. Furthermore, a fictitious batch size (equal to 1) is adopted in order to have a uniform treatment of all tasks.

Each maintenance task has to be carried out once:

$$\sum_{n<N} X_{i,j(i),n} = 1, i \in \mathbf{I}^{MT} \quad (25)$$

For the "bounded" case, the maintenance timing is enforced via inequalities (26) and (27), where τ^{MEB}_i and τ^{MLE}_i are the lower and upper limits, respectively, of the time window associated with the maintenance activity represented by task $i \in \mathbf{I}^M$. In turn, the maintenance timing for the "fixed" case is achieved via the same pair of constraints by replacing $\tau^{MEB}_i/\tau^{MLE}_i$ by the predetermined start (τ^{MB}_i)/finish (τ^{ME}_i) time (i.e. $\tau^{MLE}_i - \tau^{MEB}_i = a_{i,j(i)}$).

$$\tau^{MEB}_i \le T_n + \bar{T}^{LB}_{j(i),n} + H(1 - X_{i,j(i),n}), i \in \mathbf{I}^{MT}, n < N \quad (26)$$

$$\tau^{MLE}_i \ge T_n - \bar{T}^{EE}_{j(i),n} - H(1 - Y_{i,j(i),n}), i \in \mathbf{I}^{MT}, n > 1 \quad (27)$$

If no restriction ("free" case) is imposed on the execution period of the maintenance activity (being able to happen in any time period within the scheduling horizon), the pair of inequalities (26)–(27) should be removed. Note that as opposed to Sanmartí et al. (1997) maintenance activities were planned ahead of time to be carried out during the current scheduling horizon. They are not free to be scheduled or not.

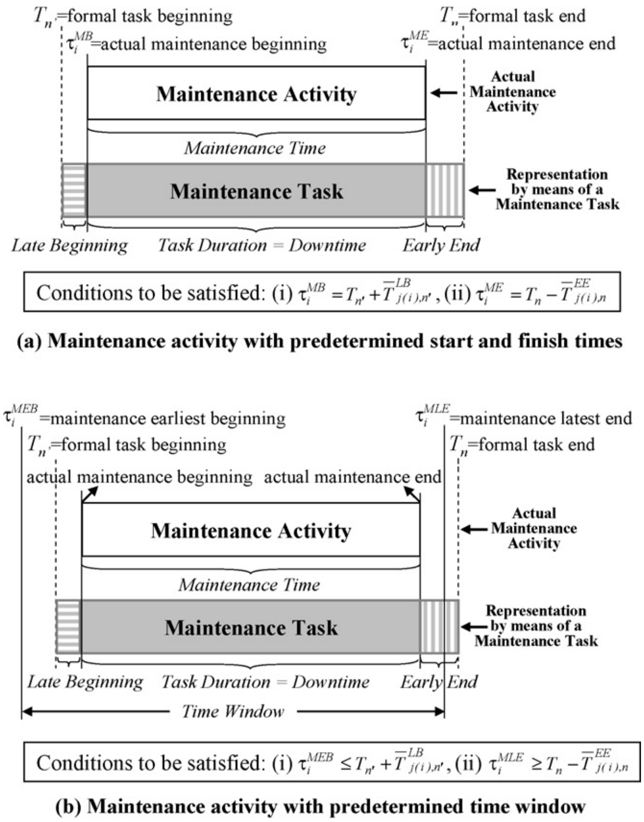


Fig. 2. Predefined preventive maintenance activity representation.

Additionally, the execution of a maintenance activity can require the use of non-unary resources. Since this requirement is known a priori we can account for it in Eq. (24) by considering only the fixed term of this expression.

4.1.2. Cleaning tasks

Cleaning activities can take place on both processing and shared storage units, and can be performed by cleaning or sterilization stations. In processing units, a cleaning task can be required between two consecutive processing tasks in case such a sequence needs a changeover operation. In turn, the storage of a given material in a shared vessel can require previous cleaning if an incompatible material was stored immediately before in the same vessel. Similarly to maintenance tasks, cleaning tasks are characterized by a fixed duration, a fixed requirement for non-unary resources, and a fictitious batch size (equal to 1).

Cleaning units are subject to the assignment constraints in Eqs. (1)–(3) but with no storage variables. Additionally, Eqs. (28) and (29) ensure that an appropriate cleaning device is engaged while a cleaning task is carried out on a processing or shared storage unit:

$$\sum_{j \in ((\mathbf{J}^P \cup \mathbf{J}^{SS}) \cap \mathbf{J}_i)} X_{i,j,n} = \sum_{j \in (\mathbf{J}^C \cap \mathbf{J}_i)} X_{i,j,n}, \ \forall i \in \mathbf{I}^C,\ n < N \tag{28}$$

$$\sum_{j \in ((\mathbf{J}^P \cup \mathbf{J}^{SS}) \cap \mathbf{J}_i)} Y_{i,j,n} = \sum_{j \in (\mathbf{J}^C \cap \mathbf{J}_i)} Y_{i,j,n}, \ \forall i \in \mathbf{I}^C,\ n > 1 \tag{29}$$

4.1.3. Transfer tasks

Regarding material transfers, we consider the flow of materials through transfer units. We define a transfer unit as a set of transfer devices (i.e. pipes, pumps, valves, manifolds, etc.) that physically connect two processing/storage units as shown in Fig. 3. Thus, $j(j', j'') \in \mathbf{J}^T$ is the transfer unit that connects the outlet of unit j' with the inlet of unit j'', $\mathbf{V}$ the set of transfer devices and $\mathbf{J}_v^T$ the subset of transfer units in which device v is a component. Therefore, a non-instantaneous transfer can be modeled as a transfer task $i \in \mathbf{I}^T$ being performed by a transfer unit. Then, $i(m,j)$ is the transfer task related to the flow of material m through transfer unit $j \in \mathbf{J}^T$. In this case, the batch size is representing the amount transferred and the proportionality constant b_{ij} denotes the variable transfer time (i.e. the inverse of the flow rate). Transfer units are also subject to the assignment constraints in Eqs. (1)–(3) but without considering storage variables. In addition, the inequality in expression (30) ensures that each transfer device is used by at most one transfer task during each time interval:

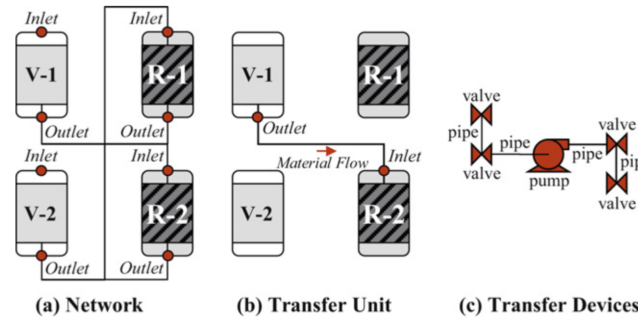


Fig. 3. Transfer unit/devices conceptualization.

$$\sum_{j \in \mathbf{J}_v^T} E_{j,n} \le 1, \ v \in \mathbf{V},\ n < N \tag{30}$$

On the other hand, processing and storage units must be in a storage state while a material is being transferred through a transfer unit from(to) another site. Fig. 4 depicts the corresponding storage state in which a given processing/storage unit remains during an incoming or outgoing transfer. Inequalities (31)–(33) relate the execution state of a transfer unit to the corresponding storage state of the processing/storage units that are connected by such transfer unit (assuming unit j' is connected to unit j via a single transfer unit $j''(j', j)$.

$$\sum_{j' \in \mathbf{J}_j} E_{j''(j',j),n} \le S^I_{j,n}, \ \forall j \in \mathbf{J}^P,\ n < N \tag{31}$$

$$\sum_{j' \in \mathbf{J}_j} E_{j''(j,j'),n} \le S^O_{j,n}, \ \forall j \in \mathbf{J}^P,\ n < N \tag{32}$$

$$\sum_{j' \in \mathbf{J}_j} E_{j''(j',j),n} + \sum_{j' \in \mathbf{J}_j} E_{j''(j,j'),n} \le S_{j,n}, \ \forall j \in \mathbf{J}^S,\ n < N \tag{33}$$

Thus, this representation allows accounting for non-instantaneous material transfers having a duration which is inversely proportional to the pumping rate. Moreover, by linking transfer tasks to storage states of the source and destination units, the use of these units (processing and storage units) by other activities is avoided.

4.1.4. Previous horizon tasks

As pointed out before, rolling horizon approaches are used when accounting for tasks that were scheduled in the preceding scheduling horizon and might still be running at the beginning of the current one. Thus, the two horizons need to be spliced. When addressing their lower level scheduling model Janak et al. (2006a) assumed that all the tasks of the previous sub-period had to be finished before its end, but allowed tasks of the current sub-period to be scheduled in the idle portions of the preceding horizon. On the contrary, this work assumes that certain processing tasks of the previous horizon may possibly be executing at the beginning of

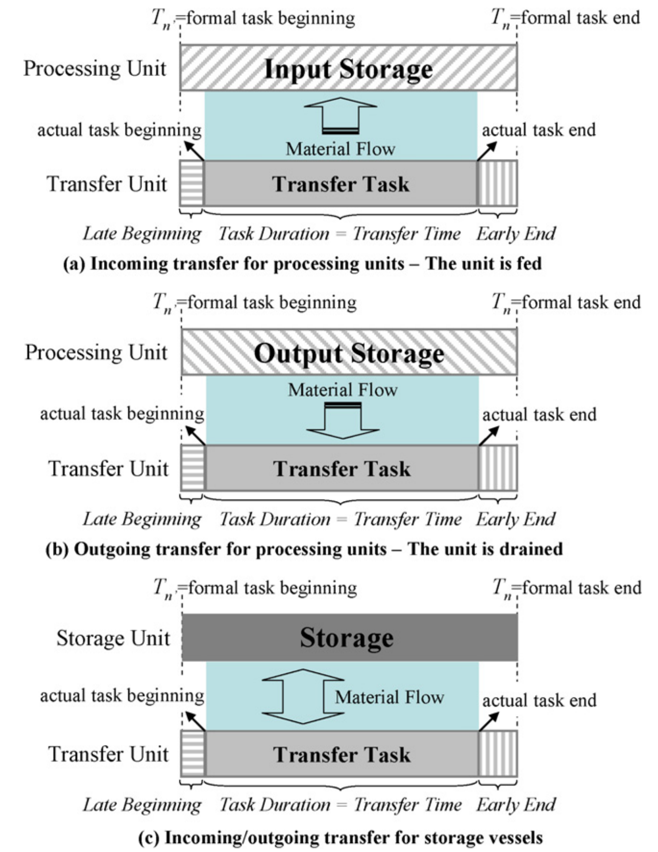


Fig. 4. Relations between transfer tasks and storage states.

the current one and will compete for resources. This is also taken into account by Janak et al. (2006b) when dealing with reactive scheduling problems.

Previous horizon tasks $i \in \mathbf{I}^{PH}$ are modeled as tasks having a fixed duration, a fixed batch size and starting exactly at time 0 ($n = 1$), which means that no late beginning is considered (see Fig. 5). Their duration is equal to the "processing" time that remains to be executed in the current horizon. In general, they can also demand a fixed amount of non-unary resources; therefore, they can compete for them with those tasks to be scheduled in the current scheduling period. Moreover, the production/transfer of materials when "previous horizon" processing/transfer tasks are finished is considered. Since this type of tasks can be easily incorporated into the representation, the scheduling activity can be seen as taking place over a rolling horizon, as it occurs in practice. A task $i \in \mathbf{I}^{PH}$ takes place on unit $j(i)$ once and starts at $n = 1$:

$$X_{i,j(i),1} = 1, X_{i,j(i),1<n<N} = 0, i \in \mathbf{I}^{PH} \qquad (34)$$

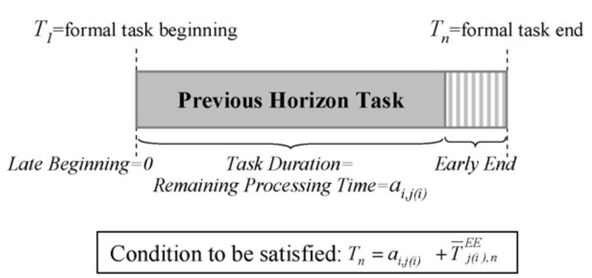


Fig. 5. Previous horizon task representation.

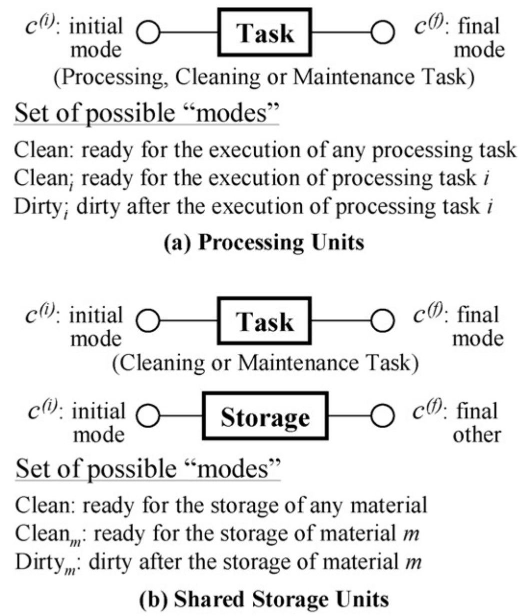


Fig. 6. Modes for processing and shared storage units.

4.2. Unit modes

In order to tackle changeover operations, we introduce a set $\mathbf{C}_j$ of *modes* defined for both processing and shared storage units $j \in (\mathbf{J}^P \cup \mathbf{J}^{SS})$. In the context of this contribution, these modes identify the "cleanness" of a given processing or shared storage unit at certain time point, unlike unit states that are used to describe the activity status of a unit during a given time interval. Thus, the concept of unit mode proposed in this section is equivalent to the one of unit state (Ustate) introduced by Crooks (1992) and later used by Barbosa-Póvoa and Macchietto (1994b) or similar to the notion of equipment conditions proposed by Castro, Barbosa-Póvoa, Matos, and Novais (2004) to handle changeovers.

In the case of processing units, a mode is "consumed" and "produced" each time a task (processing, cleaning or maintenance task) formally starts and ends, respectively. In turn, a task can start only if the corresponding processing unit is in a compatible mode. In the case of shared storage units, every storage interval and every task (cleaning or maintenance) consumes a mode when it starts and produces another one when it ends. Similarly to processing units, a task/material can start/be stored only if the corresponding shared vessel is in a compatible mode.

Fig. 6 shows the set of modes corresponding to processing (a) and shared storage (b) units, while Fig. 7 depicts the subsets of "compatible" initial modes and the final mode for a particular task/storage interval. In general, we assume that the initial and final modes of a processing task are different, so we can model situations where cleaning is needed between two consecutive batches of the same task, as it is required in many dairy, food, biotechnology and pharmaceutical industries. If this is not the case, then a task can have the same initial and final mode. Situations where no changeover is required between batches of a subset of tasks can also be modeled by considering a common subset of initial modes for a family of tasks. Cleaning tasks enable us the transition from one mode to another, thus they always have different initial and final modes. Finally, since maintenance activities can follow and precede any task, maintenance tasks are flexible to consume any mode and produce an always compatible mode (e.g., "Clean" mode). Note that this idea can be extended to model other tasks that can consume any mode (e.g., to model a standardized cleaning task which is needed

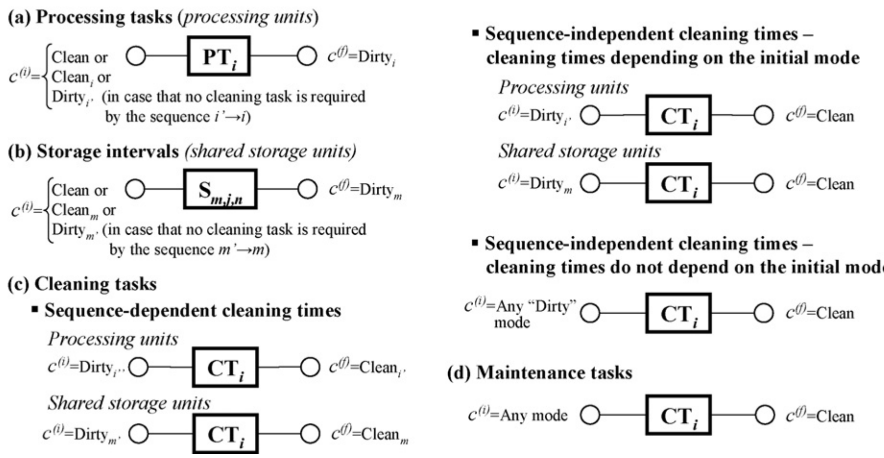


Fig. 7. Particular cases of initial and final modes.

prior to the beginning of different tasks). Next we discuss how the transition between modes is modeled.

Variable $A_{c,j,n}$ is introduced to monitor the availability of mode $c \in \mathbf{C}_j$ in unit j at time point n ($A_{c,j,n} = 1$ if available) and variables $K^C_{c,j,n}$ and $K^P_{c,j,n}$ to denote the "consumption" and "production" of mode $c \in \mathbf{C}_j$ in unit j at time point n ($K^C_{c,j,n} = 1$ if consumed, $K^P_{c,j,n} = 1$ if produced). Eq. (35) is the general balance for modes in both processing and shared storage units.

$$A_{c,j,n} = A_{c,j,n-1} - K^C_{c,j,n} + K^P_{c,j,n}, \forall j \in (\mathbf{J}^P \cup \mathbf{J}^{SS}), c \in \mathbf{C}_j,\ n \qquad (35)$$

where $A_{c,j,0} = A^0_{c,j}$ is the availability of mode $c \in \mathbf{C}_j$ in unit j at the beginning of the scheduling horizon.

In the case of processing units, the values of variables $K^C_{c,j,n}$ and $K^P_{c,j,n}$ are constrained by (36)–(38). Since a given task i can consume any of the modes included in the subset $c \in \mathbf{C}^C_i$, inequalities (36) and (37) ensure that at most one mode (the active one) is consumed in each unit at any time point.

$$\sum_{c \in (\mathbf{C}_j \cap \mathbf{C}^C_i)} K^C_{c,j,n} \geq X_{i,j,n}, \quad \forall i \in \mathbf{I}_j, j \in \mathbf{J}^P,\ n < N \qquad (36)$$

$$\sum_{c \in \mathbf{C}_j} K^C_{c,j,n} \leq 1, \quad \forall j \in \mathbf{J}^P, n \qquad (37)$$

In turn, a mode $c \in \mathbf{C}^C_j$ is generated in unit j at time point n when a task i, that belongs to the subset $\mathbf{I}^P_c$ of tasks producing mode c, ends in j at T_n.

$$K^P_{c,j,n} = \sum_{i \in (\mathbf{I}_j \cap \mathbf{I}^P_c)} Y_{i,j,n}, \quad \forall j \in \mathbf{J}^P,\ n \qquad (38)$$

In the case of shared vessels, inequalities (36) and (37) need to be adjusted to account for shared storage units, inequality (39) is incorporated and Eq. (38) is replaced by (40).

$$\sum_{c \in \mathbf{C}^C_m} K^C_{c,j,n} \geq S^S_{m,j,n}, \quad \forall j \in \mathbf{J}^{SS}, m \in \mathbf{M}^S_j,\ n < N \qquad (39)$$

$$K^P_{c,j,n} = S^S_{m(c),j,n-1} + \sum_{i \in (\mathbf{I}_j \cap \mathbf{I}^P_c)} Y_{i,j,n}, \quad \forall j \in \mathbf{J}^{SS},\ n \qquad (40)$$

where $\mathbf{C}^C_m$ is the subset of modes that can be consumed by the storage of material m, and $m(c)$ is the material whose storage during a time interval gives rise to mode c.

It is important to remark here that the explicit consideration of unit modes allows us to model different transition times in addition to cleaning times that depend on the initial and final status of the units (e.g., the time required to increase/decrease the temperature of a batch oven that operates at different conditions).

4.3. Material shipments

The last issue involves the treatment of raw material deliveries at certain release-times as well as the demand satisfaction of finished products at given due-times (see Fig. 8). These two situations are treated in a unified way: they are both considered shipments $l \in \mathbf{L}$, which are characterized by a shipment time τ_l and a shipment quantity q_l. It is assumed $q_l > 0$ for raw material deliveries and $q_l < 0$ for demand satisfaction. To simplify the presentation, only instantaneous shipments to/from a single storage unit are considered. However, we can easily account for non-instantaneous material shipments to/from multiple destinations/sources (storage and processing units).

Binary variable $O_{l,j,n}$ is introduced to denote the execution of shipment l to/from storage unit j at time point n. Clearly, each shipment occurs at exactly one time point and concerns one storage

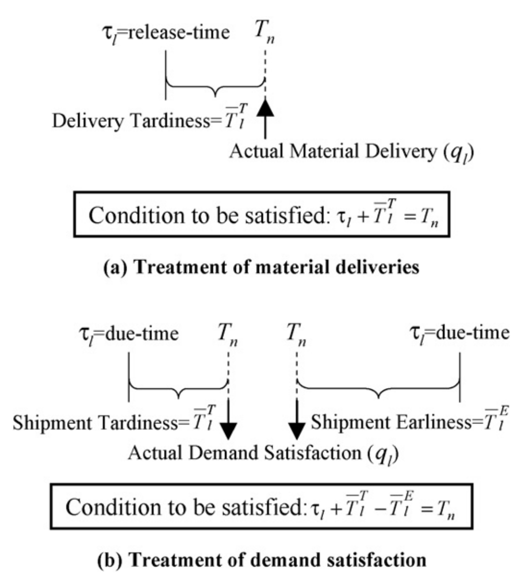


Fig. 8. Material shipments representation.

unit:

$$\sum_{j\in \mathbf{J}^S_{m(l)}} \sum_{n\in \mathbf{N}} O_{l,j,n} = 1, \quad l\in \mathbf{L} \tag{41}$$

where $m(l)$ is the material that corresponds to shipment l.

The timing of a shipment is achieved via the inequalities in expression (42), where $\bar{T}^E_l$ and $\bar{T}^T_l$ are the earliness and tardiness of shipment l (if allowed):

$$T_n - H(1 - \sum_{j\in \mathbf{J}^S_{m(l)}} O_{l,j,n}) \le \tau_l + \bar{T}^T_l - \bar{T}^E_l \le T_n + H(1 - \sum_{j\in \mathbf{J}^S_{m(l)}} O_{l,j,n}),$$
$$l\in \mathbf{L}, n\in \mathbf{N} \tag{42}$$

Note that deliveries of raw materials are typically fixed, which means that $\bar{T}^T_l = \bar{T}^E_l = 0$, but in some particular situations, some delay is permitted in the actual delivery. On the other hand, demand satisfaction can occur either earlier or later than the due-time.

4.4. Extensions on basic constraints

Finally, in this section we briefly explain how the constraints of the basic mathematical formulation are extended to tackle the issues discussed above.

4.4.1. Execution-state constraints

In the case of storage units ($j\in \mathbf{J}^S$), Eqs. (1)–(3) remain without change. For shared vessels, the storage state is activated by Eq. (43). In the case of cleaning and transfer units ($j\in \{\mathbf{J}^C \cup \mathbf{J}^T\}$), storage state is eliminated from Eq. (1), while Eqs. (2) and (3) remain without change. Eq. (4) is not considered in any of these cases.

$$S_{j,n} = \sum_{m\in \mathbf{M}^S_j} S^S_{m,j,n}, \quad \forall j\in \mathbf{J}^{SS}, \ n < N \tag{43}$$

4.4.2. Slack time constraints

Inequalities (5) and (6) are generalized in constraints (44) and (45), respectively:

$$\bar{T}^{LB}_{j,n} \le H \sum_{i\in \mathbf{I}_j, i\notin(\mathbf{I}^{CZW}\cup \mathbf{I}^{TZW}\cup \mathbf{I}^{PH})} X_{i,j,n}, \quad \forall j, \ n < N \tag{44}$$

$$\bar{T}^{EE}_{j,n} \le H \sum_{i\in \mathbf{I}_j, i\notin(\mathbf{I}^{PZW}\cup \mathbf{I}^{TZW})} Y_{i,j,n}, \quad \forall j, \ n > 1 \tag{45}$$

For previous horizon tasks ($i\in \mathbf{I}^{PH}$) and transfer tasks involving the flow of an unstable material ($i\in \mathbf{I}^{TZW}$), the late beginning is not considered. Similarly, the early end is also set to zero when a transfer task involves the flow of an unstable material.

4.4.3. Storage and idle periods constraints

For storage units ($j\in \mathbf{J}^S$), Eqs. (7)–(9) remain without change. For cleaning and transfer units ($j\in \{\mathbf{J}^C \cup \mathbf{J}^T\}$) Eq. (7) is removed and storage state is eliminated from Eq. (9).

4.4.4. Time balance constraints

In the case of storage units ($j\in \mathbf{J}^S$), Eqs. (10)–(12) remain the same. For cleaning and transfer units ($j\in \{\mathbf{J}^C \cup \mathbf{J}^T\}$), the storage state is eliminated from Eqs. (10)–(12).

4.4.5. Batch size constraints

For maintenance, cleaning, transfer, and previous horizon tasks ($i\in \{\mathbf{I}^M \cup \mathbf{I}^C \cup \mathbf{I}^T \cup \mathbf{I}^{PH}\}$), Eqs. (13)–(16) remain without change in their general form, but the meaning of “bounds” on the batch sizes differs depending on the specific case. For maintenance and cleaning tasks, lower and upper bounds are equal to the fictitious batch size (=1). In the case of transfer tasks, batch sizes are upper bounded by the minimum of the maximum capacities of the source and destination units. On the other hand, lower bounds are to be fixed based on the particular characteristics of the facility being considered. Finally, for previous horizon tasks lower and upper bounds are equal to $B^0_{i,j}$, representing the fixed batch size (real or fictitious).

4.4.6. Material storage in storage units

Expressions (17) and (19) are generalized in constraints (46) and (47), respectively:

$$I^S_{m,j,n} = I^S_{m,j,n-1} - \sum_{j'\in(\mathbf{J}^S_m\cup \mathbf{J}^I_m)} B^S_{i(m,j''),j''(j,j'),n} + \sum_{j'\in(\mathbf{J}^S_m\cup \mathbf{J}^O_m)} B^E_{i(m,j''),j''(j',j),n}$$
$$+ \sum_{l\in \mathbf{L}_m} q_l O_{l,j,n} \le \varsigma^{MAX}_{m,j}, \quad \forall m\in \mathbf{M}^S, \ j\in \mathbf{J}^S_m, \ n \tag{46}$$

with Eq. (46) accounts for incoming and outgoing material transfers as well as material shipments to/from a storage unit.

$$I^S_{m,j,n} \le \varsigma^{MAX}_{m,j} S_{j,n}, \quad \forall m\in \mathbf{M}^S, \ j\in \mathbf{J}^S_m, \ n < N \tag{47}$$

In turn, constraint (18) remains without change.

4.4.7. Material storage in processing units

Eqs. (20) and (21) are generalized in constraints (48) and (49), respectively:

$$I^I_{m,j,n} = I^I_{m,j,n-1} + \sum_{j'\in(\mathbf{J}^S_m\cup \mathbf{J}^O_m)} B^S_{i(m,j''),j''(j',j),n} + P_{m,j,n}$$
$$+ \sum_{i\in(\mathbf{I}_j\cap \mathbf{I}^C_m)} \gamma_{i,m} B^S_{i,j,n}, \quad \forall m, j\in \mathbf{J}^I_m, \ n \tag{48}$$

$$I^O_{m,j,n} = I^O_{m,j,n-1} + \sum_{i\in(\mathbf{I}_j\cap \mathbf{I}^P_m)} \gamma_{i,m} B^E_{i,j,n} - \sum_{j'\in(\mathbf{J}^S_m\cup \mathbf{J}^I_m)} B^E_{i(m,j''),j''(j,j'),n}$$
$$- P_{m,j,n}, \quad \forall m, j\in \mathbf{J}^O_m, \ n \tag{49}$$

In addition to considering material consumption and production as well as incoming and outgoing material transfers, these inventory balances take into account the amount of material m that remains in processing unit $j\in \mathbf{J}^P$ between the execution of two consecutive processing tasks. This particular case is modeled by variable $P_{m,j,n}$ that represents the amount of output material m, produced in unit j by a predecessor task, which becomes an input material (at T_n) to be consumed by a successor task taking place in the same unit later on. A similar approach is employed by Prasad, Maravelias, and Kelly (2006) to deal with materials involved in washcast operations. On the contrary, the representation proposed by Castro et al. (2005) requires an artificial transfer within the same processing unit to distinguish the case where the material is being produced from that where the material is being consumed. Thus, though the material is already inside the unit, it needs to be fictitiously transferred.

Regarding non-instantaneous material transfers, the batch size of task $i(m,j'')\in \mathbf{I}^T$ represents the amount of material $m\in \mathbf{M}$ transferred through transfer unit $j''(j',j)\in \mathbf{J}^T$ from unit $j'\in(\mathbf{J}^P\cup \mathbf{J}^S)$ to $j\in(\mathbf{J}^P\cup \mathbf{J}^S)$. On the other hand, Eqs. (22) and (23) remain without change.

4.4.8. Utility constraints

Eq. (24) does not need to be modified.

Table 1
Processing time coefficients, batch size bounds, utility requirements and maximum utility availability for Examples 1–3.

Processing Task $i \in \mathbf{I}^P$	Processing Unit $j \in \mathbf{J}^P$	$a_{i,j}$ (h)	$b_{i,j}$ (h/kg)	β_j^{MIN}(kg)	β_j^{MAX}(kg)	Utility $r \in \mathbf{R}$	$f_{i,j,r}$ (kg/min)	$g_{i,j,r}$ (kg/min kg)	ρ_r^{MAX}(kg/min)
T1	R-101	0.5	0.025	40	80	HS	6	0.25	30
	R-102	0.5	0.04	25	50	HS	4	0.25	30
T2	R-101	0.75	0.0375	40	80	CW	4	0.3	30
	R-102	0.75	0.06	25	50	CW	3	0.3	30
T3	R-103	0.25	0.0125	40	80	HS	8	0.2	30
T4	R-103	0.5	0.025	40	80	CW	4	0.5	30

Table 2
Material prices, storage capacities, and initial inventories for Examples 1–3.

Material $m \in \mathbf{M}$	π_m ($/kg)	Storage Unit $j \in \mathbf{J}^{DS}$	$\varsigma_{m,j}^{MAX}$(kg)	$I_{m,j}^0$(kg)
RM1	0[a],10[b]	V-101	UIS	50[a],AA[b]
RM2	0[a],15[b]	V-102	UIS	50[a],AA[b]
INT1	0[a],25[b]	V-103	50	0
INT2	0	NIS	0	0
INT3	0	V-104	50	0
P1	30	V-105	UIS	0
P2	40	V-106	UIS	0

AA = available as and when required, NIS = no intermediate storage, UIS = unlimited intermediate storage.
[a] Data corresponding to Example 1.
[b] Data corresponding to Examples 2 and 3.

Elements
Processing Units: R-101, R-102, R-103
Tasks: T1, T2, T3, T4
Storage Units: V-101, V-102, V-103, V-104, V-105, V-106
Materials: RM1, RM2, INT1, INT2, INT3, P1, P2
Utilities: Hot Steam (HS), Cooling Water (CW)

Logical Connections
Task/Processing Unit: T1/R-101, T1/R-102, T2/R-101, T2/R-102, T3/R-103, T4/R-103
Task/Utility: T1/HS, T2/CW, T3/HS, T4/CW
Material/Storage Unit: RM1/V-101, RM2/V-102, INT1/V-103, INT2/-, INT3/V-104, P1/V-105, P2/V-106

Stoichiometric Relations
T1 0.8 RM1 + 0.2 INT1 → INT3
T2 RM2 → 0.3 INT1 + 0.7 INT2
T3 INT3 → P1
T4 0.6 INT2 + 0.4 INT3 → P2

Plant Topology

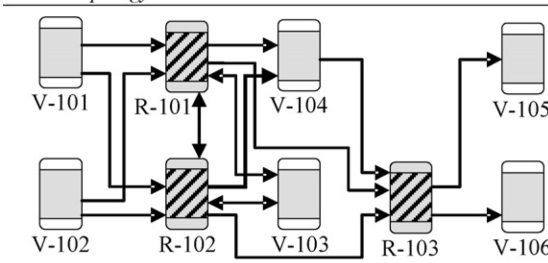


Fig. 9. Example of a simple multipurpose facility: elements, connections and task information. Four tasks consuming two types of utilities are carried out in three processing units. Two products are obtained from two raw materials and three intermediates. Six dedicated vessels are available.

4.4.9. *Objective function*

Besides traditional performance measures as profit maximization and makespan minimization, now it is possible to minimize changeover and/or transfers times/costs, to minimize tardiness or earliness, etc.

5. Representation capabilities of the general framework

To illustrate the modeling capabilities of the integrated framework we present three examples based upon the motivating multipurpose facility studied in the first part of the series (Giménez et al., 2009). The core structure of this facility is shown in Fig. 9. Basic processing task information and material related data for Examples 1–3 can be found in Tables 1 and 2, respectively. The examples were solved with the aim of getting the best possible schedules in cases where existing approaches cannot even obtain a feasible solution. The resulting MILP formulations were implemented in GAMS/CPLEX 10.2 (using two threads) on a Pentium D (2.80 GHz) PC with 1 GB of RAM and were solved to optimality (zero optimality gap). Regarding the solution method, we adopted the iterative procedure shared among all continuous-time formulations, where the same problem is solved by incrementing the number of time points at each iteration. The iterative procedure was stopped when a resource limit of 1000 CPU seconds was reached.

5.1. *Example 1*

This example deals with the treatment of previous horizon and maintenance tasks as well as material shipments. Particularly, two previous horizon tasks and one maintenance task are considered to be performed in processing units. Information related to these tasks can be found in Table 3. Note that PHT1 is a processing task of type T2 that began in the previous horizon and is still being executed in unit R-101. In turn, PHT2 is a maintenance task with predefined start and finish times. Moreover, two finished product demands and one raw material delivery take place within the current scheduling horizon. Table 4 presents data regarding material shipments. It is assumed that the delivery can occur after the raw material is available while demand cannot be satisfied later than its due-time. The objective function is the maximization of the revenue from the sales of extra

Table 3
Data regarding previous horizon and maintenance tasks for Example 1.

Previous horizon tasks

Task $i \in \mathbf{I}^{PH}$	Processing Unit $j \in \mathbf{J}^P$	$a_{i,j}$ (h)	$B_{i,j}$ (kg)	Utility $r \in \mathbf{R}$	$f_{i,j,r}$ (kg/min)	$g_{i,j,r}$ (kg/min kg)
PHT1	R-101	0.85	40	CW	4	0.3
PHT2	R-102	1.25	–	–	–	–

Maintenance task

Task $i \in \mathbf{I}^M$	Processing Unit $J \in \mathbf{J}^P$	$a_{i,j}$ (h)	τ_i^{MB}(h)	τ_i^{ME}(h)	Utility $r \in \mathbf{R}$	$f_{i,j,r}$ (kg/min)
MT1	R-103	0.5	5.0	5.5	–	–

Table 4
Material quantities and release/due-times for deliveries/demands in Example 1.

Shipment $l \in \mathbf{L}$	Material $m(l) \in \mathbf{M}$	τ_l (h)	q_l (kg)
L1	RM1	3	35
L2	P2	6	−25
L3	P1	8	−40

production over a time horizon of 8 h:

$$\max \sum_{m \in \mathbf{M}^D} \sum_{j \in \mathbf{J}_m^S} \pi_m I_{m,j,N}^S \tag{50}$$

Table 5 shows the computational results obtained from the implementation of the resulting MILP formulation. It can be seen that eight global time points (seven time intervals) are minimally required to achieve the best solution within the 1000 s resource limit.

The Gantt chart and the utility consumption profile of the best solution for Example 1 are depicted in Fig. 10. The Gantt chart shows the assignment of previous horizon tasks to the corresponding processing units at the beginning of the scheduling horizon, and the assignment of the maintenance task to time interval number 5, which coincides exactly with the predefined period for the maintenance activity. It can also be seen that RM1 delivery occurs late at T_4, P2 demand is satisfied early at T_6 and P1 demand is satisfied in time at T_8. The amount of each material stored in a given processing or storage unit is shown for each time interval. Instantaneous transfers are not drawn to simplify the figure. It is important to highlight here that existing approaches cannot find a feasible solution for this problem since the partial storage of input and output materials in processing units is not allowed.

Table 8
Transfer task data for Example 3.

Transfer Task $i \in \mathbf{I}^T$	Transfer Unit $j \in \mathbf{J}^T$	Material $m(i) \in \mathbf{M}$	Transfer Time (h/kg)
TT1	T-101	RM1	0.0015
TT2	T-102	RM2	0.0015
TT3	T-103	INT1	0.0012
TT4	T-104	INT1	0.0012
TT5	T-105	INT3	0.0012
TT6	T-106	INT2	0.0012
TT7	T-106	INT3	0.0012
TT8	T-107	INT3	0.0012
TT9	T-108	P1	0.0015
TT10	T-109	P2	0.0015

5.2. Example 2

This example considers sequence-(in)dependent changeover operations between the execution of two batches of different tasks taking place in the same processing unit. Table 6 gives the set of cleaning tasks, the associated task sequences, the initial(final) mode before(after) the execution of a cleaning task, and the corresponding cleaning times and costs. It also gives information regarding the processing units in which each cleaning task can take place and the cleaning units that perform them. Note that cleaning tasks CT1–CT4 refer to sequence-dependent changeover operations while CT5 represents a sequence-independent changeover. Two cleaning units are employed and no additional utility is required other than the ones that pertain to the cleaning systems. On the other hand, all processing units are in the “Clean” mode at the beginning of the scheduling horizon ($A^0_{Clean,j} = 1, \quad \forall j \in \mathbf{J}^P$). In this case, we seek to

Table 5
Model and solution statistics for Example 1.

N	CPU time (s)	Nodes	RMILP ($)	MILP ($)	Binary variables	Continuous Variables	Constraints	Non-zeros
5	No feasible solution exists							
6	1.08	158	3927.5	906.67	125	710	767	3671
7	11.9	2854	3927.5	906.67	150	849	906	4540
8	**24.2**	**3135**	**3927.5**	**3253.3**	**175**	**988**	**1045**	**5483**
9	240	26517	3927.5	3253.3	200	1127	1184	6480
10	1969	188771	3927.5	3253.3	225	1266	1323	7567

Table 6
Cleaning task data for Example 2.

Cleaning Task $i \in \mathbf{I}^C$	Task Sequence $i' \in \mathbf{I}^P \to i'' \in \mathbf{I}^P$	Initial Mode $c^{(i)}$	Final Mode $c^{(f)}$	Processing Unit $j \in \mathbf{J}^P$	Cleaning Unit $j \in \mathbf{J}^C$	Cleaning Time (h)	Cleaning Cost ($)
CT1	T1 → T2	$Dirty_{T1}$	$Clean_{T2}$	R-101	C-101	0.15	80
CT2	T2 → T1	$Dirty_{T2}$	$Clean_{T1}$	R-101	C-101	0.18	100
CT3	T1 → T2	$Dirty_{T1}$	$Clean_{T2}$	R-102	C-101	0.10	45
CT4	T2 → T1	$Dirty_{T2}$	$Clean_{T1}$	R-102	C-101	0.12	60
CT5	T3 → T4 T4 → T3	$Dirty_{T3}$ $Dirty_{T4}$	Clean	R-103	C-102	0.15	80

Table 7
Model and solution statistics for Example 2.

N	CPU time (s)	Nodes	RMILP ($)	MILP ($)	Binary variables	Continuous variables	Constraints	Non-zeros
3	No feasible solution exists							
4	0.13	0	1475.3	1420.7	111	675	751	2983
5	0.30	9	3455.6	2730.8	149	885	973	4127
6	1.55	389	4899.4	2730.8	187	1095	1195	5351
7	16.4	3272	5719.5	2730.8	225	1192	1417	6655
8	**47.0**	**4841**	**6138.7**	**3093.0**	**263**	**1515**	**1639**	**8039**
9	189	16377	6348.4	3093.0	301	1725	1861	9503
10	1089	79699	6497.7	3093.0	339	1935	2083	11047

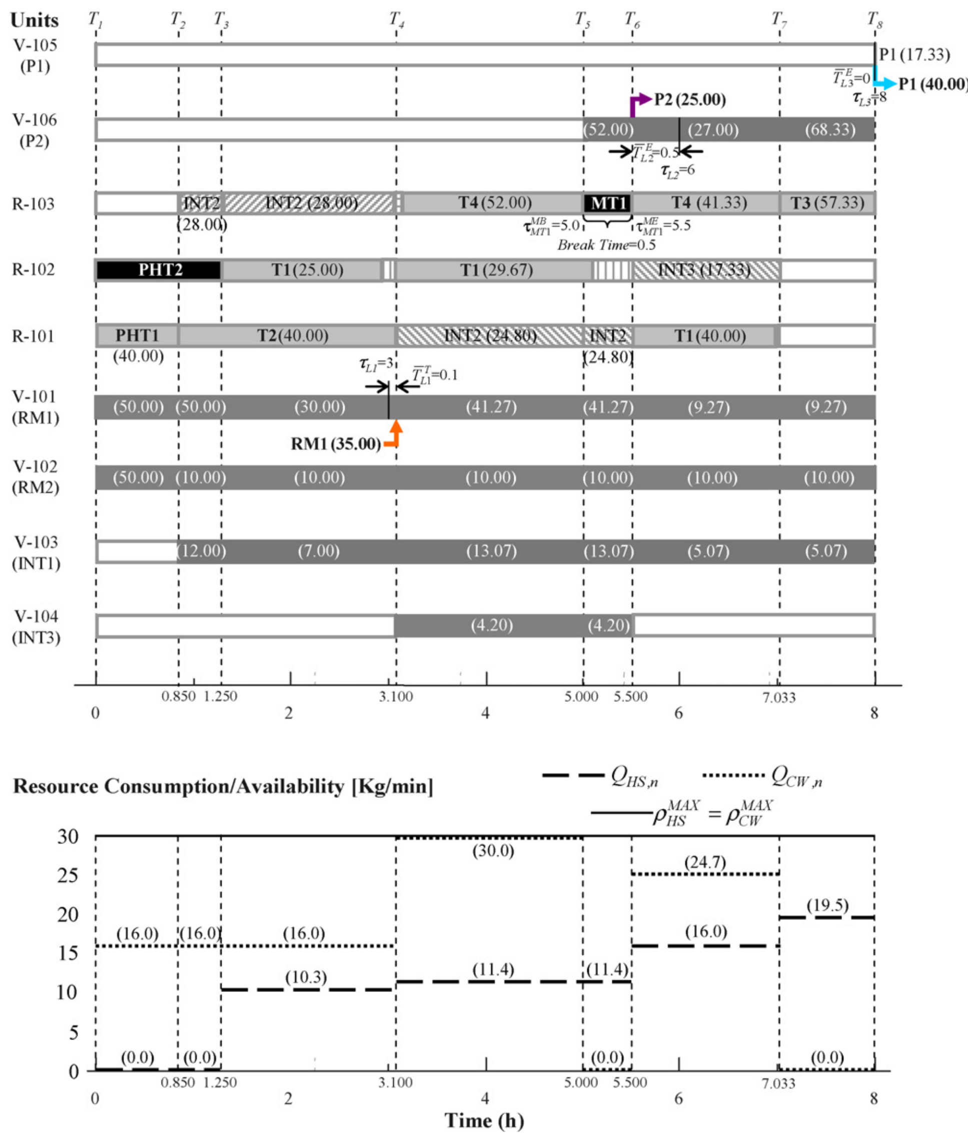


Fig. 10. Best schedule found for Example 1.

maximize our profit over a time horizon of 8 h:

$$\max \sum_{m \in \mathbf{M}^D} \sum_{j \in \mathbf{J}_m^S} \pi_m I_{m,j,N}^S + \sum_{m \in \mathbf{M}^V} \sum_{j \in \mathbf{J}_m^S} \pi_m (I_{m,j,N}^S - I_{m,j}^0) - \sum_{m \in \mathbf{M}^P} \sum_{j \in \mathbf{J}_m^S} \pi_m (I_{m,j}^0 - I_{m,j,N}^S) - \sum_{i \in \mathbf{I}^C} \sum_{j \in (\mathbf{J}^C \cap \mathbf{J}_i)} \sum_{n > 1} \sigma_i Y_{i,j,n} \quad (51)$$

Computational results are presented in Table 7. The best solution having an objective value of $ 3093.0 was reached when eight global time points (seven time intervals) were minimally considered. The Gantt chart of the best schedule is given in Fig. 11, where two changeovers are required; one of them in processing unit R-102 between T2 and T1, and the other one in processing unit R-103 between T4 and T3. Therefore, both cleaning units are used only once.

Table 9
Model and solution statistics for Example 3.

N	CPU time (s)	Nodes	RMILP ($)	MILP ($)	Binary variables	Continuous variables	Constraints	Non-zeros
9	No feasible solution exists							
10	7.59	340	6783.1	1225.7	286	1349	2439	11869
11	16.9	1030	7118.1	1225.7	318	1499	2707	13731
12	**31.4**	**698**	**7386.5**	**2258.3**	**350**	**1649**	**2975**	**15699**
13	110	2616	7603.2	2258.3	382	1799	3243	17773
14	416	9531	7782.3	2258.3	414	1949	3511	19953
15	1238	29505	7934.1	2258.3	446	2099	3779	22239

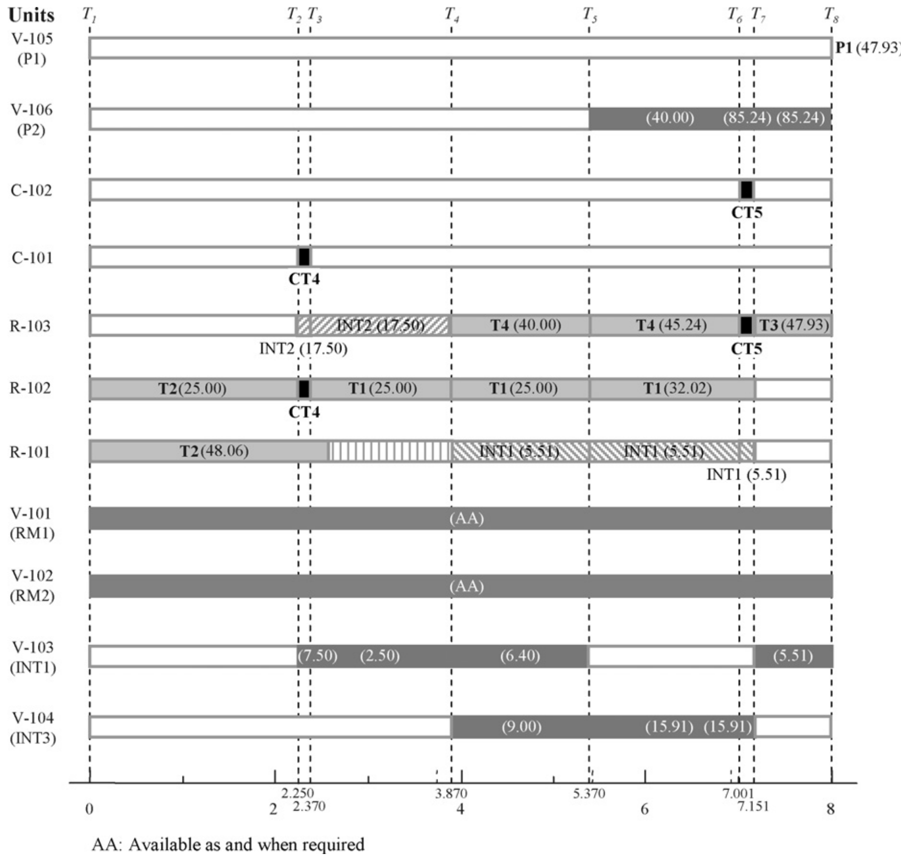
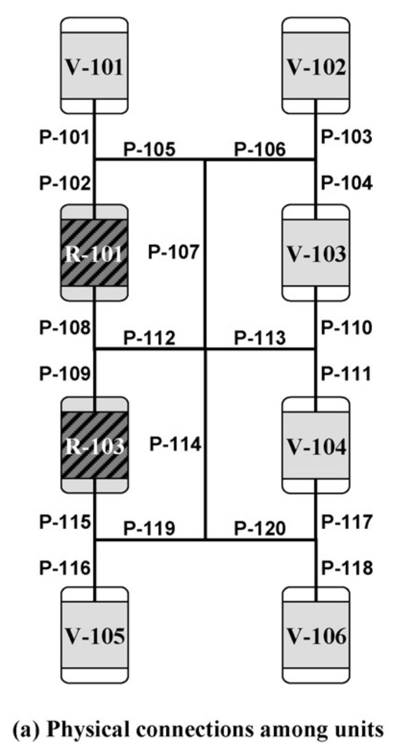
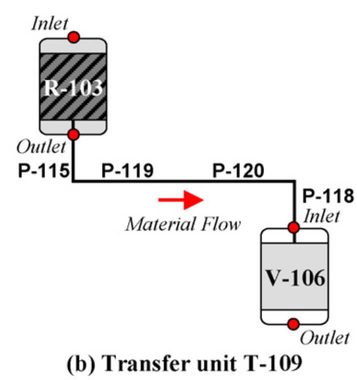


Fig. 11. Best schedule found for Example 2.





Transfer Unit $j \in \mathbf{J}^T$	Connection $j' \in (\mathbf{J}^P \cup \mathbf{J}^S) \to j'' \in (\mathbf{J}^P \cup \mathbf{J}^S)$	Transfer Devices $v \in \mathbf{V}$
T-101	V-101→R-101	P-101,P-102
T-102	V-102→R-101	P-103,P-106,P-105,P-102
T-103	V-103→R-101	P-110,P-113,P-107,P-105,P-102
T-104	R-101→V-103	P-108,P-112,P-107,P-106,P-104
T-105	R-101→V-104	P-108,P-112,P-113,P-111
T-106	R-101→R-103	P-108,P-109
T-107	V-104→R-103	P-117,P-120,P-114,P-112,P-109
T-108	R-103→V-105	P-115,P-116
T-109	R-103→V-106	P-115,P-119,P-120,P-118

(c) Transfer units-devices

Fig. 12. Transfer units/devices for Example 3.

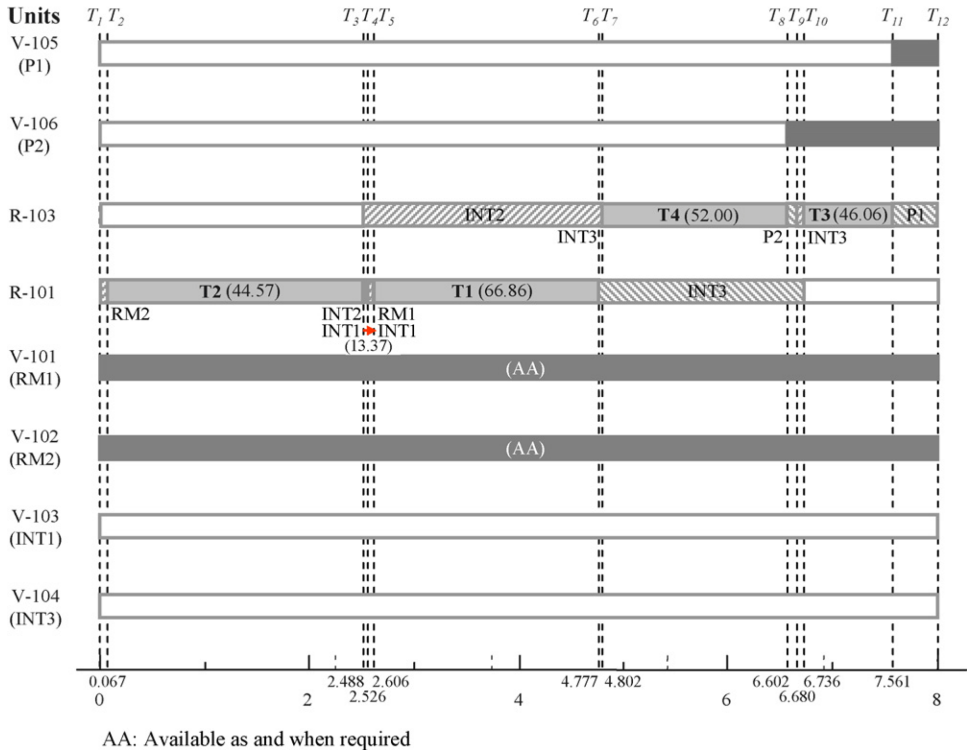


Fig. 13. Best schedule found for Example 3.

5.3. Example 3

The last example considers non-instantaneous material transfers between processing/storage units. Fig. 12a depicts the physical connections among units. In this particular case, processing unit R-102 was eliminated from the plant structure in order to better show structural details and the results obtained. Each physical connection corresponds to a transfer unit, which in turn is composed of a set of transfer devices (pipe sections). Fig. 12b illustrates the transfer unit (named T-109) that connects processing unit R-103 to storage unit V-106. It consists of four pipe sections (P-115, P-119, P-120 and P-118). Data related to transfer units/devices is presented in Fig. 12c and information regarding transfer tasks is shown in Table 8. To simplify the example, only pipe sections were considered. However, pumps and valves can be easily treated using the same notions. This example was solved with the aim of maximizing

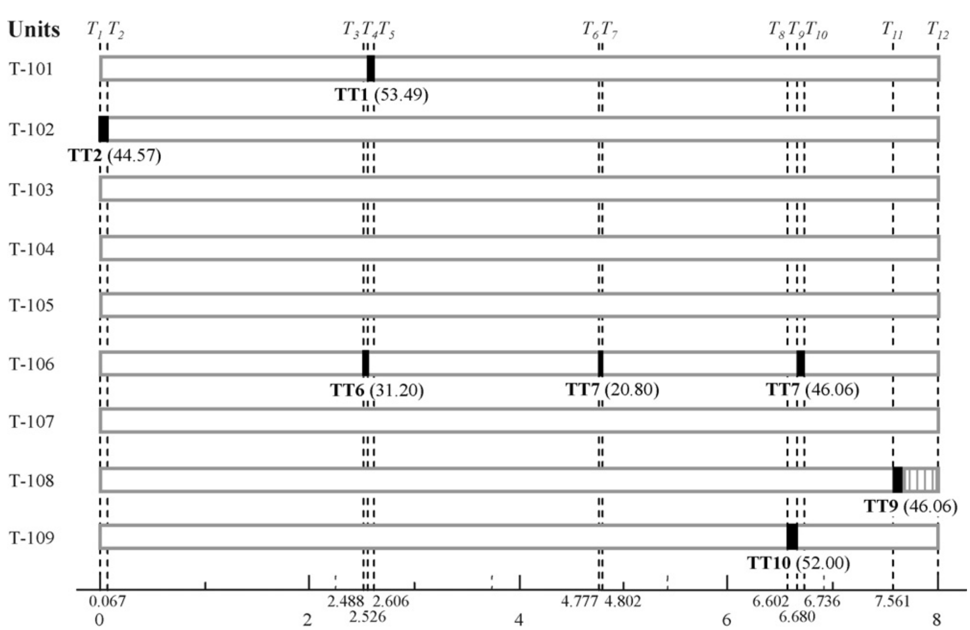


Fig. 14. Material transfers associated with the best solution of Example 3.

the profit over a time horizon of 8 h:

$$\max \sum_{m \in \mathbf{M}^D} \sum_{j \in \mathbf{J}_m^S} \pi_m I_{m,j,N}^S + \sum_{m \in \mathbf{M}^V} \sum_{j \in \mathbf{J}_m^S} \pi_m (I_{m,j,N}^S - I_{m,j}^0) - \sum_{m \in \mathbf{M}^P} \sum_{j \in \mathbf{J}_m^S} \pi_m (I_{m,j}^0 - I_{m,j,N}^S) \quad (52)$$

Table 9 shows the computational results obtained when the resulting MILP formulation was implemented. For this problem, twelve global time points (eleven time intervals) were required to achieve the best schedule shown in Fig. 13. The Gantt chart shows the execution of processing tasks and the materials stored in processing and storage units during each storage interval. In turn, Fig. 14 depicts the best schedule for transfer tasks. As it can be seen, seven transfer tasks take place in five transfer units. Since there are not transfer tasks taking place in parallel, the feasibility of the solution is guaranteed.

6. Conclusions

In this paper, we studied the general short-term batch scheduling problem that may involve (i) preventive maintenance activities, (ii) resource-constrained changeover activities in storage and processing units, (iii) resource-constrained non-instantaneous material transfer activities, (iv) intermediate release- and due-times, and (v) previously scheduled activities being carried out within the current scheduling horizon. To address this broad class of problems, we generalized the representation of Giménez et al. (2009). In particular, we extended the concept of tasks to model maintenance, cleaning, and transfer activities, as well as previously scheduled tasks; we introduced the concept of unit modes to accurately account for cleaning activities; we extended the definition of unary resources to model resource constraints for cleaning and transfer devices; and we introduced new and modified existing constraints to account for intermediate material shipments. The resulting MILP formulation can be used to obtain solutions that cannot be represented by existing methods. Furthermore, it is the first formulation that addresses the general batch-scheduling problem.

Acknowledgments

This work has been supported by CONICET (PIP 5915), Universidad Nacional del Litoral (CAI + D 3-14, 2005), and the National Science Foundation under Grant CTS-0547443.

References

Barbosa-Póvoa, A. P., & Macchietto, S. (1994a). Detailed design of multipurpose batch plants. *Computers and Chemical Engineering*, *18*, 1013–1042.
Barbosa-Póvoa, A. P., & Macchietto, S. (1994b). Redesign of a multipurpose batch pilot plant with Cleaning In Place (CIP) integration. *Computers and Chemical Engineering*, *18*, S277–S281.
Castro, P., Barbosa-Póvoa, A. P., & Matos, H. (2001). An improved RTN continuous-time formulation for the short-term scheduling of multipurpose batch plants. *Industrial and Engineering Chemistry Research*, *40*, 2059–2068.
Castro, P. M., Barbosa-Póvoa, A. P., Matos, H. A., & Novais, A. Q. (2004). Simple continuous-time formulation for short-term scheduling of batch and continuous processes. *Industrial and Engineering Chemistry Research*, *43*, 105–118.
Castro, P. M., Barbosa-Póvoa, A. P., & Novais, A. Q. (2005). Simultaneous design and scheduling of multipurpose plants using resource task network based continuous-time formulations. *Industrial and Engineering Chemistry Research*, *44*, 343–357.
Crooks, C. (1992). *Synthesis of operating procedures for chemical plants*. Ph.D. Thesis. University of London.
Dimitriadis, A. D., Shah, N., & Pantelides, C. C. (1997). RTN-based rolling horizon algorithms for medium term scheduling of multipurpose plants. *Computers and Chemical Engineering*, *21*, S1061–S1066.
Erdirik-Dogan, M., & Grossmann, I. E. (2006). A decomposition method for the simultaneous planning and scheduling of single-stage continuous multiproduct plants. *Industrial and Engineering Chemistry Research*, *45*, 299–315.
Ferrer-Nadal, S., Capón-García, E., Méndez, C., & Puigjaner, L. (2008). Material transfer operations in batch scheduling. A critical modeling issue. *Industrial and Engineering Chemistry Research*, *47*, 7721–7732.
Giménez, D. M., Henning, G. P. & Maravelias, C. T. (2009). A novel network-based continuous-time representation for process scheduling: Part I. Basic concepts and mathematical formulation. Computers and Chemical Engineering, doi:10.1016/j.compchemeng.2009.03.007.
Harjunkoski, I., & Sand, G. (2008). Flexible and configurable MILP-Models for Meltshop scheduling optimization. In B. Braunschweig, & X. Joulia (Eds.), *Computer-aided chemical engineering* (pp. 677–682). Elsevier Science Ltd.
Ierapetritou, M. G., & Floudas, C. A. (1998). Effective continuous-time formulation for short-term scheduling. 1. Multipurpose batch processes. *Industrial and Engineering Chemistry Research*, *37*, 4341–4359.
Ierapetritou, M. G., Hené, T. S., & Floudas, C. A. (1999). Effective continuous-time formulation for short-term scheduling: 3. Multiple intermediate due dates. *Industrial and Engineering Chemistry Research*, *38*, 3446–3461.
Janak, S. L., & Floudas, C. A. (2008). Improving unit-specific event based continuous-time approaches for batch processes: Integrality gap and task splitting. *Computers and Chemical Engineering*, *32*, 913–955.
Janak, S. L., Floudas, C. A., Kallrath, J., & Vormbrock, N. (2006a). Production scheduling of a large-scale industrial batch plant. I. Short-term and medium-term scheduling. *Industrial and Engineering Chemistry Research*, *25*, 8234–8252.
Janak, S. L., Floudas, C. A., Kallrath, J., & Vormbrock, N. (2006b). Production scheduling of a large-scale industrial batch plant. II. Reactive scheduling. *Industrial and Engineering Chemistry Research*, *25*, 8253–8269.
Janak, S. L., Lin, X., & Floudas, C. A. (2004). Enhanced continuous-time unit-specific event-based formulation for short-term scheduling of multipurpose batch processes: Resource constraints and mixed storage policies. *Industrial and Engineering Chemistry Research*, *43*, 2516–2533.
Kondili, E., Pantelides, C. C., & Sargent, W. H. (1993). A general algorithm for short-term scheduling of batch operations-I. MILP formulation. *Computers and Chemical Engineering*, *17*, 211–227.
Maravelias, C. T., & Grossmann, I. E. (2003a). A new general continuous-time state task network formulation for the short-term scheduling of multipurpose batch plants. *Industrial and Engineering Chemistry Research*, *42*, 3056–3074.
Maravelias, C. T., & Grossmann, I. E. (2003b). A general continuous state task network formulation for short term scheduling of multipurpose batch plants with due dates. In B. Chen, & A. Westerberg (Eds.), *Computer-aided chemical engineering* (pp. 274–279). Elsevier Science Ltd.
Maravelias, C. T. (2005). Mixed-time representation for state-task network models. *Industrial and Engineering Chemistry Research*, *44*, 9129–9145.
Mockus, L., & Reklaitis, G. V. (1999). Continuous time representation approach to batch and continuous process scheduling. 1. MINLP formulation. *Industrial and Engineering Chemistry Research*, *38*, 197–203.
Pantelides, C. C. (1994). Unified frameworks for optimal process planning and scheduling. In *In foundations of computer-aided process operations*. New York: Cache Publications., pp. 253–274
Prasad, P., Maravelias, C. T., & Kelly, J. (2006). Optimization of aluminum smelter casthouse operations. *Industrial and Engineering Chemistry Research*, *45*, 7603–7617.
Sanmartí, E., Espuña, A., & Puigjaner, L. (1997). Batch production and preventive maintenance scheduling under equipment failure uncertainty. *Computers and Chemical Engineering*, *21*, 1157–1168.
Schilling, G., & Pantelides, C. C. (1996). A simple continuous-time process scheduling formulation and a novel solution algorithm. *Computers and Chemical Engineering*, *20*, S1221–S1226.
Shah, N., Pantelides, C. C., & Sargent, W. H. (1993). A general algorithm for short-term scheduling of batch operations-II. Computational issues. *Computers and Chemical Engineering*, *17*, 229–244.
Sundaramoorthy, A., & Karimi, I. A. (2005). A simpler better slot-based continuous-time formulation for short-term scheduling in multipurpose batch plants. *Chemical Engineering Science*, *60*, 2679–2702.
Zhang, X., & Sargent, W. H. (1996). The optimal operation of mixed production facilities—A general formulation and some approaches for the solution. *Computers and Chemical Engineering*, *20*, 897–904.