Annals of Operations Research 81(1998)433–466

Assignment and sequencing models for the scheduling of process systems

Jose M. Pinto* and Ignacio E. Grossmann
Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA
E-mail: grossmann@cmu.edu

This paper presents an overview of assignment and sequencing models that are used in the scheduling of process operations with mathematical programming techniques. Although scheduling models are problem specific, there are common features which translate into similar types of constraints. Two major categories of scheduling models are identified: single-unit assignment models in which the assignment of tasks to units is known a priori, and multiple-unit assignment models in which several machines compete for the processing of products. The most critical modeling issues are the time domain representation and network structure of the processing plant. Furthermore, a summary of the major features of the scheduling model is presented along with computational experience, as well as a discussion on their strengths and limitations.

1. Introduction

Scheduling is one of the core areas of process operations. It is also an active and challenging area of research that is of practical importance. In general, scheduling deals with the allocation of available resources over time to perform a collection of tasks. Scheduling problems arise in many fields of engineering, operations research and computer science. In the context of process systems, scheduling refers to the strategies of allocating equipment and utility or manpower resources over time to execute processing tasks required to manufacture one or several products. There has been a pronounced increase in the development of optimization scheduling models in the chemical engineering literature over the past decade. Despite the substantial progress that has been made, combinatorics and time representation remain major challenges in the area. This has created a rather diverse collection of scheduling models in which there often appears to be a lack of commonalities. In an attempt to provide some unification of the models, we present a survey of the main assignment and sequencing models for process scheduling problems within the mathematical program-

*Present address: Department of Chemical Engineering, Escola Politecnica USP, São Paulo SP 05508-900, Brazil. E-mail: jompinto@usp.br

ming framework. We primarily focus on the modeling aspects regarding the plant structure and representation of the time domain. As will be shown, these models exhibit some differences to the ones commonly reported in the operations research literature.

Rippin [22] addresses the general status of batch processing systems engineering with emphasis on design, planning and scheduling. Reklaitis [19, 20] presents a comprehensive review of scheduling and planning of batch process operations. His main focus was to describe the basic components of the scheduling problem and review the existing solution methods. Pekny and Zentner [15] summarize the basic scheduling technology with association to the advances in computer technology. Grossmann et al. [9] provide an overview of mixed integer optimization techniques for the design and scheduling of batch processes, with emphasis on general purpose methods for mixed integer linear (MILP) and mixed integer nonlinear (MINLP) problems. Many scheduling and planning problems can be posed as MILP problems since the corresponding mathematical optimization models involve both discrete and continuous variables that must satisfy a set of linear equality and inequality constraints. In fact, scheduling problems can be viewed as optimization problems subject to constraints, namely problems in allocation and sequencing [1].

Although there is no absolute general model for the scheduling of process systems, three major components are always present [20]. These are: assignment of tasks to equipment, sequencing of activities and timing of utilization of equipment and resources by these processing tasks. Another important aspect is the presence of sequence dependent changeovers. Some formulations include sequence-dependent decisions as additional constraints, while others embed it in the underlying model representation. Also, in some instances, material balances are required to deal explicitly with both resources and inventories.

It is the purpose of this paper to provide a unified overview of the assignment and sequencing models for chemical process scheduling. We first present a road-map for classifying scheduling problems for process systems. The bulk of the paper then concentrates on the allocation and timing constraints which constitute the basis of the mathematical optimization models. Based on the proposed classification of problems, we first present the single-unit assignment models, developed mainly for multistage plants with one unit per stage. Next, we concentrate on describing multiple-unit assignment models. These can be broadly divided into two major categories: models based on time slots, aimed primarily to multistage serial plants with parallel units, and models based on time events focused on plants with arbitrary network structure. Finally, we present a summary of computational experience.

2. Classification of scheduling problems

Figure 1 presents a road-map for classifying scheduling problems and which should help in obtaining a better understanding of the models that are covered in the following sections. Equipment similarity and unit connectivity define the topology of

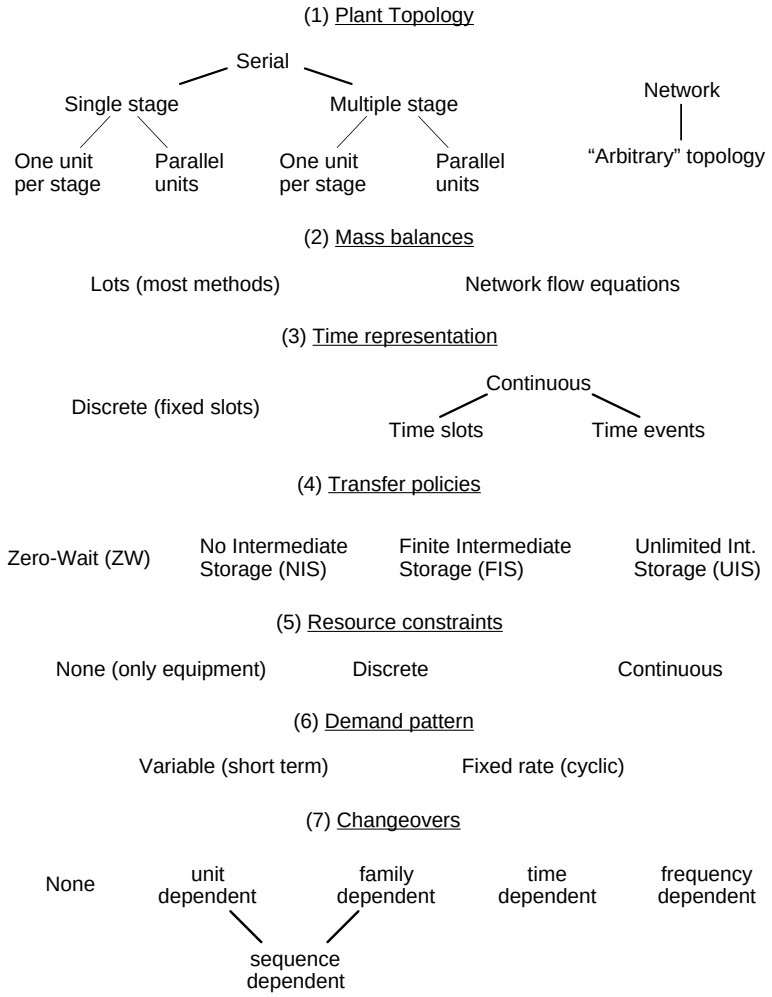

Figure 1. Road-map for scheduling problems.

the plant. In serial plants, products follow the same production path, therefore it is possible to recognize a specific direction in the plant floor. Networks of arbitrary topology tend to occur when products have low recipe similarity and/or when equipment is interconnected. Most methods do not handle mass balances explicitly; instead, production is represented by batches (or lots). Products follow a series of tasks, which are collections of elementary chemical and physical processing operations. Note the close relationship between the plant topology and the sequence of tasks for products: if all products follow the same sequence of tasks it is usually possible to define processing stages in the plant, defined as processing equipment that can perform the same operations. Moreover, lot sizes can be variables, such as in the case of the lot-sizing problem, or fixed parameters.

The fundamental issue in assignment and sequencing decisions concerns the time domain representation. A common feature among scheduling models is the definition of time slots, i.e., time intervals for unit allocation. In a continuous time domain representation, time slots have variable length. Moreover, they can be either associated with units or defined globally, in which case they are often denoted as time events. If a discrete representation is adopted, slots have equal and fixed duration. The length of the time interval is taken to be the greatest common factor of the processing times involved in the problem. Figure 2 illustrates the definition of time slots. The total

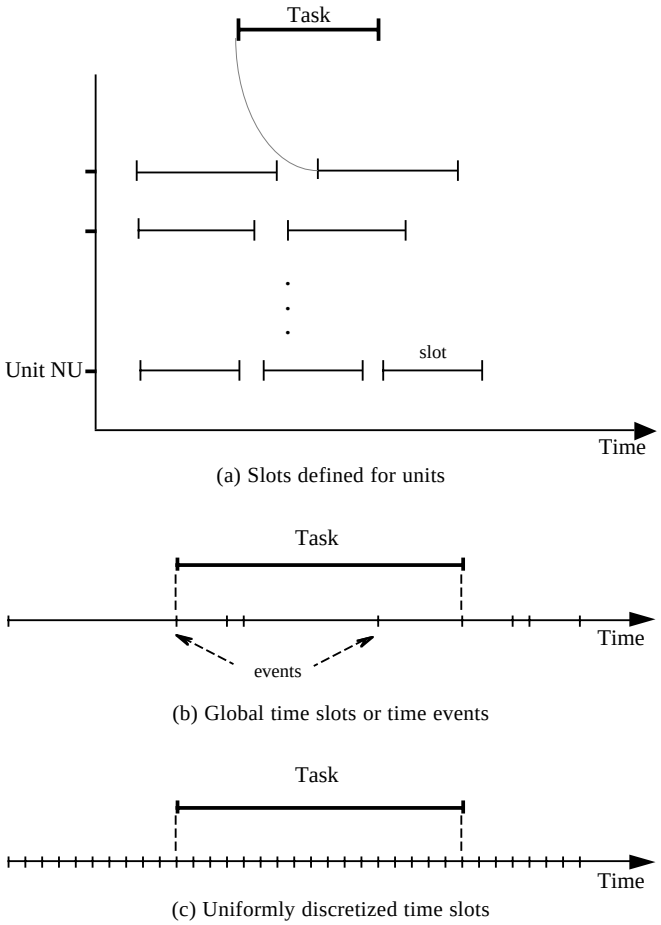

Figure 2. Time domain representation.

number of time slots also depends on the time representation adopted. When time is discretized, there is in general the need to use a sufficiently large number of slots in order to have a suitable approximation of the original problem. Note, however, that the number of slots is fixed. In continuous time formulations, it is usually possible to

postulate a much smaller number of time slots. However, it is important to select the time slots in such a way that the number of events to be executed can be accommodated within the given time horizon. In simpler cases (e.g., no parallel units), the number of slots is equal to the number of batches to be produced.

Another major issue in plant scheduling deals with the presence of intermediate storage. There are four different transfer policies: Zero-Wait (ZW), No-Intermediate-Storage (NIS), Finite-Intermediate-Storage (FIS) and Unlimited-Intermediate-Storage (UIS) [12]. It is important to note that FIS corresponds to the most general case. Nevertheless, the main advantage of the remaining three cases is that there is no need to model inventory levels.

In the scheduling of a process plant, processing tasks require utilities and manpower. Utilities may include, for example, steam, electricity, cooling water, etc. In some scheduling applications, apart from equipment, finite resources that are limited are required for these process tasks. Resource constrained scheduling problems are inherently difficult, due to the fact that, besides the efficient allocation of units to meet product demands, it is also necessary to consider the feasible grouping of simultaneously executed tasks so as to utilize resources within their availability limits. In general, scheduling problems are concerned with renewable resources (manpower, energy), while planning problems handle non-renewable resources (raw material, capital). Moreover, renewable resources may be discrete, i.e., consumed at a constant level throughout the processing of the product, or continuous, in which the demand for the resource varies with the processing time.

The short-term scheduling is relevant to plants that must satisfy individual customer orders with little similarity between demand patterns in different planning periods. In this case, product requirements are given as a set of orders, where each order has associated with it a certain product, the amount and a due date. In contrast, cyclic scheduling is relevant for plants operating with a stable market in which the product demands are given as constant rates. This allows a more simplified plant operation in which the same production sequence is executed repeatedly with a fixed frequency. It should also be noted that cyclic scheduling is often considered in longer range planning studies. In both short-term and cyclic scheduling, it is important to define the inventory policy, which in turn might also require allocation of the production to inventory build-up.

When switching between products, or even after one or more batches of the same product, units may require cleaning and setup for safety and/or product quality. Changeover requirements depend on the nature of the units and the products in the plant. Sequence-dependent changeovers represent the most general situation, in which every pair of consecutive operations may give rise to different time and/or cost requirements. A simpler situation occurs when changeovers are family dependent, i.e., products are divided into disjoint families and changeovers only occur between products of different families. Moreover, the need for unit setup may be expressed in terms of the frequency of utilization. For instance, a changeover may be needed after

every batch or after a certain number of batches, regardless of the nature of the products. Time-dependent cleaning can also be considered. In this case, there is a maximum time interval during which a unit may be utilized.

2.1. Single-unit assignment models

This class of models involve situations in which the assignment of orders or tasks to units is known. In other words, each product processing task is identified with one single unit. In this case, although it might in principle be convenient to use the concept of time slots, their application is not strictly necessary. The simplest structure corresponds to the sequential multiproduct plant which consists of multiple stages with one unit per stage and where all products are processed sequentially at each stage. This type of plant is normally referred to as the flowshop plant in the operations research literature [7]. These plants process products with a high degree of similarity that requires the same sequence of tasks. Interestingly, two other important models with fixed single-unit assignments that are extensively studied in the operations research literature are the single-machine scheduling problem, which corresponds to the single-stage case, and the classical jobshop scheduling problem, in which the sequence of processing tasks is not the same for different products. The fundamental problem for scheduling in both cases is to determine the sequence of operations for given production requirements. Nevertheless, in all cases the allocation of jobs to machines is known a priori.

Birewar and Grossmann [3, 4], Voudouris and Grossmann [29] and Pinto and Grossmann [16] considered the multistage case and assumed that each stage involves one processing unit. Additionally, they imposed the condition that all the products follow the same processing sequence. The scheduling problem corresponds to the permutation flowshop of operations research. By introducing the binary variables X_{ik} to denote the processing of product i in time slot k, the mathematical representation of the assignment and sequencing constraints is

$$\sum_i X_{ik} = 1 \qquad \forall k, \tag{1a}$$

$$\sum_k X_{ik} = 1 \qquad \forall i. \tag{1b}$$

Constraints (1a) enforce that exactly one product i is assigned to every time slot k and constraints (1b) enforce that every product i is assigned to exactly one time slot k. Note that in this case, each unit corresponds to a different stage. Moreover, as the processing sequence in all units is the same, it can be determined globally. In this case, there is a one-to-one correspondence between products and time slots. Figure 3(a) illustrates a permutation flowshop schedule of a multistage plant with one unit per stage.

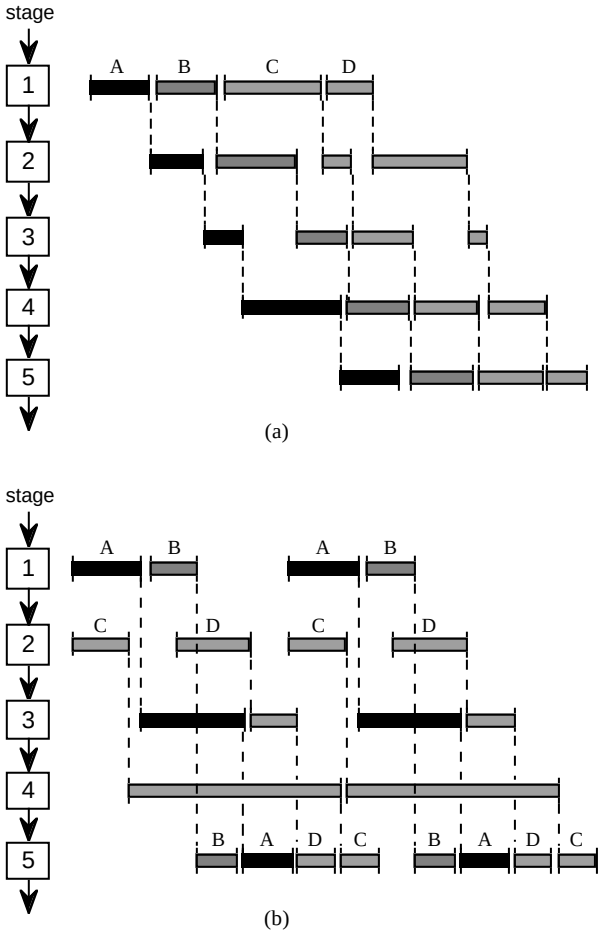

Figure 3. (a) Permutation flowshop schedule. (b) Sequential jobshop schedule.

For sequence dependent changeovers, let $Z_{ii'k}$ be a 0–1 transition variable that denotes the processing of product i at slot k immediately before processing of product i'. The transition variable $Z_{ii'k}$ has to be linked to the assignment variables in such a way that $Z_{ii'k}$ is activated when both X_{ik} and $X_{i'k+1}$ are one. One way of enforcing this condition is as follows:

$$Z_{ii'k} \geq X_{ik} + X_{i'k+1} - 1 \qquad \forall i, i', k \in K - \{\bar{k}\}, \tag{2a}$$

where $\bar{k}$ is the last element of the set of time slots K. Another way of enforcing the same condition in (2a) is to use the following set of constraints:

$$\sum_{i} Z_{ii'k} = X_{i'k+1} \qquad \forall i', k \in K - \{\bar{k}\}, \tag{2b}$$

$$\sum_{i'} Z_{ii'k} = X_{ik} \qquad \forall i, k \in K - \{\bar{k}\}. \tag{2c}$$

According to (2b), one transition from product i to product i' occurs if and only if i' is produced in the subsequent time slot. Also, from (2c), exactly one transition from i to i' occurs if and only if i is produced at time slot k. Constraints (2b)–(2c) were originally introduced by Sahinidis and Grossmann [23] for the single-stage plant with parallel lines; appendix I illustrates the procedure for deriving constraints (2b)–(2c) from nonlinear integer constraints. It is important to note that constraints (2) apply to non-cyclic schedules; for the cyclic case, the indices $k + 1$ are replaced by the "++1" cyclic operator (see Pinto and Grossmann [16]).

Furthermore, the models include timing constraints, which involve the determination of the specific start and completion times for each of the tasks undergoing scheduling. Extra variables are required to model the times associated with each unit. These are Ts_{jk}, the time at which processing in slot k of unit j is started and Te_{jk}, the time at which processing in slot k of unit j is completed. When time slots are defined for each unit, no overlap is allowed. Given the processing times T_{ij} of product i in unit j and assuming no sequence-dependent changeovers, the time variables are related in terms of the assignment variables X_{ik} by the equation

$$Te_{jk} = Ts_{jk} + \sum_{i \in I} X_{ik} T_{ij} \qquad \forall j, k. \tag{3a}$$

Additionally, the start time of slot $k + 1$ at every unit j requires that the processing of slot k be finished. That is,

$$Te_{jk} \leq Ts_{jk+1} \qquad \forall j, k \in K - \{\bar{k}\}. \tag{3b}$$

Note that sequence-independent changeovers are trivially treated by adding the setups to T_{ij} in (3a). When we consider sequence-dependent changeovers, the binary variable $Z_{ii'k}$ replaces X_{ik}. Introducing the transition time $\tau_{ii'j}$ from product i to i' in the timing constraint yields

$$Te_{jk} = Ts_{jk} + \sum_{i \in I} \sum_{i' \in I} Z_{ii'k}(T_{ij} + \tau_{ii'j}) \qquad \forall j, k. \tag{3c}$$

As for time relations in two successive stages, these depend on the sequencing policy, as described in the introduction. For the Zero-Wait (ZW) policy, where no intermediate storage is available and no idle times are allowed for processing between stages, the start time of unit $j + 1$ has to match exactly the completion time in unit j:

$$Te_{jk} = Ts_{j+1k} \qquad \forall j \in J - \{\bar{j}\}, k. \tag{3d}$$

For the Unlimited-Intermediate-Storage (UIS) policy, where it is possible to store the batch produced in unit j, the start time in unit $j + 1$ can be performed any time after the completion of unit j:

$$Te_{jk} \leq Ts_{j+1k} \qquad \forall j \in J - \{\bar{j}\}, k. \tag{3e}$$

An alternative representation for multiproduct or flowshop problems relies on the use of the binary variable $X'_{ii'}$, to denote the processing of batch of product i immediately before batch of product i'. In this case, the assignment constraints for scheduling correspond to the structure of the Traveling Salesman Problem (TSP) as shown by Gupta [10]:

$$\sum_{i} X'_{ii'} = 1 \qquad \forall i', \tag{4a}$$

$$\sum_{i'} X'_{ii'} = 1 \qquad \forall i, \tag{4b}$$

$$\sum_{i \in S} \sum_{i' \in I \setminus S} X'_{ii'} \leq |S| - 1 \qquad \forall S \subset I, S \neq \varnothing. \tag{4c}$$

Equations (4a) and (4b) guarantee that all batches have exactly one predecessor and one successor. Note that the assignment constraints corresponding to (4a) and (4b) possess the same structure of (1a) and (1b) and do not involve the definition of slots. However, this representation requires the addition of the subtour elimination constraints (4c), which ensures that a single closed cycle is obtained. The advantage of the above constraints is that changeovers can easily be handled since they can be directly expressed in terms of the variables $X'_{ii'}$. However, the drawback is that the number of subtour elimination constraints can be extremely large, and therefore a relaxation strategy must normally be used. Fortunately, when changeovers are sequence-dependent this gives rise to an asymmetric TSP, which yields very tight bounds when the subtour elimination constraints are excluded (see Pekny and Miller [14]). Birewar and Grossmann [4] considered an aggregated form of the constraints in (4a)–(4b) with which the problem can be solved in the space of products rather than in the space of batches. Provided the number of products is not very large, it is possible to handle problems with an arbitrary number of batches with this formulation.

If there is still one processing unit per stage, but we relax the assumption that each product follows the same sequence in each stage (e.g., the sequential jobshop problem), it becomes convenient to define slots for each unit. This is particularly important for sequential multipurpose plants, i.e., plants in which it is possible to recognize a specific direction followed by the production paths of all products but with units being skipped by some products. Figure 3b shows a schedule of a sequential multipurpose plant with five processing stages; note that the batch processed first in stage two skips stages one and three and is the second to be processed in stage five. Defining W_{ijk} as the assignment of product i in slot k of unit j, the assignment constraints become

$$\sum_{i \in I_j} W_{ijk} = 1 \qquad \forall j, k \in K_j, \tag{5a}$$

$$\sum_{k \in K_j} W_{ijk} = 1 \qquad \forall i, j \in J_i, \tag{5b}$$

where I_j is the set of products to be processed in unit j and J_i is the set of units in which product i has to go through. Note also that in this case, the number of slots can be defined for each unit using the set K_j, according to the number of jobs to be assigned to it. Timing constraints similar to the ones in (3) must be defined for the above problem.

The scheduling and sequencing of a multipurpose plant has also been addressed by Rich and Prokopakis [21] as a jobshop problem by considering that each product must be processed through a sequence of tasks that are assigned to a processing unit. This gives rise to a disjunctive programming problem. The fundamental constraints must enforce that two tasks i and i' in potential conflict, i.e., two operations that require the same processing unit j and with no specified precedence relationship, cannot be processed simultaneously. These disjunctive constraints can be represented in mixed integer form with the binary variables $X''_{ii'}$ that are activated if task i precedes i' and the continuous variables Tsi_i that denote the start time of operation i, as follows:

$$Tsi_{i'} - Tsi_i + U(1 - X''_{ii'}) \geq T_i \qquad \forall i \in I_j,\ i' \in I_j,\ j, \tag{6a}$$

$$Tsi_i - Tsi_{i'} + UX''_{ii'} \geq T_{i'} \qquad \forall i \in I_j,\ i' \in I_j,\ j, \tag{6b}$$

where U is some valid upper bound. Equations (6) are "either-or" (disjunctive) constraints. The interpretation is that *either* i precedes i' (when $X''_{ii'}$ is one) *or* i' precedes i (when $X''_{ii'}$ is zero). Note that, in this model, the start and end times for each product can be interpreted as event times.

2.2. Multiple-unit assignment – models based on time slots

When several units in parallel compete for processing orders, i.e., a task can be performed in more than one unit, there is the need to decide the assignment of products to machines. One approach is to define a certain number of time slots for each of the units. This is particularly suitable for sequential plants.

A simpler situation occurs when there is only one production stage. In addition to being of interest in themselves (e.g., parallel machines), single-stage problems often apply to more complex problems, either because a single ("bottleneck") stage dominates or because multiple-stage problems can be decomposed (exactly or approximately) into independent or loosely coupled single-stage problems. In this case, there is a one-to-one correspondence between task and order. If a continuous time representation is used, Pinto and Grossmann [17] suggest that the assignment and sequencing constraints be written as follows. Let W_{ijk} represent a binary variable for assigning order i in time slot k of unit j. Since for all orders exactly one time slot k of one unit j is allotted

$$\sum_{j \in J_i} \sum_{k \in K_j} W_{ijk} = 1 \qquad \forall i. \tag{7a}$$

Note that constraint (7a) accounts only for the units $j \in J_i$ in which order i is allowed to be processed. Moreover, time slots are unit dependent. Also, since at most one order can be assigned to each time slot of a unit,

$$\sum_{i \in I_j} W_{ijk} \leq 1 \qquad \forall j, k \in K_j. \tag{7b}$$

Some time slots may be empty, due to an overestimation on their total number. Sahinidis and Grossmann [23] considered that for the case of single-stage continuous multiproduct plants, products can be assigned to consecutive time slots and to more than one unit. Therefore, constraint (7a) is dropped and (7b) becomes an equality constraint, i.e., all the slots are occupied.

If changeovers are relevant to the problem, the equations (2b)–(2c) by Sahinidis and Grossmann [23] can be easily extended by introducing variables $Z_{ii'jk}$ that explicitly account for sequence-dependent transitions. The constraints that relate the assignment variables W_{ijk} and the transition variables $Z_{ii'jk}$ can be written as

$$\sum_{i \in I_j} Z_{ii'jk} = W_{i'jk+1} \qquad \forall i' \in I_j,\ j, k \in K_j - \{\overline{k_j}\}, \tag{8a}$$

$$\sum_{i' \in I_j} Z_{ii'jk} = W_{ijk} \qquad \forall i \in I_j,\ j, k \in K_j - \{\overline{k_j}\}. \tag{8b}$$

As a result of constraint (8a), exactly one transition from a product i to the given product i' occurs in the end of any time slot if and only if i' is produced during the subsequent time slot. According to (8b), a transition from i to exactly one i' occurs at the end of any time slot if and only if i is being produced during that time slot. Appendix I illustrates that in fact constraints (8a) and (8b) can be obtained from the linearization of nonlinear integer constraints. As previously mentioned for equations (2), constraints (8) apply to the non-cyclic case.

Time variables model the start and completion times of the slots in the units, namely Ts_{jk} and Te_{jk}. As in equations (3) of the single-unit assignment models, when time slots are defined for each unit, no overlap is allowed. In other words, the start time of slot $k + 1$ at every unit j requires that the processing of slot k be finished:

$$Te_{jk} \leq Ts_{jk+1} \qquad \forall j, k \in K_j - \{\overline{k_j}\}. \tag{9a}$$

Also, the start and completion times in the slots are related as follows:

$$Te_{jk} = Ts_{jk} + \sum_{i \in I_j} W_{ijk} T_{ij} \quad \forall j, k \in K_j. \tag{9b}$$

When we consider sequence-dependent changeovers following a similar reasoning as in (3c), the binary variable $Z_{ii'jk}$ replaces W_{ijk} by introducing the transition time from product i to i' in constraint (9b), yielding the timing constraint (9c),

$$Te_{jk} = Ts_{jk} + \sum_{i \in I_j} \sum_{i' \in I_j} Z_{ii'jk}(T_{ij} + \tau_{ii'j}) \qquad \forall j, k \in K_j. \tag{9c}$$

By defining binary variables $X'_{ii'j}$ to denote if product i precedes product i' in unit j, it is possible to derive a model with a similar structure to the TSP model for the scheduling of single-stage plants with parallel units [5]. The assignment and sequencing constraints could be seen as a generalization of (4), combined with the fact that products are assigned to exactly one processing unit j.

Gooding et al. [8] introduce an alternative formulation based on a uniform time discretization that considers sequence-dependent assignments explicitly with the binary variables $Z_{ii'jk}$ that were previously introduced in equations (8). The assignment constraints (10) force each job to be assigned to a unique successor, a unique predecessor and a unique time slot,

$$\sum_{i'} \sum_{j \in (J_i \cap J_{i'})} \sum_{k \in K_j} Z_{ii'jk} = 1 \qquad \forall i, \tag{10a}$$

$$\sum_{i} \sum_{j \in (J_i \cap J_{i'})} \sum_{k \in K_j} Z_{ii'jk} = 1 \qquad \forall i', \tag{10b}$$

$$\sum_{i \in I_j} \sum_{i' \in I_j} Z_{ii'jk} \leq 1 \qquad \forall j, k \in K_j. \tag{10c}$$

In this approach, although time is uniformly discretized, each unit may have its own interval length since products are not allowed in all units. Interval sizes are selected based on the greatest common denominator of the processing times of jobs that may be processed in unit j, given by the set I_j. Also, "dummy" jobs are assigned to the initial and to the final slots of each unit. The following constraint states that if job i is processed on line j in slot k, no job can begin production during the next $T_{ij} + \tau_{ii'j} - 1$ slots associated with the production of i and transition from product i to i':

$$\sum_{r \in I_j} \sum_{s \in I_j} \sum_{k'=1}^{T_{ij}+\tau_{ii'j}-1} Z_{r,s,j,k+k'} \leq (T_{ij} + \tau_{ii'j} - 1)(1 - Z_{ii'jk}) \quad \forall i, i', j \in (J_i \cap J_{i'}), k \in K_j. \tag{11}$$

It is important to note that for discrete time models, a key assumption is that any event occurs at interval boundaries. Moreover, as previously mentioned, time slots have fixed duration. Therefore, timing of events such as non-overlapping of tasks is done explicitly with the assignment constraints. Nevertheless, it is possible to enforce certain timing restrictions using the assignment variables. For instance, Gooding et al. [8] consider the following constraints:

$$\sum_{s \in I_j} Z_{i',s,j,k+T_{ij}+\tau_{ii'j}} \geq Z_{ii'jk} \qquad \forall i, i', j \in (J_i \cap J_{i'}), k \in K_j. \tag{12}$$

Constraint (12) enforces that the units are always busy, except for changeovers. It states that if job i begins production on line j in slot k with successor i', then job i' must begin in line j in time slot $k + T_{ij} + \tau_{ii'j}$.

It is interesting to note that constraints (10c) and (11) are analogous to constraints (23a) and (23b) by Kondili et al. [11], to be defined later in the paper. It is possible to define an equivalent representation, which is based on a backward aggregation of time and similar to constraints (24) developed by Shah et al. [27]. The equivalent representation is given by

$$\sum_{i \in I_j} \sum_{i' \in I_j} \sum_{k' = k - T_{ij} - \tau_{ii'j} + 1}^{k} Z_{ii'j,\, k+k'} \leq 1 \qquad \forall j, k \in K_j. \tag{13}$$

A detailed proof of equivalence of constraints (10c)–(11) and (13) is given in appendix II. Following a similar treatment as in the comparison between constraints (23) and (24) of the STN model presented later in the paper, it is also shown in appendix II that (13) is not only tighter than (10c)–(11), but is also a more compact representation.

The multistage flowshop problem with units in parallel is studied in Pinto and Grossmann [17, 18]. In this case, product i goes through several stages and each of the stages l is assigned to time slots k of units j, as several parallel units may correspond to a single processing stage, given by the set J_l:

$$\sum_{j \in (J_i \cap J_l)} \sum_{k \in K_j} W_{ijkl} = 1 \qquad \forall i, l \in L_i, \tag{14a}$$

$$\sum_{i \in I_j} \sum_{l \in (L_i \cap L_j)} W_{ijkl} \leq 1 \qquad \forall j, k \in K_j. \tag{14b}$$

Equation (14a) represents the fact that for all orders and in all stages, exactly one time slot of one piece of equipment is allotted. Also, not necessarily all processing stages are involved in the production of i; the stages actually included in the processing of i are defined in L_i. Moreover, only a subset of the units in which stage l of order i (set $J_i \cap J_l$) can be processed is considered in the summation. The extension to sequence-dependent changeovers is analogous to the single stage case (see equations (8)).

Pinto and Grossmann [17] suggest a time-matching approach. When stage l of order i is assigned to slot k of unit j matching takes place by enforcing equality of the start and end times of these coordinates through mixed integer constraints (see equation (15)). Binary variables W_{ijkl} are used to model the potential assignment of stage l of order i to time slot k of unit j. When order i is assigned to unit j ($W_{ijkl} = 1$), the start times in both coordinates are enforced to be the same. Otherwise, the constraints are relaxed:

$$-U(1 - W_{ijkl}) \leq Tsi_{il} - Ts_{jk} \qquad \forall i, j \in (J_i \cap J_l), k \in K_j, l \in L_i, \tag{15a}$$

$$U(1 - W_{ijkl}) \geq Tsi_{il} - Ts_{jk} \quad \forall i, j \in (J_i \cap J_l), k \in K_j, l \in L_i. \quad (15b)$$

Another way of enforcing the time-matching is as follows. Consider equation (16) that imposes the same condition of (15a) and (15b) which, however, is in nonlinear form:

$$(Tsi_{il} - Ts_{jk})W_{ijkl} = 0 \quad \forall i, j \in (J_i \cap J_l), k \in K_j, l \in L_i. \quad (16)$$

Constraint (16) can be linearized, as shown in appendix I of Pinto and Grossmann [17]. Extra continuous variables are introduced to the problem, namely θ_{ijkl} and γ_{jk}, which represent disaggregated variables for the start times. The linearized form relies on the use of the multiple-choice assignment constraints (14a) and (14b). The reformulated time-matching constraints are as follows:

$$Tsi_{il} = \sum_{j \in (J_i \cap J_l)} \sum_{k \in K_j} \theta_{ijkl} \quad \forall i, l \in L_i, \quad (17a)$$

$$Ts_{jk} = \sum_{i \in I_j} \sum_{l \in (L_i \cap L_j)} \theta_{ijkl} + \gamma_{jk} \quad \forall j, k \in K_j, \quad (17b)$$

$$0 \leq \theta_{ijkl} \leq U^{il} W_{ijkl} \quad \forall i, j \in (J_i \cap J_l), k \in K_j, l \in L_i, \quad (17c)$$

$$\gamma_{jk} - U y_{jk} \leq 0 \quad \forall j, k \in K_j, \quad (17d)$$

where y_{jk} is a slack 0–1 variable for time slot k of unit j (y_{jk} is one if the slot is empty; zero otherwise). An important feature of representing the time-matching equations (17a) –(17d) is that fewer constraints are necessary, although there is an increase in the number of continuous variables. Also, a tighter upper bound than the ones in (15) can be used in this representation. Note that the variables θ_{ijkl} are bounded by Tsi_{il} as seen in equation (17a). Therefore, upper bounds on the start times of the orders U^{il} can be used in (17c), which produces a tighter LP relaxation than the constraints in (15).

Another major issue in scheduling plants with multiple stages deals with the presence of intermediate storage. Timing constraints relating consecutive stages depend on the storage assumptions. For instance, if ZW or NIS are used, the following constraints should be imposed:

$$Tei_{il} = Tsi_{il+1} \quad \forall i, l \in L_i - \{\bar{l}_i\}. \quad (18)$$

The above constraint states that the start time of order i in stage $l + 1$ should coincide with its completion time in stage l. When UIS is assumed, constraint (18) can be relaxed to

$$Tei_{il} \leq Tsi_{il+1} \quad \forall i, l \in L_i - \{\bar{l}_i\}. \quad (19)$$

Moreover, neglecting sequence-dependent changeovers, the start and completion times of stage l of order i are related in terms of the variables W_{ijkl} for ZW and UIS policies as follows:

$$Tei_{il} = Tsi_{il} + \sum_{j \in (J_i \cap J_l)} \sum_{k \in K_j} W_{ijkl} T_{ij} \quad \forall i, l \in L_i. \tag{20}$$

As NIS allows storing of intermediates in the units after their processing is done, equation (20) can be relaxed to represent this policy:

$$Tei_{il} \geq Tsi_{il} + \sum_{j \in (J_i \cap J_l)} \sum_{k \in K_j} W_{ijkl} T_{ij} \quad \forall i, l \in L_i. \tag{21}$$

As for the FIS policy, two approaches can be taken. The first one is to consider intermediate storage as another processing stage. Another approach is to model intermediate storage explicitly such as in Pinto and Grossmann [16], who considered a multistage continuous multiproduct plant with one unit per stage. Models based on time discretization, such as the STN network model, handle the modeling of finite inventory with greater ease.

Voudouris and Grossmann [30] developed a model for the sequential multipurpose plants with equipment in parallel. Their approach is to enumerate all the production paths for the products. In this case, the assignment variables are y_h that denote whether path h exists and $y'_{hh'j}$ that denote whether path h precedes path h' in unit j. The assignment and sequencing constraints are

$$y'_{hh'j} + y'_{h'hj} \geq y_h + y_{h'} - 1 \qquad \forall h, h', j \in (J_h \cap J_{h'}), \tag{22a}$$

$$\sum_{h'} \sum_{j \in (J_h \cap J_{h'})} (y'_{hh'j} + y'_{h'hj}) \leq y_h \qquad \forall h, \tag{22b}$$

$$\sum_{h} \sum_{j \in (J_h \cap J_{h'})} (y'_{hh'j} + y'_{h'hj}) \leq y_{h'} \qquad \forall h'. \tag{22c}$$

The sets of constraints (22a)–(22c) enforce that if paths h and h' are selected, then a precedence relationship between the paths has to be established in unit j, which is a common unit for both paths. Otherwise, the condition is relaxed. Moreover, timing constraints similar to the inequalities in (6) are defined. By defining variables Ts_{hj} as the start time of the operation of path h in unit j and parameters T_{hj} as the processing time of the operation of h in unit j yields

$$Ts_{h'j} - Ts_{hj} + U(1 - y'_{hh'j}) \geq T_{hj} \qquad \forall h, h', j \in (J_h \cap J_{h'}), \tag{22d}$$

$$Ts_{hj} - Ts_{h'j} + U(y'_{hh'j}) \geq T_{h'j} \qquad \forall h, h', j \in (J_h \cap J_{h'}). \tag{22e}$$

An alternative formulation relies on incorporating the assignment constraints (22a)–(22c) in (22d) and (22e), yielding

$$Ts_{h'j} - Ts_{hj} + U(3 - y'_{hh'j} - y_h - y_{h'}) \geq T_{hj} \qquad \forall h, h', j \in (J_h \cap J_{h'}), \tag{22f}$$

$$Ts_{hj} - Ts_{h'j} + U(2 + y'_{hh'j} - y_h - y_{h'}) \geq T_{h'j} \qquad \forall h, h', j \in (J_h \cap J_{h'}). \tag{22g}$$

The two formulations (22a)–(22e) and (22f)–(22g) generate the same integer solutions, although they exhibit significant differences in computational performance. Constraint set (22f)–(22g) relies on a smaller number of integer variables and constraints; however, its LP relaxation is looser. Note that when the paths are fixed, both formulations reduce to the equations (6) of Rich and Prokopakis [21].

2.3. Multiple-unit assignment – models based on event times

Another approach is to define global time slots, i.e., not associated with units but with events, as shown in figure 2. In general, scheduling models for networks of arbitrary topology are based on this representation of time that was suggested by Kondili et al. [11]. These formulations rely on discretization of time by considering time intervals of uniform size. Continuous time formulations can also be developed, as will be discussed later. Additionally, mass balance equations are key elements of these models thereby allowing a rather general treatment of batches.

The State Task Network (STN) representation introduced by Kondili et al. [11] relies on the representation of the process recipes as transformations of states (material) through a series of tasks in a process network. It is interesting to note that the STN readily handles multipurpose plants, but unlike the job shop case (see equations (6) from Rich and Prokopakis [21]), tasks are not pre-assigned to units. An STN representation for a simple multipurpose plant that processes three products, taken from Barbosa-Póvoa [2], is shown in figure 4(a). The STN contains the following original assignment and sequencing constraints [11]:

$$\sum_{i \in I_j} W_{ijk} \leq 1 \qquad \forall j, k, \tag{23a}$$

$$\sum_{i' \in I_j} \sum_{k'=k}^{k+T_{ij}-1} W_{i'jk'} - 1 \leq U_{ij}(1 - W_{ijk}) \qquad \forall i \in I_j,\ j, k. \tag{23b}$$

Constraints (23a) enforce that at any time k, an idle piece of equipment j can start at most one task i. Moreover, constraints (23b) state that if the item j does start performing a given task i, then it cannot start any other task until the current one is finished after T_{ij} time slots. Shah et al. [27] found an equivalent representation for the assignment and sequencing constraints in (23):

$$\sum_{i \in I_j} \sum_{k'=k}^{k-T_{ij}+1} W_{ijk'} \leq 1 \qquad \forall j, k. \tag{24}$$

The above equations correspond to the tightest formulation, which is based on a full backward aggregation of time. Constraints (24) ensure the same conditions of (23a) and (23b); the proof of equivalence is shown in Grossmann et al. [9]. Interestingly, constraint (24) not only yields a much tighter representation, but it also involves far fewer inequalities than equations (23a) and (23b).

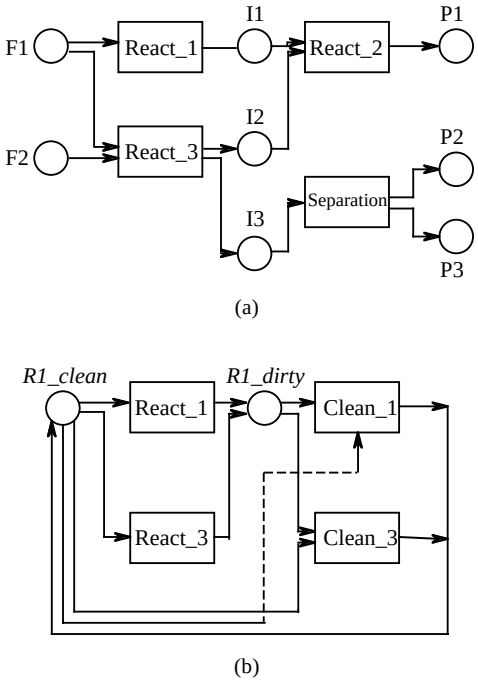

Figure 4. (a) Example of a State Task Network.
(b) Example of a Unit STN (Reactor 1).

In the STN formulation, the treatment of sequence-dependent events can be rather complex. Kondili et al. [11] proposed sequence-dependent constraints based on families of products and defined additional transition tasks $i_{ff'}$ that are activated according to the following constraints:

$$\sum_{i \in I_j^{(f)}} W_{ijk} + \sum_{i \in I_j^{(f')}} W_{ijk'} - \sum_{t=k+1}^{k'-1} \left(W_{i_{ff'}jt} + \sum_{i \in I_j} W_{ijt} \right) \leq 1 \qquad \forall f, f', j, k, k' > k. \quad (25)$$

Constraints (25) enforce that if unit j starts a task of family f at time k and a task of family f' at time k', without starting any other task between this interval, then the transition task $i_{ff'}$ is activated. A less restrictive constraint can be applied to sequence-dependent changeovers which, however, does not attempt to define the timing of the cleaning operation or take into account any demands for resources. As shown by Kondili et al. [11] and Shah [26], this yields a more aggregated constraint for the STN:

$$\sum_{i \in I_j^{(f)}} W_{ijk} + \sum_{i \in I_j^{(f')}} \sum_{k'=k-T_{ij}-\tau_{ff'}-1}^{k-T_{ij}} W_{ij\,k'} \leq 1 \qquad \forall f, f', j, k. \quad (26)$$

Constraint (26) states that if unit j starts processing any task of family f at time k, then no task of family f' could have started for at least $\tau_{ff'}$ units before the start of this task. Note that this formulation does not rely on extra transition tasks and has fewer constraints.

Crooks [6] addresses sequence-dependent changeovers that account for both tasks and units by introducing the Unit-STN. The representation comprises the states in which a unit can exist and the changes that occur when a task is carried out. Tasks that perform changeovers, such as cleaning and maintenance, are incorporated in the set of processing tasks defined. An example of a Unit-STN, taken from Barbosa-Póvoa [2], is shown in figure 4(b), in which a reactor R1 can exist in two states: R1_clean and R1_dirty. Two reactions (React_1 and React_3) can be performed in the reactor and both lead to the R1_dirty state. Two cleaning tasks (Clean_1 and Clean_3) drive it to the R1_clean state. Additional binary variables $\widehat{W}_{jkt}$ denote the states t of unit j at time k. In this formulation, the following constraints apply: (a) at any time k, unit j can exist in exactly one of the states t; (b) a unit j can start processing a task i only if the unit-state allows the operation; (c) if no task starts at time k, the unit-state remains unchanged at $k + 1$; (d) a task i that starts at k modifies the unit to a unique unit-state at $k + 1$. The conditions (a)–(d), as described in Barbosa-Póvoa [2], were formulated as

$$\sum_{t \in S_j} \widehat{W}_{jkt} = 1 \qquad \forall j, k, \tag{27a}$$

$$W_{ijk} \le \sum_{t : i \in \bigcup_{t'} I_{jtt'}} \widehat{W}_{jkt} \qquad \forall i \in I_j,\ j, k, \tag{27b}$$

$$\sum_{i \in \bigcup_{t'} I_{jtt'}} W_{ijt} \le \widehat{W}_{jkt} - \widehat{W}_{j,k+1,t} \le \sum_{i \in \cup I_{jtt'}} W_{ijk} \qquad \forall j, k, t \in S_j, \tag{27c}$$

$$\widehat{W}_{jkt} - \widehat{W}_{j,k+1,t'} + \sum_{i \in \bigcup_{t'} I_{jtt'}} W_{ijk} \le 1 \qquad \forall j, k, t \in S_j,\ t' \in S_j, \tag{27d}$$

where S_j is the set of allowed states for unit j and $I_{jtt'}$ is the set of tasks which cause transition from state t to state t' in unit j.

Consecutive processing stages are connected in the STN representation by the state nodes. Continuous variables S_{sk} and B_{ijk} are introduced to denote the amount of material stored in state s at the beginning of time k and the amount of material that starts task i in unit j at time k, respectively. In addition, variables D_{sk} and F_{sk} represent the sales and purchases of state s at the beginning of time k. A mass balance in these nodes is given by

$$S_{sk} + \sum_{i \in I_s} \rho_{is} \sum_{j \in J_i} B_{ijk} + D_{sk} = S_{s,k-1} + \sum_{i \in \bar{I}_s} \bar{\rho}_{is} \sum_{j \in J_i} B_{ij,k-T_{ij}} + F_{sk} \quad \forall k, s. \tag{28}$$

In (28), ρ_{is} and $\overline{\rho}_{is}$ are constants that represent for task i the proportions of input and output material in state s, respectively. The amount of material that can perform a task is limited by equipment size V_j:

$$0 \leq B_{ijk} \leq V_j W_{ijk} \qquad \forall i \in I_j,\ j, k. \tag{29a}$$

The STN representation allows a general treatment of intermediate storage. Capacity limits C_s are defined for storage of each state:

$$0 \leq S_{sk} \leq C_s \qquad \forall k, s. \tag{29b}$$

If UIS is imposed, then C_s is unbounded. NIS is obtained with $C_s = 0$. Zero-Wait policy can be imposed by adding constraints that specify that task i' follows task i, that is,

$$\sum_{j \in J_i} W_{ijk} = \sum_{j \in J_i} W_{i'j,\, k+T_{ij}} \qquad \forall k \in K - \{\bar{k} - T_{i'j} + 1, \ldots, \bar{k}\}. \tag{29c}$$

Pantelides [13] introduced a very concise representation for scheduling problems based on an extension of the STN framework, namely the Resource Task Network (RTN). The basic motivation is to describe processing equipment, storage, material transfer and utilities uniformly as resources, which are consumed and generated during the plant operation. Two variables represent the operation of task i at time k, the integer variables N_{ik} and the continuous variables ξ_{ik} that determine the demands that the tasks place on the various resources. The following equations express the available resource balances R_{rk}:

$$R_{rk} = R_{rk-1} + \sum_{i \in I_r} \sum_{k'=0}^{T_i} (\mu_{irk'} N_{i,k-k'} + \nu_{irk'} \xi_{i,k-k'}) + \Pi_{rk} \qquad \forall k, r, \tag{30a}$$

$$0 \leq R_{rk} \leq R_{rk}^U \qquad \forall k, r. \tag{30b}$$

In constraints (30a), μ_{irk} and ν_{irk} are known constants that may be positive if the resource is produced or negative if it is consumed. Moreover, Π_{rk} is the amount of resource r available from external sources at time k. The continuous variables ξ_{ik} may be related to the integer variables N_{ik} to impose operational constraints. Constraints (30b) simply bound the resources. Interestingly, Pantelides [13] showed that constraints (30a) correspond to the general case of the STN unit allocation constraints (26). Also, equations (30a) have the same form and can be easily reduced to the STN equations (28).

Xueya and Sargent [33] have developed continuous time formulations that are based on the STN and the RTN. The basic approach is to divide the time horizon into a sequence of event times of variable length T_k. In order to track all the operations, the start and completion of each task are mapped to the event times. Furthermore, they partition the processing time of each task in several transfer times to represent changes in operating conditions. Since the main concern is with scheduling problems, we

restrict the analysis to a single processing time. In Xueya and Sargent's formulation, binary assignment variables W_{ijk} are activated if the task starts at T_k in unit j and binary timing variables $X_{ijkk'}$ are activated if the task starts at instant T_k in unit j and is completed at $T_{k'}$. Therefore, constraints (31a) enforce that if task i starts in unit j at T_k, it will finish one and only one later time $T_{k'}$, $k' > k$:

$$W_{ijk} = \sum_{k'>k} X_{ijkk'} \quad \forall i \in I_j,\ j, k. \tag{31a}$$

Moreover, the following constraints ensure that at each time k', a unit must be occupied by at most one task:

$$\sum_{i \in I_j} \sum_{k \le k'} X_{ijkk'} \le 1 \quad \forall j, k'. \tag{31b}$$

As mentioned before, Xueya and Sargent [33] rely on time-events T_k, which are postulated in strictly ascending order. Therefore, the following constraint is imposed:

$$T_k - T_{k-1} \ge \varepsilon \quad \forall k - \{1\}. \tag{32}$$

In (32), the strict inequality is enforced by introducing the small parameter ε on the right-hand side. In order to define the sequence of events, the actual task processing time is identified with one of the event times $T_{k'}$. This is achieved by time constraints as follows:

$$t_{ijk} = \sum_{k'>k} X_{ijkk'}(T_{k'} - T_k) \quad \forall i \in I_j,\ j, k. \tag{33a}$$

Note that equation (33a) is in nonlinear form. A linearization procedure leads to constraints (33b), which are in fact very similar to the time-matching constraints (15). These are

$$-U(1 - X_{ijkk'}) \le T_{k'} - T_k - t_{ijk} \le U(1 - X_{ijkk'}) \quad \forall i \in I_j,\ j, k, k' \ge k, \tag{33b}$$

where U represent bounds on the time differences. A disaggregation procedure of variables t_{ijk} (analogous to the one developed for the time-matching constraints (15)–(17)) can be performed for the constraints in (33a):

$$t_{ijkk'} = X_{ijkk'}(T_{k'} - T_k) \quad \forall i \in I_j,\ j, k, k' \ge k. \tag{34a}$$

Similarly, equation (34a) is nonlinear. Applying linearization to (34a) yields constraints (34b) and (34c):

$$L\ X_{ijkk'} \le t_{ijkk'} \le U\ X_{ijkk'} \quad \forall i \in I_j,\ j, k, k' \ge k, \tag{34b}$$

$$-U(1 - X_{ijkk'}) \le T_{k'} - T_k - t_{ijkk'} \le U(1 - X_{ijkk'}) \quad \forall i \in I_j,\ j, k, k' \ge k. \tag{34c}$$

Constraint set (34b)–(34c) is tighter than (33b). For instance, if the binary timing variable $X_{ijkk'}$ is zero, it implies from equation (34b) that the time variable $t_{ijkk'}$ is

also zero. However, this is not the case in (33b). Nevertheless, the representation in (34b)–(34c) yields more constraints and variables.

As for changeovers, Xueya [34] introduces measures of usability U_{jku} that denote the usability u of a task in unit j at time k. The basic idea is to propose a formulation that comprises sequence-dependent changeovers as well as frequency- and time-dependent transitions. Parameters, based on precedence relations, are as follows: α_{iju} and β_{iju} reflect how the processing of a task affects the unit usability, while γ_{iju} represents the threshold usability of a task. Constraint (35a) relates the measures of usability in consecutive time events through assignment variables:

$$U_{j,k+1,u} = \sum_{i \in I_j} (\alpha_{iju} U_{ijku} + \beta_{iju} W_{ijk}) \qquad \forall j, k, u, \tag{35a}$$

while constraint (35b) is a feasibility condition for use of unit j for processing order i:

$$U_{jku} \geq \sum_{i \in I_j} \gamma_{iju} W_{ijk} \qquad \forall j, k, u, \tag{35b}$$

and constraints (35c)–(35d) represent the disaggregation of the measures of usability:

$$U_{jku} = \sum_{i \in I_j} U_{ijku} \qquad \forall j, k, u, \tag{35c}$$

$$0 \leq U_{ijku} \leq U_{ju}^{\max} W_{ijk} \qquad \forall i, j, k, u. \tag{35d}$$

Since there is no straightforward procedure to determine the set of parameters α_{iju}, β_{iju} and γ_{iju} (in fact, there is more than one feasible set), the simplest way of illustrating the concept of usability is through a small example. Consider the reactor R1, whose Unit-STN representation is presented in figure 4(b) and which was also used to illustrate changeovers defined by Crooks [6] for the STN in (27). Table 1 illustrates the set of precedence relations as well as the usability parameters involved.

Table 1

Xueya's [34] usability parameters.

Task i	Precedence sets	α_{iju}
R_1	{R_1, C_1, C_3}	0
R_3	{R_3, C_1, C_3}	0
C_1	{R_1}	0
C_3	{R_3}	0

Task i	$\beta_{ij\text{UR\_1}}$	$\beta_{ij\text{UR\_3}}$	$\beta_{ij\text{UC\_1}}$	$\beta_{ij\text{UC\_3}}$	$\gamma_{ij\text{UR\_1}}$	$\gamma_{ij\text{UR\_3}}$	$\gamma_{ij\text{UC\_1}}$	$\gamma_{ij\text{UC\_3}}$
R_1	1	0	1	0	1	0	0	0
R_3	0	1	0	1	0	1	0	0
C_1	1	1	0	0	0	0	1	0
C_3	1	1	0	0	0	0	0	1

According to the sets of precedence relations, task R_3 is the only one that cannot immediately precede task R_1; moreover, the only task that can precede C_3 is R_3. In this small problem, the usabilities are: $u \in \{UR\_1, UR\_3, UC\_1, UC\_3\}$. Parameters α_{iju} are all zero, since all tasks change the unit usability after the processing. Note that parameters β_{iju} reflect the precedence relations. As for the feasibility parameters γ_{iju}, they are chosen in such a way that a task can be assigned to a unit only if the usability of the same task is one.

3. Computational experience

The objective of this section is not to present a detailed computational comparison among the various scheduling models. The idea is simply to provide a general description of the specific methods and summarize results for the largest scheduling problems reported.

Table 2 provides a summary of the main features of the process scheduling models. Note that the entries correspond to the items discussed in the road-map of figure 1, as well as the optimization objectives used in the models. The models follow the same order of presentation used in the previous sections. The first five entries are single-unit assignment models, while the remaining are based on multiple-unit assignments. The complete model by Birewar and Grossmann is presented in [3] and a specialized aggregation procedure for Zero-Wait transfer is developed in [4]. Pinto and Grossmann [16] and Sahinidis and Grossmann [23] developed models for continuous multiproduct plants (in which processing times are variables); the former models a multistage plant with one unit per stage, while the latter deals with a single-stage parallel-unit plant. Also, the entry of the model by Shah et al. [28] corresponds to the extension of the short term model of Kondili et al. [11] and Shah et al. [27] to cyclic scheduling. Although it was not mentioned explicitly in the previous sections, it relies on the STN representation; the constraints are modified to accommodate the periodic mode of operation. The entry corresponding to the model by Schilling et al. [25] can be regarded in terms of assignment and sequencing as somewhat similar to the model by Xueya and Sargent [33] in the sense that it relies on continuous time represented by events.

Single-unit assignment models search through a much smaller integer space and are, therefore, in principle able to solve larger problems. Moreover, simpler model structures facilitate the development of specialized solution methods. For instance, Pekny and Miller [14] developed a special purpose code for solving asymmetric TSP problems for the case of Zero-Wait transfer. Also, Birewar and Grossmann [4] were able to develop an aggregated LP model by transforming a Zero-Wait model from the space of batches to the space of products. Another example is the work by Gooding et al. [8], in which they were able to transform their single-stage, parallel unit model with sequence-dependent changeovers into a member of the TSP problems, namely the Branch Office TSP. On the other hand, models based on process networks provide

Table 2

Assignment and sequencing models.

Model	Ref.	Plant topology	Mass balances	Representation time/model	Resource constraints	Transfer policy	Demand pattern	Changeovers	Optimization objective
Birewar and Grossmann	[3] [4]	multistage one unit/stage	lots	continuous linear	No	UIS, NIS ZW	cyclic	sequence dependent	cycle
Pinto and Grossmann	[16]	multistage one unit/stage	network flow	continuous nonlinear	No	FIS	cyclic	sequence dependent	profit
Pekny and Miller	[14]	multistage ident. parallel units	lots	continuous linear	No	ZW	short term	sequence dependent	cost
Rich and Prokopakis	[21]	multipurpose one unit/stage	lots	continuous linear	No	ZW	short term	unit dependent	tardiness
Sahinidis and Grossmann	[23]	single stage parallel units	lots	continuous nonlinear	No	–	cyclic	sequence dependent	cost
Cerdá et al.	[5]	single stage parallel units	lots	continuous linear	Yes	–	short term	sequence dependent	tardiness
Gooding et al.	[8]	single stage parallel units	lots	continuous linear	No	–	short term	sequence dependent	cost
Pinto and Grossmann	[17]	multistage parallel units	lots	continuous linear	No	UIS, NIS ZW	short term	unit dependent	earliness
Voudouris and Grossmann	[30]	seq. multipurp. parallel units	lots	continuous linear	No	MIS [1)]	cyclic	sequence dependent	makespan
Kondili et al Shah et al.	[11] [27]	network	network flow	discrete linear	Yes	FIS	short term	family dependent	profit
Shah et al.	[28]	network	network flow	discrete linear	Yes	FIS	cyclic	family [2)] dependent	profit
Pantelides	[13]	network	network flow	continuous linear	Yes	FIS	short term	No	profit
Schilling et al.	[25]	multipurpose	network	discrete linear	Yes	FIS	short term	family dependent	profit
Xueya and Sargent	[33]	network	network flow	continuous linear	Yes	FIS	short term	sequence dependent	profit

[1)] Mixed Intermediate Storage: zero-wait subtrains separated by a number of intermediate storage vessels.
[2)] Original formulation. More efficient treatment of changeovers with the Unit-STN by Crooks [6].

Table 3

Largest scheduling problems reported.

Model	Largest problem reported					
	Features	Margin of optimality	Discrete variables	Continuous variables	Constraints	CPU time
Birewar and Grossmann[1)]	50 products/9 units 2392 batches	0.028	2500	2500	2610	24.2 CPU min GAMS/ZOOM Microvax II
Pinto and Grossmann	8 products/3 stages	0.00	448	1920	3002	48.8 CPU min GAMS/OSL DICOPT++ HP 9000-730
Pekny and Miller[2)]	500 batches/1 line 3 equal units	0.00	$\approx 2.5 \times 105$	–	1000 + subtour	4.78 CPU sec SunSparc 2
Rich and Prokopakis	12 batches 43 units	0.00	16	22	48	1.83 CPU min LINDO–IBM PC
Sahinidis and Grossmann	26 products 3 parallel units	0.02	780	23,000	3200	22 CPU min GAMS/MPSX Minos IBM 3090
Cerdá et al.	20 orders 4 units	0.00	122	61	288	2.38 CPU min Sciconic IBM RS6000
Gooding et al.[3)]	36 batches 3 parallel units	0.003	≈ 900	–	> 1800	2.5 CPU h HP 9000-730
Pinto and Grossmann	50 orders/5 stages 25 units	0.00	1050	47,917	48,843	4.15 CPU h GAMS/OSL HP 9000-730
Voudouris and Grossmann	3 products/5 paths 96 batches/7 units	0.00	24	82	141	16 CPU sec GAMS/Sciconic IBM RS6000
Kondili et al.	12 states/6 tasks 6 units/3 weeks horizon/6 h int.	0.01	1173	2193	1446	15.5 CPU min SunSparc I
Shah et al.	11 states/9 tasks 9 units/24 h cycle	0.05	552	775	753	94 CPU sec SunSparc 2
Pantelides[5)]	6 units/36 h horizon/1 h int.	0.01	648	1368	2377	26.2 CPU min SunSparc 10
Schilling et al.[6)]	3 products/12 states 8 tasks/2 units	n.a.	266	502	1124	n.a. Sciconic
Xueya and Sargent	25 states/22 tasks 11 units 120 h horizon	0.07	1318	4555	4801	18.1 CPU min Cplex SunSparc 10/41

[1)] ZW aggregated model in space of products. Reported in Birewar and Grossmann [4].
[2)] Asymmetric TSP. No continuous variables. Number of binary variables ≈ (number of jobs)^2.
[3)] Branch office TSP. No continuous variables. 1800 constraints reported for problem with 30 batches.
[4)] Results reported in Shah et al. [27].
[5)] Results reported in Wilkinson et al. [31].
[6)] No information; no report on CPU resources, 3527 nodes searched.

a very general structure for scheduling. The foremost example is the State Task Network developed by Kondili et al. [11] and Shah et al. [27], which has been extended to produce a number of formulations covering many process scheduling problems. However, many instances of problems resulting from larger industrial applications are not solvable, due to the size of the mixed integer programming problems.

Another very important aspect is the diversity in scheduling models with respect to time domain representation. As shown in table 2, continuous time models have typically been applied to simpler structures such as sequential plants, while discrete time models have shown the capability of representing complex task networks. Moreover, discrete time representations handle resource constraints naturally. Nevertheless, Xueya and Sargent [33] recently developed a continuous time model with the capability of dealing with network structures as well as resource constraints. The computational requirements, however, are quite expensive.

It can also be observed that the objectives are in general explicit cost functions. In these cases, demands are normally specified as lower bounds rather than amounts to be exactly met. In the case of fixed demands, time based objective functions are usually adopted; however, even in these situations, cost minimization is indirectly implied, such as the minimization of cycle time/makespan and minimization of earliness/tardiness, which implicitly enforce minimization of inventory costs. All problems are formulated as MILP models, with the exception of Sahinidis and Grossmann [23] and Pinto and Grossmann [16], which are MINLPs. Also, Xueya and Sargent [33] model the problem originally as an MINLP with bilinear constraints, which are then linearized exactly for solving the example problems.

Table 3 contains the largest instances reported for the various scheduling problems in a variety of computer platforms. It is important to note that only results corresponding to scheduling problems were included. For instance, both Birewar and Grossmann [3] and Voudouris and Grossmann [30] consider simultaneous design and scheduling; nevertheless, examples corresponding to the simultaneous cases were excluded. Most solution methods rely on commercial LP-based branch and bound codes. Although not all the problems in table 3 were solved to optimality, it can be seen that in a number of cases, the computation time and size of the problems are becoming of industrial significance. Also, in most cases, the size of the MILP models is becoming rather large, although the number of binary variables for unstructured mixed integer problems is limited to less than 1,500. It will be interesting to see in the next few years the extent to which larger and more general problems can be tackled.

4. Concluding remarks

Given the diversity of optimization models developed for the scheduling of process systems over the last decade, the major goal of this paper has been to provide a classification of problems and a unified overview of models. The major classification adopted in this paper closely follows the combinatorics implied by the various

scheduling problems, largely in terms of the complexity of the plant structure. It has been shown that the most significant factor that distinguishes the models is whether tasks are assigned to units or not. The major emphasis has also been to review the specific equations of the various assignment and sequencing models to show both their similarities and differences, particularly with regard to the general problem classification. It is clear that to have an increased impact in practice, the research effort should be directed to address more complex plant structures, as well as larger instances.

5. Notation

Indices

f, f' product families,
h, h' production paths,
i, i' products (orders, tasks, batches),
j units,
$\bar{j}$ last processing unit,
k, k' time slots or time event points,
$\bar{k}$ last time slot,
$\overline{k_j}$ last time slot defined for unit j,
l, l' production stages,
$\bar{l}_i$ stage at which order i is withdrawn,
r resources,
s states,
t unit-states,
u task usability.

Sets

I set of orders (tasks),
I_j set of orders (tasks) which can be processed in unit j ($I_j \subseteq I$),
$I_{jtt'}$ set of tasks which cause transition from state t to t' in unit j,
I_r set of tasks that utilize resource r ($I_r \subseteq I$),
I_s set of tasks that receive material from state s ($I_s \subseteq I$),
I_s set of tasks that receive material from state s ($I_s \subseteq I$),
$\bar{I}_s$ set of tasks that produce material for state s ($\bar{I}_s \subseteq I$),
J set of units,
J_i set of units which can process order i ($J_i \subseteq J$),
J_l set of units which belong to stage l ($J_l \subseteq J$),
K set of time slots,
K_j set of time slots postulated for unit j ($K_j \subseteq K$),

L set of stages,
L_i stages involved in the production of i ($L_i \subseteq L$),
L_j stage corresponding to unit j ($L_j \subseteq L$),
S_j set of allowed states for unit j.

Parameters

C_s capacity of storage of state s,
L lower bound on time differences,
T_{ij} processing time of product i in unit j,
U upper bound on time differences,
U^{il} upper bound on start times for order i in stage l,
α_{iju} usability parameter,
β_{iju} usability parameter,
γ_{iju} threshold usability parameter,
μ_{irk} consumption (production) of resource r by task i at time k,
ν_{irk} consumption (production) of resource r by task i at time k,
ρ_{is} proportion input to task i from state s,
$\overline{\rho}_{is}$ proportion output of task i for state s,
$\tau_{ii'j}$ transition time from product i to product i' in unit j.

Variables

B_{ijk} amount of material for task i in unit j at time k,
D_{sk} amount from sales of state s at time k,
F_{sk} amount from purchases of state s at time k,
N_{ik} discrete variable associated with task i at time k,
R_{rk} available resource r at time k,
S_{sk} amount of material in state s at time k,
t_{ijk} processing time of task i in unit j for time slot k,
T_k event time k,
Te_{jk} completion time in unit j for time slot k,
Tei_{il} completion time for product i in stage l,
Ts_{hj} starting time of path h in unit j,
Ts_{jk} starting time in unit j for time slot k,
Tsi_i starting time for operation i,
Tsi_{il} starting time for product i in stage l,
U_{jku} measure of usability u in unit j at time k,
U_{ijku} measure of usability u for task i in unit j at time k,
W_{ijk} binary variable that assigns product i to time slot k of unit j,
W_{ijkl} binary variable that assigns stage l of order i to time slot k of unit j,

$\widehat{W}_{jkt}$ binary variable that denotes the state t of unit j at time k,
X_{ik} binary variable that assigns product i to slot k,
$X'_{ii'}$ binary variable that denotes the processing of product i immediately before product i',
$X_{ijkk'}$ binary timing variable that is activated if operation j of product i starts at T_k and ends at $T_{k'}$,
y_h binary variable that denotes the existence of path h,
$y'_{hh'j}$ binary variable that denotes the whether path h precedes path h' in unit j,
y_{jk} slack variable that denotes empty slot k at unit j; see equations (14b) and (17d),
$Z_{ii'k}$ binary transition variable that is activated if product i is assigned at the beginning of time slot k to be followed by product i',
$Z_{ii'jk}$ binary transition variable that is activated if product i is assigned at the beginning of time slot k in unit j to be followed by product i',
θ_{ijkl} start time of stage l of product i in slot k unit j,
γ_{jk} slack time variable for slot k of unit j,
Π_{rk} available resource r at time k,
ξ_{ik} continuous variable associated with task i at time k.

Appendix I: Derivation by linearization of sequence dependent constraints (8)

Transition variables $Z_{ii'jk}$ have to be linked to assignment variables W_{ijk} in such a way that

$$W_{ijk} \wedge W_{i'j,k+1} \Rightarrow Z_{ii'jk} \qquad \forall i \in I_j,\ i' \in I_j,\ k \in K_j - \{\overline{k_j}\}. \tag{I.1}$$

Sahinidis and Grossmann [23] compared two formulations that enforce proposition (I.1). The following constraint (I.2) can be obtained directly from propositional calculus [32]:

$$Z_{ii'jk} \geq W_{ijk} + W_{i'j,k+1} - 1 \qquad \forall i \in I_j,\ i' \in I_j,\ j,k \in K_j - \{\overline{k_j}\}. \tag{I.2}$$

Another formulation proposed is the one corresponding to constraints (8):

$$\sum_{i \in I_j} Z_{ii'jk} = W_{i'j,k+1} \quad \forall i' \in I_j,\ j,k \in K_j - \{\overline{k_j}\}, \tag{8a}$$

$$\sum_{i' \in I_j} Z_{ii'k} = W_{ijk} \qquad \forall i \in I_j,\ j,k \in K_j - \{\overline{k_j}\}. \tag{8b}$$

Sahinidis and Grossmann [23] showed that constraints (8) in conjunction with the constraint

$$\sum_{i \in I_j} W_{ijk} = 1 \qquad \forall j,k \in K_j \tag{I.3}$$

and the integrality of the assignment variables W_{ijk} have the unique solution

$$Z_{ii'jk} = W_{ijk}W_{i'j,k+1} \quad \forall i \in I_j,\ i' \in I_j,\ j, k \in K_j - \{\overline{k_j}\}. \tag{I.4}$$

In fact, (I.4) is the algebraic equivalent of (I.1). We show here that constraints (8) can be obtained by linearization. Adding (I.4) over i and adding (I.4) over i' yields (I.5a) and (I.5b), respectively:

$$\sum_{i \in I_j} Z_{ii'jk} = \sum_{i \in I_j} W_{ijk}W_{i'j,k+1} \quad \forall i' \in I_j,\ j, k \in K_j - \{\overline{k_j}\}, \tag{I.5a}$$

$$\sum_{i' \in I_j} Z_{ii'jk} = \sum_{i' \in I_j} W_{ijk}W_{i'j,k+1} \quad \forall i \in I_j,\ j, k \in K_j - \{\overline{k_j}\}. \tag{I.5b}$$

As $W_{i'j,k+1}$ does not depend on i in (I.5a) and W_{ijk} does not depend on i' in (I.5b), they can be removed from the summation terms. Using (I.3) yields

$$\sum_{i \in I_j} Z_{ii'jk} = W_{i'j,k+1} \quad \forall i' \in I_j,\ j, k \in K_j - \{\overline{k_j}\}, \tag{8a}$$

$$\sum_{i' \in I_j} Z_{ii'jk} = W_{ijk} \quad \forall i \in I_j,\ j, k \in K_j - \{\overline{k_j}\}. \tag{8b}$$

Appendix II: On the equivalence of the allocation constraints for the discrete-time parallel-line scheduling model by Gooding et al. [8]

The proof of equivalence of constraints (10c) and (11) with constraint (13) follows the same development as in Grossmann et al. [9]. Index sets are omitted for clarity of presentation. Sahinidis and Grossmann [24] proposed a reformulated allocation constraint by using a disaggregated form of (23b),

$$W_{ijk} + W_{i'jk'} \le 1 \quad \forall i, i', j, k, k' = k-1, \ldots, k + T_{i'j} - 1. \tag{II.1}$$

The demonstration is then based on the equivalence of constraints (II.1) and constraints (24) by Shah et al. [27]. It relies on a backward time aggregation and on the following

Lemma (Grossmann et al. [9]). The integer constraint

$$y_1 + y_2 + \cdots + y_k + y_{k+1} \le 1 \tag{II.2}$$

is equivalent to and sharper than the set of integer constraints:

$$y_1 + y_2 + \cdots + y_k \le 1, \tag{II.3}$$

$$y_1 + y_{k+1} \le 1, \tag{II.4-1}$$

$$y_2 + y_{k+1} \le 1, \tag{II.4-2}$$

$$\cdots$$

$$y_k + y_{k+1} \le 1. \tag{II.4-k}$$

Proof. First we note that (II.2) can easily be seen to be equivalent to the constraints (II.4-1) to (II.4-k) since in these, at most one variable can take an integer value of 1. Multiplying constraint (II.3) by $(k-1)$ and adding it to the constraints in (II.4-1)–(II.4-k) yields

$$k(y_1 + y_2 + \cdots + y_k + y_{k+1}) \leq k - 1 + k,$$

$$y_1 + y_2 + \cdots y_k + y_{k+1} \leq 1 + (k-1)/k.$$

Since $y_i = \{0, 1\}$, the right-handside can be rounded down to obtain the inequality

$$y_1 + y_2 + \cdots y_k + y_{k+1} \leq 1.$$

□

Proof of constraint (13). First we present a disaggregated form of equation (11), which is in fact similar to equation (II.1) and states the fact that if job i is processed on unit j in slot k, no job can begin production during the next $T_{ij} + \tau_{ii'j} - 1$ slots associated with the production of i and the transition from i to i':

$$Z_{ii''jk} + \sum_{i'''} Z_{i'i'''jk'} \leq 1 \quad \forall i, i', i'', j, k, k' = k+1, \ldots, k + T_{ij} + \tau_{ii''j} - 1.$$

We know, from (10c),

$$\sum_{i} \sum_{i''} Z_{ii''jk} \leq 1 \quad \forall j, k, \tag{10c}$$

which in expanded form yields

$$\sum_{i''} Z_{i1i''jk} + \sum_{i''} Z_{i2i''jk} + \cdots + \sum_{i''} Z_{i|I|i''jk} \leq 1 \quad \forall j, k.$$

If $T_{i1j} + \tau_{i1i2j} > 1$,

$$Z_{i1i2jk-1} + \sum_{i''} Z_{i1i''jk} \leq 1 \qquad \forall j, k,$$

$$Z_{i1i2jk-1} + \sum_{i''} Z_{i2i''jk} \leq 1 \qquad \forall j, k,$$

$$\ldots$$

$$Z_{i1i2jk-1} + \sum_{i''} Z_{i|I|i''jk} \leq 1 \quad \forall j, k.$$

From the lemma, we obtain

$$Z_{i1i2jk-1} + \sum_{i''} Z_{i1i''jk} + \sum_{i''} Z_{i2i''jk} + \cdots + \sum_{i''} Z_{i|I|i''jk} \leq 1 \quad \forall j, k.$$

If $T_{i1j} + \tau_{i1i3j} > 1$,

$$Z_{i1i3jk-1} + \sum_{i''} Z_{i1i''jk} \leq 1 \quad \forall j, k,$$

$$Z_{i1i2\,jk-1} + \sum_{i''} Z_{i2i''jk} \le 1 \qquad \forall j, k,$$

$$\cdots$$

$$Z_{i1i3\,jk-1} + \sum_{i''} Z_{i\,|\,I\,|\,i''jk} \le 1 \quad \forall j, k.$$

Also,

$$Z_{i1i2\,jk-1} + Z_{i1i3\,jk-1} \le 1 \quad \forall j, k.$$

This leads to

$$Z_{i1i3\,jk-1} + Z_{i1i2\,jk-1} + \sum_{i''} Z_{i1i''jk} + \sum_{i''} Z_{i2i''jk} + \cdots + \sum_{i''} Z_{i\,|\,I\,|\,i''jk} \le 1 \quad \forall j, k.$$

Repeat for all i, $i' \neq i$ for which $T_{ij} + \tau_{ii'j} > 1$ to get

$$\sum_{i''} Z_{i1i''jk-1} + \cdots + \sum_{i''} Z_{i\,|\,I\,|\,i''jk-1} + \sum_{i''} Z_{i1i''jk} + \cdots + \sum_{i''} Z_{i\,|\,I\,|\,i''jk} \le 1 \quad \forall j, k.$$

Now, if $T_{i1j} + \tau_{i1i2j} > 2$,

$$Z_{i1i2\,jk-2} + \sum_{i''} Z_{i1i''jk-1} \le 1 \qquad \forall j, k,$$

$$Z_{i1i2\,jk-2} + \sum_{i''} Z_{i2i''jk-1} \le 1 \qquad \forall j, k,$$

$$\cdots$$

$$Z_{i1i2\,jk-2} + \sum_{i''} Z_{i\,|\,I\,|\,i''jk-1} \le 1 \qquad \forall j, k,$$

$$Z_{i1i2\,jk-2} + \sum_{i''} Z_{i1i''jk} \le 1 \qquad \forall j, k,$$

$$Z_{i1i2\,jk-2} + \sum_{i''} Z_{i2i''jk} \le 1 \qquad \forall j, k,$$

$$\cdots$$

$$Z_{i1i2\,jk-2} + \sum_{i''} Z_{i\,|\,I\,|\,i''jk} \le 1 \qquad \forall j, k.$$

From the lemma,

$$Z_{i1i''jk-2} + \sum_{i''} Z_{i1i''jk-1} + \cdots + \sum_{i''} Z_{i\,|\,I\,|\,i''jk-1} + \sum_{i''} Z_{i1i''jk} + \cdots + \sum_{i''} Z_{i\,|\,I\,|\,i''jk} \le 1 \ \ \forall j, k.$$

Repeat for all $Z_{ii'jk-2}$ for $T_{ij} + \tau_{ii'j} > 2$.
Repeat for all $Z_{ii'jk-3}$ for $T_{ij} + \tau_{ii'j} > 3$.
…
Repeat for all $Z_{ii'jk-k'+1}$ for $T_{ij} + \tau_{ii'j} > k' - 1$.

Finally, we obtain

$$
\begin{aligned}
&Z_{i1,i2,j,k-T_{1j}-\tau_{i1i2j}+1} &&+\cdots+ Z_{i1,i2,j,k-1} + Z_{i1,i2,j,k} +\\
&Z_{i1,i3,j,k-T_{1j}-\tau_{i1i3j}+1} &&+\cdots+ Z_{i1,i3,j,k-1} + Z_{i1,i3,j,k} +\\
&\cdots + &&\\
&Z_{i1,i|I|,j,k-T_{1j}-\tau_{i1i|I|j}+1} &&+\cdots+ Z_{i1,i|I|,j,k-1} + Z_{i1,i|I|,j,k} +\\
&\cdots + &&\\
&Z_{i|I|,i1,j,k-T_{i|I|j}-\tau_{i|I|i1j}+1} &&+\cdots+ Z_{i|I|,i1,j,k-1} + Z_{i|I|,i1,j,k} +\\
&\cdots + &&\\
&Z_{i|I|,i|I-1|,j,k-T_{i|I|j}-\tau_{i|I|i|I-1|j}+1} &&+\cdots+ Z_{i|I|,i|I-1|,j,k-1} + Z_{i|I|,i|I-1|,j,k} \le 1 \quad \forall j,k.
\end{aligned}
$$

Grouping terms in the above inequality yields

$$
\begin{aligned}
&\sum_{k'=k-T_{i1j}-\tau_{i1i2j}+1}^{k} Z_{i1,i2,j,k'} + \sum_{k'=k-T_{i1j}-\tau_{i1i3j}+1}^{k} Z_{i1,i3,j,k'}\\
&\quad +\cdots+ \sum_{k'=k-T_{i1j}-\tau_{i1i|I|j}+1}^{k} Z_{i1,i|I|,j,k'}\\
&\quad +\cdots+ \sum_{k'=k-T_{i|I|j}-\tau_{i|I|i1j}+1}^{k} Z_{i|I|,i1,j,k'} + \sum_{k'=k-T_{i|I|j}-\tau_{i|I|i2j}+1}^{k} Z_{i|I|,i2,j,k'}\\
&\quad +\cdots+ \sum_{k'=k-T_{i|I|j}-\tau_{i|I|i|I-1|j}+1}^{k} Z_{i|I|,i|I-1|,j,k'} \le 1 \quad \forall j,k.
\end{aligned}
$$

Summing over all i':

$$
\sum_{i'} \sum_{k'=k-T_{i1j}-\tau_{i1i'j}+1}^{k} Z_{i1,i',j,k'} +\cdots+ \sum_{i'} \sum_{k'=k-T_{i|I|j}-\tau_{i|I|i'j}+1}^{k} Z_{i|I|,i',j,k'} \le 1 \quad \forall j,k.
$$

Further summing over all i, we obtain constraint (13):

$$
\sum_{i} \sum_{i'} \sum_{k'=k-T_{ij}-\tau_{ii'j}+1}^{k} Z_{i,i',j,k'} \le 1 \quad \forall j,k. \tag{13}
$$

□

Acknowledgements

The authors would like to acknowledge support received from FAPESP (Fundação de Amparo à Pesquisa do Estado de São Paulo, Brazil) under grant 91/1186-0 and from the National Science Foundation under grant CTS-9209671.

References

[1] K.R. Baker, *Introduction to Sequencing and Scheduling*, Wiley, New York, 1974.

[2] A.P. Barbosa-Póvoa, Detailed design and retrofit of multipurpose batch plants, Ph.D. Thesis, University of London, London, 1994.

[3] D.B. Birewar and I.E. Grossmann, Incorporating scheduling in the optimal design of multiproduct batch plants, Computers Chem. Eng. 13(1989)141–161.

[4] D.B. Birewar and I.E. Grossmann, Efficient optimization algorithms for zero-wait scheduling of multiproduct plants, I&EC Research 28(1989)1333–1345.

[5] J. Cerdá, G. Henning and I.E. Grossmann, A mixed-integer linear programming model for short term batch scheduling in parallel lines, presented at the *ORSA/TIMS Meeting – Global Manufacturing in the 21st Century*, Detroit, MI, 1994.

[6] C.A. Crooks, Synthesis of operating procedures for chemical plants, Ph.D. thesis, University of London, London, 1992.

[7] S. French, *Sequencing and Scheduling: An Introduction to the Mathematics of the Job-Shop*, Ellis Horwood, England, 1982.

[8] W.B. Gooding, J.F. Pekny and P.S. McCroskey, Enumerative approaches to parallel flowshop scheduling via problem transformation, Computers Chem. Eng. 18(1994)909–927.

[9] I.E. Grossmann, J. Quesada, R. Raman and V. Voudouris, Mixed integer optimization techniques for the design and scheduling of batch processes, in: *Batch Processing Systems Engineering*, eds. G.V. Reklaitis, A.K. Sunol, D.W.T. Rippin and O. Hortacsu, Springer, Berlin, 1996, pp. 451–494.

[10] J.N.D. Gupta, Optimal flowshop schedules with no intermediate storage space, Nav. Res. Logist. Quart. 23(1976)235–243.

[11] E. Kondili, C.C. Pantelides and R.W.H. Sargent, A general algorithm for short-term scheduling of batch operations. I. MILP formulation, Computers Chem. Eng. 17(1993)211–227.

[12] H. Ku, D. Rajagopalan and I.A. Karimi, Scheduling in batch processes, Chem. Eng. Prog. 83 (1987)35-45.

[13] C.C. Pantelides, Unified frameworks for optimal process planning and scheduling, in: *Foundations of Computer Aided Process Operations*, eds. D.W.T. Rippin, J.C. Hale and J.F. Davis, CACHE, Austin, TX, 1994, pp. 253–274.

[14] J.F. Pekny and D.L. Miller, Exact solution of the no-wait flowshop scheduling problem with a comparison to heuristic methods, Computers Chem. Eng. 15(1991)741–748.

[15] J.F. Pekny and M.G. Zentner, Learning to solve process scheduling problems: The role of rigorous knowledge acquisition frameworks, in: *Foundations of Computer Aided Process Operations*, eds. D.W.T. Rippin, J.C. Hale and J.F. Davis, CACHE, Austin, TX, 1994, pp. 275–309.

[16] J.M. Pinto and I.E. Grossmann, Optimal cyclic scheduling of multistage continuous multiproduct plants, Computers Chem. Eng. 18(1994)797–816.

[17] J.M. Pinto and I.E. Grossmann, A continuous time mixed-integer linear programming model for short term scheduling of multistage batch plants, I&EC Research 34(1995)3037–3051.

[18] J. Pinto and I.E. Grossmann, An alternate MILP model for short term batch scheduling with preordering constraints, Ind. Eng. Chem. Research 35(1996)338–342.

[19] G.V. Reklaitis, Perspectives on scheduling and planning of process operations, presented at the *4th International Symposium on Process Systems Engineering*, Montebello, Canada, 1991.

[20] G.V. Reklaitis, Overview of scheduling and planning of batch process operations, *NATO Advanced Study Institute – Batch Process Systems Engineering*, Antalya, Turkey, 1992.

[21] S.H. Rich and G.J. Prokopakis, Scheduling and sequencing of batch operations in a multipurpose plant, Ind. Eng. Chem. Process Des. Dev. 25(1986)979–988.

[22] D.W.T. Rippin, Batch process systems engineering: a retrospective and prospective review, Computers Chem. Eng. 17(suppl. issue)1993, S1–S13.

[23] N.V. Sahinidis and I.E. Grossmann, MINLP model for cyclic multiproduct scheduling on continuous parallel lines, Computers Chem. Eng. 15(1991)85–103.

[24] N.V. Sahinidis and I.E. Grossmann, Reformulation of multiperiod MILP models for planning and scheduling of chemical processes, Computers Chem. Eng. 15(1991)255–272.
[25] G. Schilling, Y.-E. Pineau, C.C. Pantelides and N. Shah, Optimal scheduling of multipurpose continuous plants, presented at the *AIChE National Meeting*, San Francisco, CA, 1994.
[26] N. Shah, Efficient scheduling, planning and design of multipurpose batch plants, Ph.D. Thesis, University of London, London, 1992.
[27] N. Shah, C.C. Pantelides and R.W.H. Sargent, A general algorithm for short-term scheduling of batch operations. II. Computational issues, Computers Chem. Eng. 17(1993)229–244.
[28] N. Shah, C.C. Pantelides and R.W.H. Sargent, Optimal periodic scheduling of multipurpose batch plants, Ann. Oper. Res. 42(1993)193–228.
[29] V.T. Voudouris and I.E. Grossmann, Optimal synthesis of multiproduct batch plants with cyclic scheduling and inventory considerations, Ind. Eng. Chem. Res. 32(1993)1962–1980.
[30] V.T. Voudouris and I.E. Grossmann, An MILP model for the optimal design and scheduling of a special class of multipurpose plants, Computers Chem. Eng. 20(1996)1335–1360.
[31] S.J. Wilkinson, N. Shah and C.C. Pantelides, Scheduling of multisite flexible production systems, paper presented at the *AIChE Annual Meeting*, San Francisco, CA, 1994.
[32] P. Williams, *Model Building in Mathematical Programming*, 2nd ed., Wiley, Chichester, Northern Ireland, 1985.
[33] Z. Xueya and R.W.H. Sargent, The optimal operation of mixed production facilities – a general formulation and some approaches to the solution, *Proceedings of the 5th Symposium on Process Systems Engineering*, Kyongju, Korea, 1994.
[34] Z. Xueya, Algorithms for optimal process scheduling using nonlinear models, Ph.D. Thesis, University of London, London, 1995.