Generic mathematical formulations for scheduling of multipurpose batch plants

Nikolaos Rakovitis,[1] Yueting Pan,[1] Nan Zhang,[1] Jie Li,[1,*] and Giorgos Kopanos[2]

[1]Centre for Process Integration, Department of Chemical Engineering and Analytical Science, The University of Manchester, Manchester, M13 9PL, United Kingdom
[2]Flexciton Limited, London, 145 City Rd, Hoxton, London EC1V 1AZ

Abstract

In this work, we develop two generic mixed-integer linear programming formulations for scheduling of multipurpose batch plants using the unit-specific event-based modelling approach. While related non-recycling production and consumption tasks are allowed to take place at the same event points but in different real time in the first model, they are not allowed in the second model. We also introduce the concept of indirect and direct material transfer, which allows to conditionally align the operational sequence of related production and consumption tasks. Processing units are able to hold materials previously produced over multiple event points. The computational results demonstrate that the proposed models do not require a task to span over multiple event points to generate the optimal solution. As a result, the proposed models are able to generate the same or better solutions with up to one order of magnitude less computational time compared to the existing models.

Keywords: Scheduling, Multipurpose batch processes, mixed-integer linear programming, unit-specific event-based approach

[*] To whom correspondence should be addressed. Email: jie.li-2@manchester.ac.uk. Tel: +44 (0) 161 306 8622

1 Introduction

Multipurpose batch plants widely exist in the chemical industry for the production of a large number of low-volume, high-value products. To achieve higher utilization of resources, lower inventory costs and better responsiveness to a fluctuating manufacturing environment, optimal scheduling of the multipurpose facilities is desirable and has attracted much interest of both academia and industry in the past decades. Many mathematical formulations have been developed using either the State Task Network (STN) representation[1] or the Resource Task Network (RTN) representation[2]. These models can be classified based on the time representation of the scheduling horizon into discrete-time and continuous-time representations. The discrete-time representation divides the scheduling horizon into time intervals with fixed and known length. The time intervals can either be or non-uniform[3] within the scheduling horizon, and a task or activity can only start and finish at these time intervals. The continuous-time representation uses time points, slots, or event points to divide the scheduling horizon with a variable and unknown length. It can be further classified into global event-based[4-6], slot-based including process-slot based[7-8] and unit-slot based[8-9], unit-specific event-based[10-17] and sequence-based[18-20] time representations. These mathematical models are also classified into single- and multiple- time grid mathematical models[21]. For more details about these time representations, the reader can be referred to[22-24], which provide excellent reviews for scheduling in chemical industries.

All existing time-grid mathematical models divide the scheduling horizon using time points/slots/event points on which a task or activity can both start and finish. Therefore, the number of time points/slots/event points required directly affects the efficiency of the existing mathematical models. More specifically, an increase in the total time points/slots/event points can lead to an exponential increase in the number of binary variables, continuous variables and constraints. This can potentially increase the computational time by at least one order of magnitude is required to generate the optimal solution. Some task must be allowed to span over multiple time points/slots/event points to generate the optimal solution, which further increases the computational burden. The capabilities of the unit-specific event-based formulations are well established in the literature[11-12, 17]. However, they still require excessive computational time for industrial-scale problems due to unnecessary event points required to generate the optimal solution. The main possible reason is that most existing unit-specific event-based formulations unconditionally impose that a consumption task starts after its related production tasks (with the same states) even if the consumption task does not consume materials from the related production tasks, or there is enough storage available.

Two works[14-15] in the literature have attempted to relax such unconditional alignment. Seid and Majozi[14] investigated whether consumption tasks consume materials from storage tanks and whether materials produced by production tasks can be stored in the storage tanks. For the former case, if there are not enough materials in the storage tanks for all consumption tasks, then the unconditional

sequencing of all related production and consumption tasks are imposed. In the latter case, if a production task produces materials that cannot be stored in the storage tanks, then all related production and consumption tasks should be unconditionally aligned. The problem of unconditional sequencing of related production and consumption tasks even if the consumption task does not consume any materials from the production task is still not addressed. Furthermore, they did not consider to conditionally align a production task with a related consumption task, if the producing materials can be stored in storage tanks. Additionally, their formulation can generate schedules with a real-time violation, as demonstrated by Vooradi and Shaik[15]. To address these issues, Vooradi and Shaik[15] explicitly examined if a consumption task consumes materials from a specifically related production task or if there is enough storage for materials produced by a specific production task. They sequenced a production task with a related consumption task only if the consumption task consumes materials from the production task. Additionally, they aligned a production task with a related consumption task only if storage tanks cannot store the materials from a specific production task. With this approach, they have managed to further reduce the number of event points in comparison to the model of Seid and Majozi[14], while they avoided generating a solution with a real-time violation. However, Vooradi and Shaik[15] used an increased number of binary variable sets to denote whether tasks have to be sequenced or aligned during an event point in their model, leading to computational inefficiency. Most of the existing models fail to generate the optimal solution in some cases, especially when the materials have to be temporarily stored in processing units, as illustrated later.

In this work, we develop two generic mixed-integer linear programming formulations for scheduling of multipurpose batch plants using the unit-specific event-based modelling approach. While we follow the approach of Rakovitis et al[17] to allow all related non-recycling production and consumption tasks to take place at the same event points but in different real times in the first model, we do not allow all related non-recycling production and consumption tasks to take place at the same event points in the second model. We also introduce the concept of indirect and direct material transfer, which allows us to conditionally and unconditionally align the operational sequences of related production and consumption tasks. More specifically, production and consumption tasks related to the same state are sequenced if there is an indirect material transfer between the units that are processing these tasks, while we align them if there is a direct material transfer between these units. Additionally, we allow the processing units to hold materials that are previously produced from these units over multiple event points. Nonsimultaneous material transfer[8] is also allowed in both models. We solve several well-established examples in the literature to illustrate the capability of the proposed formulations. The computational results demonstrate that both models require a smaller number of binary variables in most cases, especially in the cases where a processing unit can process

multiple tasks, compared to the existing mathematical formulation[15]. It is interesting to note that the proposed models do not need to allow a task to span over multiple event points to generate the optimal solution. As a result, the computational time is significantly reduced by one order of magnitude in most cases. More importantly, the proposed models can generate better solutions than the existing models such as Vooradi and Shaik[15] and Mostafaei and Harjunkoski[21]. Additionally, the first model, which allows related non-recycling production and consumption tasks to take place at the same event points is slightly more efficient than the second one. Finally, we use the proposed model to solve a large-scale industrial batch plant scheduling problem from Janak et al.[25] using the rolling-horizon decomposition algorithm. The results demonstrate that the proposed model can improve productivity by 26.7% in significantly less computational time compared to that of Janak et al.[25].

2 Problem statement

Figure 1 illustrates a general STN representation of a multipurpose batch plant. There are I (i = 1, 2, 3, …, I) tasks that are processed in total J (j = 1, 2, …, J) processing units. In a batch plant, a task means heating, reaction, separation, and so on. Each unit can process $\mathbf{I}_j$ suitable tasks. Multiple tasks are allowed to be processed in a processing unit. However, at most one task can be processed in a unit at a time. Raw materials, intermediate products, and final products are denoted as states in the STN representation. There are S (s = 1, 2, 3, …, S) states in total. The raw materials are denoted as $\mathbf{S}^R$, intermediate materials are denoted as $\mathbf{S}^{IN}$ and final products are denoted as $\mathbf{S}^P$. A state is consumed or produced by $\mathbf{I}_s$ tasks including $\mathbf{I}_s^C$ consumption tasks and $\mathbf{I}_s^P$ production tasks. The proportion of a state s that a task i in a unit j consumes or produces is known which is denoted by a parameter ρ_{sij}. While this proportion parameter is positive if the state is produced, it is negative if the state is consumed. A task i on unit j processes a batch size (b_{ij}) of material state. The processing time is assumed to be a linear function of the batch size, which is calculated by $\alpha_{ij} + \beta_{ij} \cdot b_{ij}$.

Once a batch is produced in a processing unit, it may be transferred immediately or remain in the processing unit for some limited or unlimited time before it is transferred. It may be transferred into a dedicated storage tank, split into different small batches or mixed together with other batches for downstream processing. In other words, batch splitting and mixing are allowed. There are several different storage policies including unlimited intermediate storage (UIS) policy and finite intermediate storage (FIS) policy. With this, the entire scheduling problem can be stated as follows,

Given:

a) J units, suitable $\mathbf{I}_j$ tasks, minimum (b_{ij}^{min}) and maximum (b_{ij}^{max}) capacities and constant processing time coefficients;
b) S states, suitable $\mathbf{I}_s$ tasks including production tasks and consumption tasks, detailed processing paths and recipes, their initial inventories, and minimum and maximum capacities.
c) The production recipe (i.e., the coefficients of processing time for each task, and the

consuming or producing proportions of each batch).

The product prices;

d) The scheduling horizon for maximization of productivity problems or the product demand for minimization of makespan problems.

Determine:

a) Optimal production schedule including allocation, sequence, timings of tasks in a unit;
b) The amount of material being processed in each unit at each time;
c) Inventory profiles of all material states through the scheduling horizon.

Operating rules:

a) At most one task can be processed in a unit at a time;
b) Batch mixing and splitting is allowed.

Assumptions:

a) All parameters are deterministic with no batch/unit failures or operational interruptions;
b) The processing time of a task in a processing unit depends on the batch size;
c) Unlimited feed materials are available;
d) Unlimited storage policy for raw materials and final products;
e) Unlimited or Finite storage policy for intermediate products;
f) Unlimited resources are available;
g) Unlimited wait policy for intermediate states.
h) Negligible transfer times between units (i.e., processing units and storage units).
i) Setup or changeover times are lumped into batch processing times.
j) All processing units can hold a batch temporarily before its start and after its end.
k) Each material state has its dedicated storage unit.

We consider two objectives. The first objective is to maximize the productivity in the given scheduling horizon. The second objective is to minimize the total time required to fulfill the product demand, which is known as minimization of makespan.

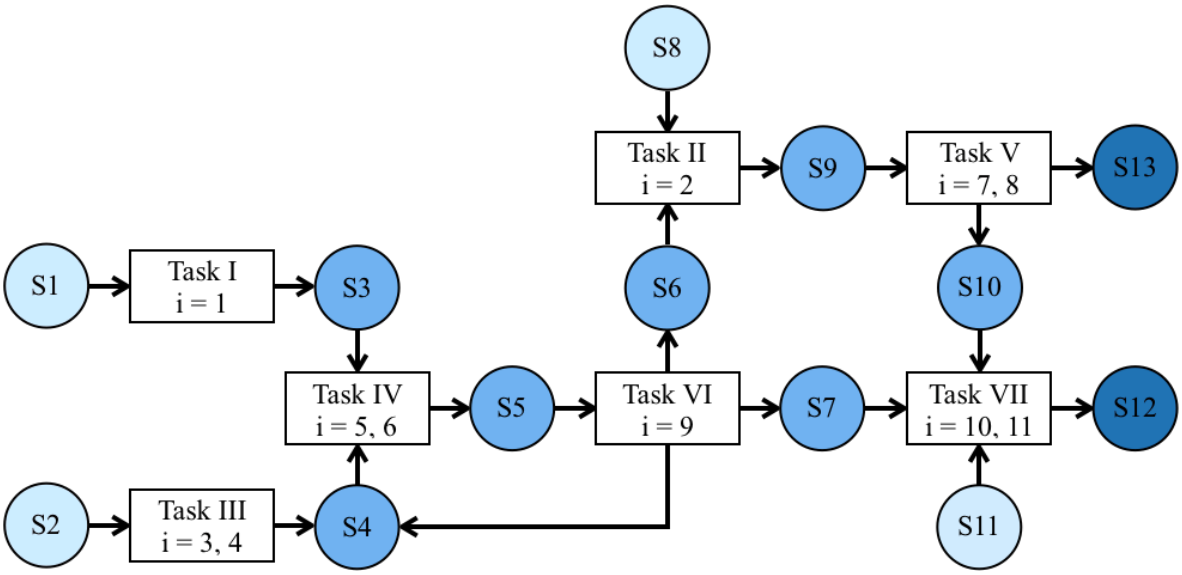

Figure 1 STN representation of a multipurpose batch plant

3 Motivating Example 1

Let consider a motivating example. The data is given in Table 1. The STN is illustrated in Figure 2. There are two processing units (J1-J2), two tasks (I1-I2) and three states (S1-S3). I1 is processed on unit J1 and I2 is processed on unit J2. There is no initial amount for the intermediate state S2. The maximum storage capacity of state S2 is 10 mu.

Table 1 Data for the Motivating Example

Task	Processing Unit	α_i	β_i	B_i^{min}	B_i^{max}
1	1	3.00	0.02	0	100
2	2	1.00	0.01	0	50

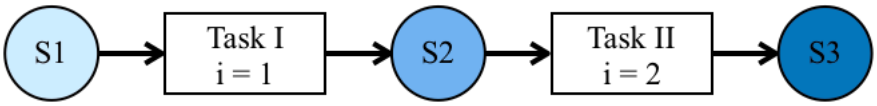

Figure 2 STN representation of the Motivating Example

We use the mathematical models of Li and Floudas[12], Vooradi and Shaik[15] and Mostafaei and Harjunkoski[21] to solve this motivating example. The computational results are provided in Table 2. The optimal schedule with maximum productivity of 300 cu obtained from the existing models[12, 15, 21] is illustrated in Figure 3. We can observe that 60 cu of S2 is produced by task I1 in unit J1 at 5 hr. Then, 50 cu of S2 is consumed immediately after it is produced. 10 cu of S2 is stored in the storage tank, which does not violate the storage capacity. Another 10 cu of S2 are also consumed at 7 hr. Finally, product S3 with a total price of 300 cu is produced. It can be concluded that the intermediate S2 is immediately transferred to storage tank and consumption unit after it is produced. However, we

can generate another schedule with a maximum productivity of 500 cu through trial and errors, as illustrated in Figure 4. This means that all these existing models generate suboptimal solution for this example. Through detailed analysis of the schedules in Figures 3 and 4, the possible reason is that the schedule in Figure 4 allows material S2 produced in the unit J1 to be held in this unit. Only 50 cu is transferred into unit J2 for further processing after it is produced. Even though the model of Vooradi and Shaik[15] allows materials to be temporarily stored at the production task, they can only be stored at event point N1. For N2, the materials have to be either consumed by a consumption task or stored in the storage task. However, since there is no storage available, J1 cannot produce the same amount of state S2. Instead, only 60 units can be produced. Therefore, the model of Vooradi and Shaik[15] also fail to generate the optimum solution. This example motivates us to develop a new generic mathematical formulation with consideration of these additional features that can result in a significant increase in the productivity of the batch plant.

Table 2 Computational results for Motivating Example 1 from the models of Li and Floudas[12], Vooradi and Shaik[15] and Mostafaei and Harjunkoski[21]

Example	Model	Number of event points	CPU time (s)	RMILP	MILP (h)	Binary Variables	Continuous Variables	Constraints
M. E.	LF2010	3	0.11	500.00	300.00	6	29	41
(H = 8 h)	VS2013	3	0.09	500.00	300.00	14	33	72
	MH2019	4 (ΔR=1)	0.08	500.00	300.00	6	34	78

LF2010: Li and Floudas[12] model. VS2013: Vooradi and Shaik[15] model. MH2019: Mostafaei and Harjunkoski[21]

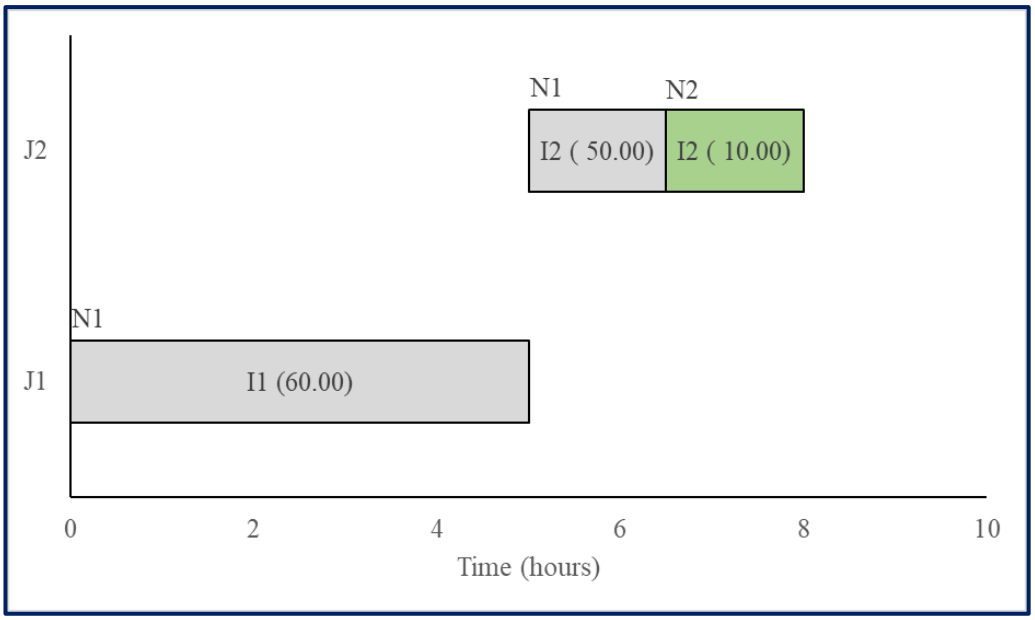

Figure 3 Optimal schedule for the Motivating Example 1 with maximum productivity of 300 cu

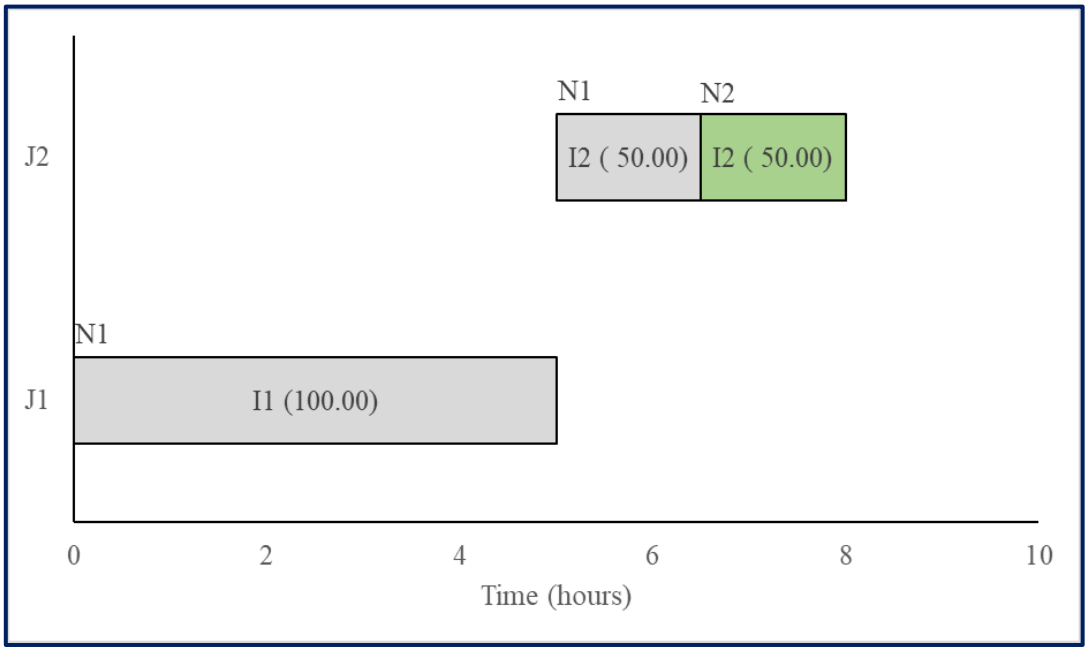

Figure 4 A feasible schedule for the Motivating Example 1 with maximum productivity of 500 cu

4 Generic mathematical formulation

It is of great importance to represent time horizon for scheduling problems before developing mathematical formulation. Although there are several existing time representations for scheduling problems including discrete-time, slot-based, global event-based, unit-specific event-based, and sequence-based time representations as discussed, the well-established unit-specific event-based time representation is adopted in this work because it often leads to a smaller model size and less computational effort in comparison to other time representations. The details about this time representation can be referred to Ierapetritou and Floudas[10].

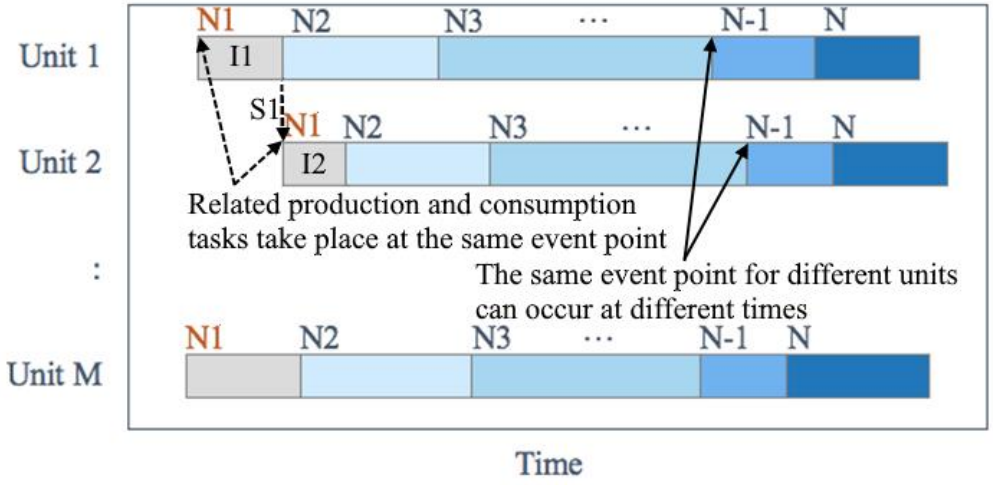

Figure 5 The unit-specific event-based representation where related production and consumptions tasks are allowed to take place at the same event points

4.1 Model M1

In this model **M1**, we allow production and consumption tasks related to the same state to take place at the same event points, which is similar to those of Rakovitis et al.[17]. We also use the definition of recycling tasks presented on Rakovitis et al.[17] and we only allow non-recycling production and consumption tasks related to the same state to take place at the same event point. Furthermore, the timing variables are defined based on units, not tasks. The unit-specific event-based time representation for model **M1** is illustrated in Figure 5. In Figure 5, task I1 produces S1, which is consumed by task I2. I1 and I2 takes place at the same event point N1 but in different real time.

4.1.1 Allocation constraints

We introduce four-index binary variables $w_{ijnn'}$ to denote the allocation of tasks to units below,

$$w_{ijnn'} = \begin{cases} 1 & \text{if a task } i \text{ is processed in a unit } j \text{ from an event point } n \text{ to } n' \\ 0 & \text{otherwise} \end{cases}$$

where $n \le n' \le n + \Delta n$. The parameter Δn is used to denote the maximum number of event points that a task is allowed to span over.

Based on the operating policy, at most one task is allowed to be processed in a processing unit at a time.

$$\sum_{i \in \mathbf{I}_j} \sum_{n-\Delta n \le n' \le n} \sum_{n \le n'' \le n'+\Delta n} w_{ijn'n''} \le 1 \qquad \forall j, n \qquad (1)$$

4.1.2 Capacity constraints

The amount of materials processed in a unit j should not exceed its minimum (B_{ij}^{min}) and maximum (B_{ij}^{max}) capacities.

$$B_{ij}^{\min} \cdot w_{ijnn'} \le b_{ijnn'} \le B_{ij}^{\max} \cdot w_{ijnn'} \qquad \forall j,\ i \in \mathbf{I}_j,\ n \le n' \le n+\Delta n \qquad (2)$$

4.1.3 Material balance constraints

The amount of a state s that has to be stored at event point n (ST_{sn}) should be equal to the amount of the state that has been stored at event point $(n-1)$, plus the amount of the state produced by recycling tasks at event point $(n-1)$ and by non-recycling tasks at event point n, minus the amount of the state consumed at event point n. At the first event point, the amount of a state s that has to be stored should be equal to the initial amount of the state ($ST0_s$) plus the amount of the state produced by non-recycling tasks, minus the amount of state s consumed at event point n.

$$ST_{sn} = ST_{s(n-1)} + \sum_{i \in \mathbf{I}_s^P \setminus \mathbf{I}^R} \sum_{j \in (\mathbf{J}_s \cap \mathbf{J}_i)} \sum_{n-\Delta n \le n' \le n} \rho_{sij} \cdot b_{ijn'n} + \sum_{i \in (\mathbf{I}_s^P \cap \mathbf{I}^R)} \sum_{j \in (\mathbf{J}_s \cap \mathbf{J}_i)} \sum_{(n-1)-\Delta n \le n' \le (n-1)} \rho_{sij} \cdot b_{ijn'(n-1)}$$
$$+ \sum_{i \in \mathbf{I}_s^C} \sum_{j \in (\mathbf{J}_s \cap \mathbf{J}_i)} \sum_{n \le n' \le n+\Delta n} \rho_{sij} \cdot b_{ijnn'} \qquad \forall s,\ n > 1 \qquad (3)$$

$$ST_{sn} = ST0_s + \sum_{i \in \mathbf{I}_s^P \setminus \mathbf{I}^R} \sum_{j \in (\mathbf{J}_s \cap \mathbf{J}_i)} \sum_{n-\Delta n \le n' \le n} \rho_{sij} \cdot b_{ijn'n} + \sum_{i \in \mathbf{I}_s^C} \sum_{j \in (\mathbf{J}_s \cap \mathbf{J}_i)} \sum_{n \le n' \le n+\Delta n} \rho_{sij} \cdot b_{ijnn'} \qquad \forall s,\ n = 1 \qquad (4)$$

where $I_S^P \backslash I^R$ means all production tasks except recycling tasks.

4.1.4 Duration constraints

The finish time of a unit *j* at event point *n* must be after its start time plus the processing time of the task i that the unit starts processing at event point *n*.

$$T_{jn}^{f} \geq T_{jn}^{s} + \sum_{i \in \mathbf{I}_j} \sum_{n \leq n' \leq n+\Delta n} \left(\alpha_{ij} \cdot w_{ijnn'} + \beta_{ij} \cdot b_{ijnn'} \right) \qquad \forall j, n \qquad (5)$$

4.1.5 Material transfer

Marterial transfer in the batch process is more flexible and complex compared to that in the continuous process. There are several scenarios of material transfer. Figure 6 illustrates all scenarios of material transfer. First, materials can be transferred to storage or downstream processing units immediately after production (e.g. material transfer MT1 in Figure 6). Second, materials can be held in the production units after production and then transferred to storage or downstream processing units (e.g. material transfer MT2 in Figure 6). If the storage capacity is large enough, then material can be first transferred to storage and then transferred to the downstream processing units (e.g. material transfer MT3 in Figure 6). If the storage capacity is not enough large, then some material has to be transferred directly to the downstream processing units (e.g. material transfer MT4 in Figure 6). In addition, materials produced from several production units can be transferred at the same time to storage or downstream processing units. This is called simultaneous material transfer. Alternatively, material produced from several production units can be transferred to storage or downstream processing units at different times, which is called nonsimultaneous material transfer. We generally classify the material transfer as indirect and direct material transfer. If all material is transferred to storage tank first and then to downstream processing units, then it is indirect material transfer. Otherwise, it is direct material transfer.

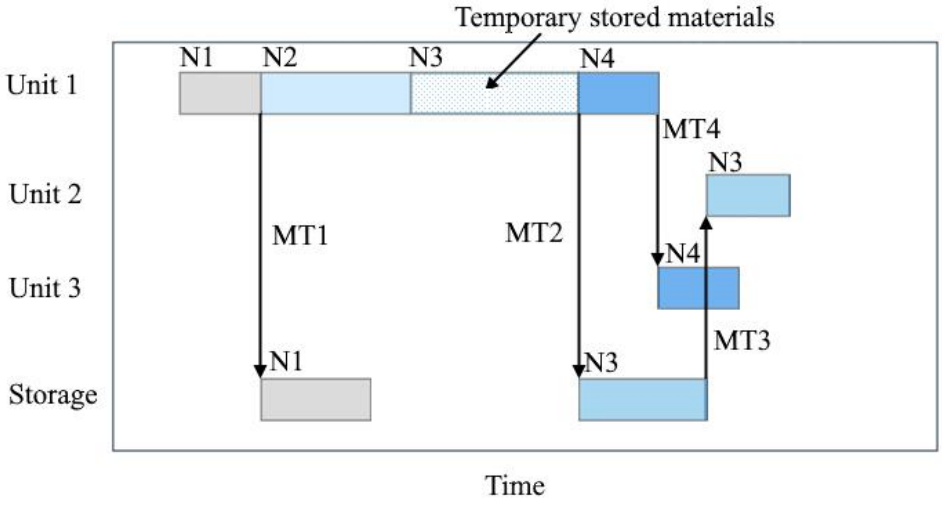

Figure 6 Different scenarios of material transfer

Indirect material transfer

In this scenario, the storage capacity is usually large enough. As a result, materials produced can always be transferred to the storage first and then transferred to the downstream processing units from the storage. It is an indirect material transfer from the production units to the downstream consumption units. To model this indirect material transfer, we define an additional binary variable $zI_{jj'n}$ as follows,

$$zI_{jj'n} = \begin{cases} 1 & \text{if material transfer happens between units } j \text{ and } j' \text{ at event point } n \\ 0 & \text{otherwise} \end{cases} \qquad \forall\, j \neq j', n$$

We also define continuous variables $bTi_{iji'j'n}$ to denote the amount of material transferred from a production task i in unit j to a consumption task i' in unit j' at event point n. Note that the material is first transferred from the production task i to the storage tank and then it is transferred to a consumption task i'. Therefore, it is an indirect material transfer from the production task i to the consumption task i'. The total amount of materials through indirect transfer from a production task i should not exceed that produced from this task i.

$$\rho_{sij} \cdot \sum_{n-\Delta n \le n' \le n} b_{ijn'n} \ge \sum_{j' \in \mathbf{J}_s} \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C)} bTi_{iji'j'n} \qquad \forall s \in \mathbf{S}^{IN}, j \in \mathbf{J}_s, i \in (\mathbf{I}_j \cap \mathbf{I}_s^P) \backslash \mathbf{I}^R, n \qquad (6)$$

$$\rho_{sij} \cdot \sum_{(n-1)-\Delta n \le n' \le (n-1)} b_{ijn'(n-1)} \ge \sum_{j' \in \mathbf{J}_s} \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C)} bTi_{iji'j'n} \qquad \forall s \in \mathbf{S}^{IN}, j \in \mathbf{J}_s, i \in (\mathbf{I}_j \cap \mathbf{I}_s^P \cap \mathbf{I}^R), n > 1 \qquad (7)$$

While constraint (6) is used for non-recycling tasks, constraint (7) is proposed for recycling tasks only.

Similarly, the amount of materials through indirect transfer to a consumption task i' at a time should not exceed the amount of materials consumed by this consumption task at event point n.

$$-\rho_{si'j'} \cdot \sum_{n \le n' \le n+\Delta n} b_{i'j'nn'} \ge \sum_{j \in \mathbf{J}_s} \sum_{i \in (\mathbf{I}_j \cap \mathbf{I}_s^P)} bTi_{iji'j'n} \qquad \forall s \in \mathbf{S}^{IN}, j' \in \mathbf{J}_s, i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C), n \qquad (8)$$

The total amount of materials consumed at event point n should not exceed the material stored at previous event point $(n-1)$ plus the amount of materials through indirect transfer.

$$\sum_{j' \in \mathbf{J}_s} \sum_{i' \in (\mathbf{I}_s^C \cap \mathbf{I}_{j'})} \left(-\rho_{si'j'} \cdot \sum_{n \le n' \le n+\Delta n} b_{i'j'nn'} \right) \le ST_{s(n-1)} + \sum_{j \in \mathbf{J}_s} \sum_{j' \in \mathbf{J}_s} \sum_{i \in (\mathbf{I}_j \cap \mathbf{I}_s^P)} \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C)} bTi_{iji'j'n} \qquad \forall s \in \mathbf{S}^{IN}, n \qquad (9)$$

When there is no indirect material transfer between two processing units, the amount through this indirect transfer should be zero.

$$\sum_{i \in (\mathbf{I}_j \cap \mathbf{I}_s^P)} \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C)} bTi_{iji'j'n} \le \min\left[B_j^{\max}, B_{j'}^{\max} \right] \cdot zI_{jj'n} \qquad \forall s \in \mathbf{S}^{IN}, j \neq j', j \in \mathbf{J}_s, j' \in \mathbf{J}_s, n \qquad (10)$$

where $B_j^{max} = \max_{i \in (\mathbf{I}_j \cap \mathbf{I}_s^P)} [B_{ij}^{max}]$ and $B_{j'}^{max} = \max_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C)} [B_{ij'}^{max}]$.

Direct material transfer

For states with FIS policy, if there is no storage available then these states cannot be transferred to a storage tank. Instead, they must be transferred directly from the production task i to a consumption task i'. For such a direct material transfer, we introduce an additional binary variable $zD_{jj'n}$ as follows,

$$zD_{jj'n} = \begin{cases} 1 & \text{if there is a direct material transfer between units } j \text{ and } j' \text{ at event point } n \\ 0 & \text{otherwise} \end{cases} \qquad \forall\, j \neq j', n$$

Similar to indirect material transfer, we also define continuous variables $bTd_{iji'j'n}$ to denote the amount of material directly transferred from a production task i in unit j to a consumption task i' in unit j' at event point n. The amount of materials directly transferred from between processing a production task i in unit j and a consumption task i' in unit j' must not exceed the amount of state produced from production task i. Constraints (11) and (12) are used for non-recycling tasks and recycling tasks respectively.

$$\rho_{sij} \cdot \sum_{n-\Delta n \le n' \le n} b_{ijn'n} + bs_{ijn} \ge \sum_{j' \in \mathbf{J}_s} \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C)} bTd_{iji'j'n} \qquad \forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}), j \in \mathbf{J}_s, i \in (\mathbf{I}_j \cap \mathbf{I}_s^P) \backslash \mathbf{I}^R, n \qquad (11)$$

$$\rho_{sij} \cdot \sum_{n-1-\Delta n \le n' \le n-1} b_{ijn'(n-1)} + bs_{ij(n-1)} \ge \sum_{j' \in \mathbf{J}_s} \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C)} bTd_{iji'j'n}$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}), j \in \mathbf{J}_s, i \in (\mathbf{I}_j \cap \mathbf{I}_s^P \cap \mathbf{I}^R), n > 1 \qquad (12)$$

The amount of materials through direct transfer to a consumption task i' at a time should not exceed the amount of materials consumed by this consumption task at event point n.

$$-\rho_{si'j'} \cdot \sum_{n \le n' \le n+\Delta n} b_{i'j'nn'} \ge \sum_{j \in \mathbf{J}_s} \sum_{i \in (\mathbf{I}_j \cap \mathbf{I}_s^P)} bTd_{iji'j'n} \qquad \forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}), j' \in \mathbf{J}_s, i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C), n \qquad (13)$$

A direct material transfer between a production task i in unit j and a consumption task i' in unit j' takes place only if the amount of state s produced at event point n for recycling tasks or at event point $(n-1)$ for non-recycling tasks, plus the amount of state s stored at event point $(n-1)$ exceeds the maximum storage capacity, plus the amount of materials stored in processing units. In this case, there are no storage tanks or processing units to temporary store the materials produced.

$$\sum_{j \in \mathbf{J}_s} \sum_{i \in \left(\mathbf{I}_j \cap \mathbf{I}_s^P\right) \backslash \mathbf{I}^R} \left(\rho_{sij} \sum_{n-\Delta n \le n' \le n} b_{ijn'n} \right) + ST_{s(n-1)} \le ST_s^{\max} + \sum_{j \in \mathbf{J}_s} \sum_{j' \in \mathbf{J}_s} \sum_{i \in \left(\mathbf{I}_{j'} \cap \mathbf{I}_s^P\right) \backslash \mathbf{I}^R} \sum_{i' \in \left(\mathbf{I}_{j'} \cap \mathbf{I}_s^C\right),} bTd_{iji'j'n} + \sum_{j \in \mathbf{J}_s} \sum_{i \in \left(\mathbf{I}_j \cap \mathbf{I}_s^P\right) \backslash \mathbf{I}^R} bs_{ij(n+1)}$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}), n \qquad (14)$$

$$\sum_{j \in \mathbf{J}_s} \sum_{i \in \left(\mathbf{I}_j \cap \mathbf{I}_s^P \cap \mathbf{I}^R\right)} \left(\rho_{sij} \sum_{n-1-\Delta n \le n' \le n-1} b_{ijn'(n-1)} \right) + ST_{s(n-1)} \le ST_s^{\max} + \sum_{j \in \mathbf{J}_s} \sum_{j' \in \mathbf{J}_s} \sum_{i \in \left(\mathbf{I}_j \cap \mathbf{I}_s^P \cap \mathbf{I}^R\right)} \sum_{i' \in \left(\mathbf{I}_{j'} \cap \mathbf{I}_s^C\right),} bTd_{iji'j'n} + \sum_{j \in \mathbf{J}_s} \sum_{i \in \left(\mathbf{I}_j \cap \mathbf{I}_s^P \cap \mathbf{I}^R\right)} bs_{ijn}$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}), n > 1 \qquad (15)$$

where variable $bs_{i,j,n}$ denotes the amount of materials stored in a unit j at event point n, previously

produced by task i in this unit, which will be explained later.

When there is no direct material transfer between two related processing units, the amount through this direct transfer should be zero, similar to the indirect material transfer.

$$\sum_{i\in(\mathbf{I}_j\cap\mathbf{I}_s^P)}\sum_{i'\in(\mathbf{I}_{j'}\cap\mathbf{I}_s^C)} bTd_{iji'j'n} \le \min\left[B_j^{\max}, B_{j'}^{\max}\right]\cdot zD_{jj'n} \qquad \forall s\in(\mathbf{S}^{IN}\cap\mathbf{S}^{FIS}), j\ne j', j\in\mathbf{J}_s, j'\in\mathbf{J}_s, n \qquad (16)$$

where $B_j^{max} = \max_{i\in(\mathbf{I}_j\cap\mathbf{I}_s^P)}[B_{ij}^{max}]$ and $B_{j'}^{max} = \max_{i'\in(\mathbf{I}_{j'}\cap\mathbf{I}_s^C)}[B_{ij'}^{max}]$.

4.1.6 Sequencing constraints

Different tasks in the same unit

The start time of a unit j at event point $(n + 1)$ must always be after its end time at the previous event point n.

$$T_{j(n+1)}^{\mathrm{f}} \ge T_{jn}^{\mathrm{s}} \qquad \forall j, n<N \qquad (17)$$

Different task in different unit

In order to make sure correct operational sequences between production and consumption tasks in different processing units, we define continuous variables T_{sjn} to denote the time when a state s produced by a unit j is available to be transferred (i.e., consumed or stored) at event point n. Then we require that the time when a state s produced by a unit j is available to be consumed at event point $(n + 1)$ is always after the time when the state is available at the previous event point n.

$$T_{sj(n+1)} \ge T_{sjn} \qquad \forall s\in\mathbf{S}^{IN}, j\in\mathbf{J}_s, n<N \qquad (18)$$

When a state s produced by a unit j is available at event point n, the production of this state in the same unit j must be completed at this event point n. In other words,

$$T_{sjn} \ge T_{jn}^{\mathrm{f}} - M\left(1-\sum_{i\in(\mathbf{I}_j\cap\mathbf{I}_s^P)}\sum_{n-\Delta n\le n'\le n} w_{ijn'n}\right) \qquad \forall s\in\mathbf{S}^{IN}, j\in\mathbf{J}_s, \sum_{i\in(\mathbf{I}_j\cap\mathbf{I}_s^P)}\rho_{sij}>0, n \qquad (19)$$

If a unit j' processes a task i', which consumes state s at event point n and also receives materials from unit j, then this unit should start after the time that state s, which was produced by unit j from a non-recycling task at event point n, is available.

$$T_{sjn} \le T_{j'n}^{\mathrm{s}} + M\left(1-zI_{jj'n}\right) \qquad \forall s\in\mathbf{S}^{IN}, j, j'\in\mathbf{J}_s, j\ne j', \sum_{i\in(\mathbf{I}_j\cap\mathbf{I}_s^P)\setminus\mathbf{I}^R}\rho_{sij}>0, \sum_{i'\in(\mathbf{I}_{j'}\cap\mathbf{I}_s^C)}\rho_{si'j'}<0, n \qquad (20)$$

Similarly, if a unit j' process a task i' at event point $(n + 1)$ and also receives materials from task j then the start time of this unit should be after the time that state s, which was produced by a unit j from a recycling task at event point n, is available.

$$T_{sjn} \le T_{j'(n+1)}^{\mathrm{s}} + M\left(1-zI_{jj'n}\right) \quad \forall s\in\mathbf{S}^{IN}, j, j'\in\mathbf{J}_s,\ j\ne j', \sum_{i\in(\mathbf{I}_j\cap\mathbf{I}_s^P\cap\mathbf{I}^R)}\rho_{sij}>0, \sum_{i'\in(\mathbf{I}_{j'}\cap\mathbf{I}_s^C)}\rho_{si'j'}<0, n<N \quad (21)$$

If the materials produced by a non-recycling task in a processing unit at event point n is not

transferred to a consumption task in a processing unit at the same event point *n*, then all material should be stored in its dedicated storage tank, before another production task is processed in the unit. The start time of this consumption task at event point $(n+1)$ should always exceed the time that the state is available at event point *n*.

$$T_{sjn} \le T^{\mathrm{s}}_{j'(n+1)} + M\left(1 - \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}^{C}_{s})} \sum_{(n+1) \le n' \le (n+1)+\Delta n} w_{i'j'(n+1)n'}\right)$$

$$\forall s \in \mathbf{S}^{IN}, j, j' \in \mathbf{J}_s,\ j \ne j',\ \sum_{i \in (\mathbf{I}_j \cap \mathbf{I}^{P}_{s}) \setminus \mathbf{I}^{R}} \rho_{sij} > 0,\ \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}^{C}_{s})} \rho_{si'j'} < 0,\ n < N \tag{22}$$

In other words, a unit that processes a consumption task at event point (*n* + 1) are unconditionally sequenced with the units that process a related non-recycling production task at event point *n*. The units that are processing a consumption task at event point (*n* + 2) are unconditionally sequenced with units that process a related recycling production task at event point *n*.

$$T_{sjn} \le T^{\mathrm{s}}_{j'(n+2)} + M\left(1 - \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}^{C}_{s})} \sum_{n+2 \le n' \le n+2+\Delta n} w_{i'j'(n+2)n'}\right)$$

$$\forall s \in \mathbf{S}^{IN}, j, j' \in \mathbf{J}_s,\ j \ne j',\ \sum_{i \in (\mathbf{I}_j \cap \mathbf{I}^{P}_{s} \cap \mathbf{I}^{R})} \rho_{sij} > 0,\ \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}^{C}_{s})} \rho_{si'j'} < 0,\ n < N\text{–}1 \tag{23}$$

If there is a direct material transfer at event point *n* from a unit *j* that processes a non-recycling production task *i*, to a unit *j*′ that processes a related consumption task *i*′, then the finish time of the unit *j*′ at the previous event point (*n* – 1) must be before the finish time of the unit *j*.

$$T^{\mathrm{f}}_{j'(n-1)} \le T^{\mathrm{f}}_{jn} + M\left(1 - zD_{jj'n}\right)$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}), j, j' \in \mathbf{J}_s,\ j \ne j',\ \sum_{i \in (\mathbf{I}_j \cap \mathbf{I}^{P}_{s}) \setminus \mathbf{I}^{R}} \rho_{sij} > 0,\ \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}^{C}_{s})} \rho_{si'j'} < 0,\ n > 1 \tag{24}$$

If there is a material direct transfer at event point *n* from unit *j*, that process a recycling production task *i*, to unit *j*′, that process a related consumption task *i*′, then the finish time of unit *j*′ at event point *n* must be before the finish time of unit *j*.

$$T^{\mathrm{f}}_{j'n} \le T^{\mathrm{f}}_{jn} + M\left(1 - zD_{jj'(n+1)}\right)$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}), j, j' \in \mathbf{J}_s,\ j \ne j',\ \sum_{i \in (\mathbf{I}_j \cap \mathbf{I}^{P}_{s} \cap \mathbf{I}^{R})} \rho_{sij} > 0,\ \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}^{C}_{s})} \rho_{si'j'} < 0,\ n < N \tag{25}$$

Finally, in order to avoid real time violations, between production and consumption tasks occurring at the same event for recycling tasks or at the previous event for non-recycling tasks the following constraints are introduced.

$$T^{\mathrm{f}}_{jn} \ge T^{\mathrm{s}}_{j'(n-1)} - M\left(1 - \sum_{n-\Delta n \le n' \le n} w_{ijn'n}\right)$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}), j, j' \in \mathbf{J}_s,\ j \neq j',\ \sum_{i \in (\mathbf{I}_j \cap \mathbf{I}_s^P) \setminus \mathbf{I}^R} \rho_{sij} > 0,\ \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C)} \rho_{si'j'} < 0,\ n > 1 \tag{26}$$

$$T_{jn}^{\mathrm{f}} \geq T_{j'n}^{\mathrm{s}} - M\left(1 - \sum_{n-\Delta n \leq n' \leq n} w_{ijn'n}\right)$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}), j, j' \in \mathbf{J}_s,\ j \neq j',\ \sum_{i \in (\mathbf{I}_j \cap \mathbf{I}_s^P \cap \mathbf{I}^R)} \rho_{sij} > 0,\ \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C)} \rho_{si'j'} < 0,\ n \tag{27}$$

4.1.7 Allowing processing units to store materials

In this work, we allow processing units to store materials for multiple event points. Generally, most existing mathematical models even though they allow processing units to store materials, they only allow these materials to be stored at the event point that they were produced. At the next event point, these materials should be either consumed by another task or transferred to the storage tanks. To avoid this case, we introduce an additional binary variable $ys_{i,j,n}$ as follows,

$$ys_{ijn} = \begin{cases} 1 & \text{if unit } j \text{ stores materials at event point } n, \text{ previously produced by task } i \\ 0 & \text{otherwise} \end{cases}$$

We also introduce a new continuous variable $bs_{i,j,n}$ which denotes the amount of materials stored in a unit j at event point n, previously produced by task i in this unit. The amount of materials stored in a unit j cannot exceed its maximum capacity.

$$bs_{ijn} \leq B_{ij}^{\max} \cdot ys_{ijn} \qquad \forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}), j \in \mathbf{J}_s, i \in (\mathbf{I}_j \cap \mathbf{I}_s^P), n \tag{28}$$

Additionally, the amount of materials stored in a unit j at event point n cannot exceed the amount produced at the previous event point (n–1).

$$bs_{ijn} \leq \sum_{n-1-\Delta n \leq n' \leq n} \left(\rho_{sij} \cdot b_{ijn'(n-1)}\right) + bs_{ij(n-1)} \qquad \forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}), j \in \mathbf{J}_s, i \in (\mathbf{I}_j \cap \mathbf{I}_s^P), n > 1 \tag{29}$$

Materials stored in a processing unit can only be directly transferred to another unit that process a consumption task. Constraint (30) is used if materials are produced by non-recycling tasks, while constraint (31) is used if materials are produced by recycling tasks.

$$bs_{ijn} \geq bs_{ij(n-1)} - \sum_{j' \in \mathbf{J}_s} \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C)} bTd_{iji'j'n} \qquad \forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}), j \in \mathbf{J}_s, i \in (\mathbf{I}_j \cap \mathbf{I}_s^P) \setminus \mathbf{I}^R, n > 1 \tag{30}$$

$$bs_{ijn} \geq bs_{ij(n-1)} - \sum_{j' \in \mathbf{J}_s} \sum_{i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C)} bTd_{iji'j'(n+1)} \qquad \forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}), j \in \mathbf{J}_s, i \in (\mathbf{I}_j \cap \mathbf{I}_s^P \cap \mathbf{I}^R), 1 < n < N \tag{31}$$

Finally, if a unit j holds some material at event point n, then it cannot process any task at this event point n.

$$\sum_{i \in \mathbf{I}_j} ys_{ijn} \leq 1 - \sum_{i \in (\mathbf{I}_j \cap \mathbf{I}_s^P)} \sum_{n-\Delta n \leq n' \leq n} \sum_{n \leq n'' \leq n' + \Delta n} w_{ijn'n''} \qquad \forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}), j \in \mathbf{J}_s, n \tag{32}$$

4.1.8 Additional constraints

A number of additional constraints are introduced so as to improve the performance of the proposed

model. Constraints (33)-(36) relate $w_{ijnn'}$ with $zI_{jj'n}$. More specifically, if a unit j' process a consumption task i', and there is indirect material transfer between units j and j' then unit j must process the related production task i according to (33). Similarly, if a unit j processes a production task i, and there is indirect material transfer between units j and j' then unit j' must process the related consumption task i' according to (34). While (33) and (34) are used for non-recycling production tasks, constraints (35) and (36) are used for recycling production tasks.

$$\sum_{n-\Delta n\le n'\le n} w_{ijn'n} \ge \sum_{n\le n'\le n+\Delta n} w_{i'j'nn'} + zI_{jj'n} - 1$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{UIS}),\ j, j' \in \mathbf{J}_s,\ j \ne j',\ i \in (\mathbf{I}_j \cap \mathbf{I}_s^P)\backslash \mathbf{I}^R,\ i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C),\ n \tag{33}$$

$$\sum_{n\le n'\le n+\Delta n} w_{i'j'nn'} \ge \sum_{n-\Delta n\le n'\le n} w_{ijn'n} + zI_{jj'n} - 1$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{UIS}),\ j, j' \in \mathbf{J}_s,\ j \ne j',\ i \in (\mathbf{I}_j \cap \mathbf{I}_s^P)\backslash \mathbf{I}^R,\ i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C),\ n \tag{34}$$

$$\sum_{n-\Delta n\le n'\le n} w_{ijn'n} \ge \sum_{n+1\le n'\le n+1+\Delta n} w_{i'j'(n+1)n'} + zI_{jj'(n+1)} - 1$$

$$\forall\ s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{UIS}),\ j, j' \in \mathbf{J}_s,\ j \ne j',\ i \in (\mathbf{I}_j \cap \mathbf{I}_s^P \cap \mathbf{I}^R),\ i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C),\ n < N \tag{35}$$

$$\sum_{n+1\le n'\le n+1+\Delta n} w_{i'j'(n+1)n'} \ge \sum_{n-\Delta n\le n'\le n} w_{ijn'n} + zI_{jj'(n+1)} - 1$$

$$\forall\ s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{UIS}),\ j, j' \in \mathbf{J}_s,\ j \ne j',\ i \in (\mathbf{I}_j \cap \mathbf{I}_s^P \cap \mathbf{I}^R),\ i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C),\ n < N \tag{36}$$

If an intermediate state s has a FIS policy, then a unit j that transfers materials at unit j', then unit j can either process a production task or store materials at event point n. Constraints (37) and (38) handle cases with non-recycling production tasks, while (39) and (40) handle cases with recycling tasks.

$$\sum_{n-\Delta n\le n'\le n} w_{ijn'n} + ys_{ijn} \ge \sum_{n\le n'\le n+\Delta n} w_{i'j'nn'} + zI_{jj'n} - 1$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}),\ j, j' \in \mathbf{J}_s,\ j \ne j',\ i \in (\mathbf{I}_j \cap \mathbf{I}_s^P)\backslash \mathbf{I}^R,\ i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C),\ n \tag{37}$$

$$\sum_{n\le n'\le n+\Delta n} w_{i'j'nn'} \ge \sum_{n-\Delta n\le n'\le n} w_{ijn'n} + ys_{ijn} + zI_{jj'n} - 1$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}),\ j, j' \in \mathbf{J}_s,\ j \ne j',\ i \in (\mathbf{I}_j \cap \mathbf{I}_s^P)\backslash \mathbf{I}^R,\ i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C),\ n \tag{38}$$

$$\sum_{n-\Delta n\le n'\le n} w_{ijn'n} + ys_{ijn} \ge \sum_{n+1\le n'\le n+1+\Delta n} w_{i'j'(n+1)n'} + zi_{jj'(n+1)} - 1$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}),\ j, j' \in \mathbf{J}_s,\ j \ne j',\ i \in (\mathbf{I}_j \cap \mathbf{I}_s^P \cap \mathbf{I}^R),\ i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C),\ n < N \tag{39}$$

$$\sum_{n+1\le n'\le n+1+\Delta n} w_{i'j'(n+1)n'} \ge \sum_{n-\Delta n\le n'\le n} w_{ijn'n} + ys_{ijn} + zI_{jj'(n+1)} - 1$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}),\ j, j' \in \mathbf{J}_s,\ j \ne j',\ i \in (\mathbf{I}_j \cap \mathbf{I}_s^P \cap \mathbf{I}^R),\ i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C),\ n < N \tag{40}$$

In the same manner we relate $w_{ijnn'}$ and ys_{ijn} with $zD_{jj'n}$.

$$\sum_{n-\Delta n\le n'\le n} w_{ijn'n} + ys_{ijn} \ge \sum_{n\le n'\le n+\Delta n} w_{i'j'nn'} + zD_{jj'n} - 1$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}),\ j, j' \in \mathbf{J}_s,\ j \neq j',\ i \in (\mathbf{I}_j \cap \mathbf{I}_s^P) \backslash \mathbf{I}^R,\ i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C),\ n \tag{41}$$

$$\sum_{n \le n' \le n+\Delta n} w_{i'j'nn'} \ge \sum_{n-\Delta n \le n' \le n} w_{ijn'n} + ys_{ijn} + zD_{jj'n} - 1$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}),\ j, j' \in \mathbf{J}_s,\ j \neq j',\ i \in (\mathbf{I}_j \cap \mathbf{I}_s^P) \backslash \mathbf{I}^R,\ j' \in \mathbf{J}_s,\ i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C),\ n \tag{42}$$

$$\sum_{n-\Delta n \le n' \le n} w_{ijn'n} + ys_{ijn} \ge \sum_{n+1 \le n' \le n+1+\Delta n} w_{i'j'nn'} + zD_{jj'(n+1)} - 1$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}),\ j, j' \in \mathbf{J}_s,\ j \neq j',\ i \in (\mathbf{I}_j \cap \mathbf{I}_s^P \cap \mathbf{I}^R),\ i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C),\ n < N \tag{43}$$

$$\sum_{n+1 \le n' \le n+1+\Delta n} w_{i'j'(n+1)n'} \ge \sum_{n-\Delta n \le n' \le n} w_{ijn'n} + ys_{ijn} + zD_{jj'n} - 1$$

$$\forall s \in (\mathbf{S}^{IN} \cap \mathbf{S}^{FIS}),\ j, j' \in \mathbf{J}_s,\ j \neq j',\ i \in (\mathbf{I}_j \cap \mathbf{I}_s^P \cap \mathbf{I}^R),\ i' \in (\mathbf{I}_{j'} \cap \mathbf{I}_s^C),\ n < N \tag{44}$$

Objective functions

As already discussed, two objectives have been considered. While constraint (45) is used for maximization of productivity, constraint (46) is used for minimization of makespan.

$$z = \sum_s p_s \sum_{j \in \mathbf{J}_s} \sum_{i \in \left(\mathbf{I}_j \cap \mathbf{I}_S^P\right)} \sum_n \sum_{n \le n' \le n+\Delta n} \rho_{ijs} \cdot b_{ijnn'} \tag{45}$$

$$MS \ge T_{jn}^{\mathrm{f}} \qquad \forall j,\ n = N \tag{46}$$

In the minimization of makespan problem, it should be also ensured that the total demand is satisfied.

$$ST_{s,n''} + \sum_{i \in \left(\mathbf{I}_S^P \cap \mathbf{I}^R\right)} \rho_{i,s} \sum_{n-\Delta n \le n' \le n} b_{i,n',n} \ge D_s \qquad \forall s \in \mathbf{S}^p,\ n'' = N \tag{47}$$

Finally, (48) and (49) denote all the continuous and binary variables of the model respectively

$$b_{ijnn'},\ bs_{ijn},\ bTi_{iji'j'n},\ bTd_{iji'j'n},\ MS,\ ST_{sn},\ T_{sjn},\ T_{jn}^{\mathrm{s}},\ T_{jn}^{\mathrm{f}} \ge 0 \tag{48}$$

$$w_{ijnn'},\ ys_{ijn},\ zD_{jj'n},\ zI_{jj'n} \in \{0, 1\} \tag{49}$$

We complete the mathematical model **M1**, which consists of constraints (1)-(45) and (48-49) for maximization of productivity, and (1)-(44) and (46)-(49) for minimization of makespan. We consider two different variations of this model.

4.2 Model M2

In the mathematical model **M2**, we also use unit-specific event-based approach with timing variables based on units. The main difference from the mathematical model **M1** is that related production and consumption tasks are not allowed to take place at the same event point. Therefore, we use the following material balance constraints instead.

$$ST_{sn} = ST_{s(n-1)} + \sum_{i \in (\mathbf{I}_s^P \cap \mathbf{I}^{\mathrm{R}})} \sum_{j \in (\mathbf{J}_s \cap \mathbf{J}_i)} \sum_{(n-1)-\Delta n \le n' \le (n-1)} \rho_{sij} \cdot b_{ijn'(n-1)} + \sum_{i \in \mathbf{I}_s^C} \sum_{j \in (\mathbf{J}_s \cap \mathbf{J}_i)} \sum_{n \le n' \le n+\Delta n} \rho_{sij} \cdot b_{ijnn'}$$

$$\forall s,\ n > 1 \tag{50}$$

$$ST_{sn} = ST0_s + \sum_{i \in \mathbf{I}_s^C} \sum_{j \in (\mathbf{J}_s \cap \mathbf{J}_i)} \sum_{n \le n' \le n+\Delta n} \rho_{sij} \cdot b_{ijnn'} \qquad \forall s,\ n = 1 \tag{51}$$

The mathematical model **M2** consists of constraints (1)-(2), (5)-(7), (8)-(13), (15), (17), (19)-(20),

(22)-(23), (25), (27)-(29), (31)-(32), (35)-(36), (39)-(40), (43)-(45), (48)-(49) and (50)-(51) for maximization of productivity and (1)-(2), (5)-(7), (8)-(13), (15), (17), (19)-(20), (22)-(23), (25), (27)-(29), (31)-(32), (35)-(36), (39)-(40), (43)-(44) and (46)-(47), (48)-(49), (50)-(51) for minimization of makespan.

5 Computational studies

To examine the performance of the proposed mathematical models **M1** and **M2**, we revisit the motivating example 1 and solve additional three motivating examples. The maximum computational time is one hour for all examples. The optimality gap is set to zero. All examples are solved using CPLEX 12/GAMS 24.6.1. on a desktop computer with Intel® Core™ i5-2500 3.3 GHz and 8 GB RAM running Windows 7.

Table 3 Computational results for motivating examples 1-3

Motivating Example	Model	Event Points	CPU Time (s)	RMILP (cu)	MILP (cu)	Binary Variables	Continuous Variables	Constraints
1	**LF2010**[a]	3	0.11	500.00	300.00	6	29	41
(H = 8 h)	**VS2013**[b]	3	0.09	500.00	300.00	14	33	72
	MH2019[c]	4 (ΔR=1)	0.08	500.00	300.00	6	34	78
	M1	2	0.02	500.00	500.00	12	28	60
	M2	3	0.03	500.00	500.00	17	39	86
2	**LF2010**	7	5.4	3281.50	**2385.32**	56	235	518
(H=12 h)	**VS2013**	7	29.5	3281.50	**2392.46**	256	403	1268
	MH2019	9 (ΔR=2)	30.8	3332.63	**2385.32**	120	361	962
	M1	7	39.2	3281.50	**2433.16**	218	463	1442
	M2	7	38.0	3281.50	**2433.16**	216	459	1430
3	**LF2010**	9 (Δn=1)	58.4	3879.34	**887.68**	187	507	1727
(H=12 h)	**VS2013**	9	5.6	3879.34	**989.03**	541	723	2549
	MH2019	9 (ΔR=2)	0.2	887.68	**887.68**	165	506	1424
	M1	9	10.4	3879.34	**1033.60**	453	813	2935
	M2	9	10.5	3879.84	**1033.60**	453	813	2935
4	**LF2010**	-	-	-	-	-	-	-
(H=12 h)	**VS2013**	-	-	-	-	-	-	-
	MH2019	-	-	-	-	-	-	-
	M1	9	27.9	4297.11	2503.15	453	813	2935
	M2	9	27.3	4297.11	2503.15	453	813	2935

[a] Li and Floudas[12] model. [b] Vooradi and Shaik[15] model. [c] Mostafaei and Harjunkoski[21] model

Revisit of Motivating Example 1

We use the proposed models **M1** and **M2** to solve the motivating example 1. The optimal solution of 500.00 cu is generated in less than 0.1 CPU s for both models. The model statistics is provided in Table 3. It involves 12 binary variables, 28 continuous variables, and 60 constraints for model **M1** and 17 binary variables, 39 continuous variables, and 86 constraints for model **M2**. The optimal schedule is the same as that is illustrated in Figure 4. As discussed before, intermediate state S2 is held in unit J2 after production because of small storage capacity.

As illustrated in Table 3, it is able to generate the optimum solution for the motivating example using the proposed models **M1** and **M2** as both of them allow production units to store materials over multiple event points. As already discussed, even though the model of Vooradi and Shaik[15] allows materials to be temporarily stored in the processing unit during an event point *n*, these materials cannot be stored to the processing unit for the next event points. Similarly, Li and Floudas[12] and Mostafaei and Harjunkoski[21] does not allow materials to be stored in the processing unit for the next event points. As a result, both proposed models **M1** and **M2** can generate a significantly better solution.

Motivating Example 2

This example is very similar to Example 2c from Li *et al.*[16] but with the maximum capacity of state S7 being changed 10 mu. The objective is to maximize productivity. Similar to the Motivating Example 1, we use the model of Li and Floudas[12], Vooradi and Shaik[15], Mostafaei and Harjunkoski[21] and the proposed model **M1** and **M2** to solve this motivating example. The computational results are provided in Table 3. From Table 3, it can be seen that both proposed mathematical models **M1** and **M2** are able to generate a solution of 2433.16 mu, whilst the model of Li and Floudas[12], Vooradi and Shaik[15] and Mostafaei and Harjunkoski[21] are only able to generate a suboptimum solution (2392.46 mu and 2385.32 mu respectively). This is mainly due to the fact that the proposed models **M1** and **M2** allow production units to storage materials over multiple event points. The optimal schedule from model M1 is illustrated in Figure 7. As seen from Figure 7, unit J2 produces 50 mu from 6.3h to 8.9h by processing task I2 at event point N4. Those materials can be stored in the processing unit J2 and processed in the same unit at event point N7. However, this is not possible with the model of Vooradi and Shaik[15] and as a result less materials can be produced during the same period as depicted in Figure 8 (2.80 mu by processing task I2 at event point N4 and 39.03 mu by processing task I2 at event point N6). This leads to less productivity and as a result a suboptimum solution.

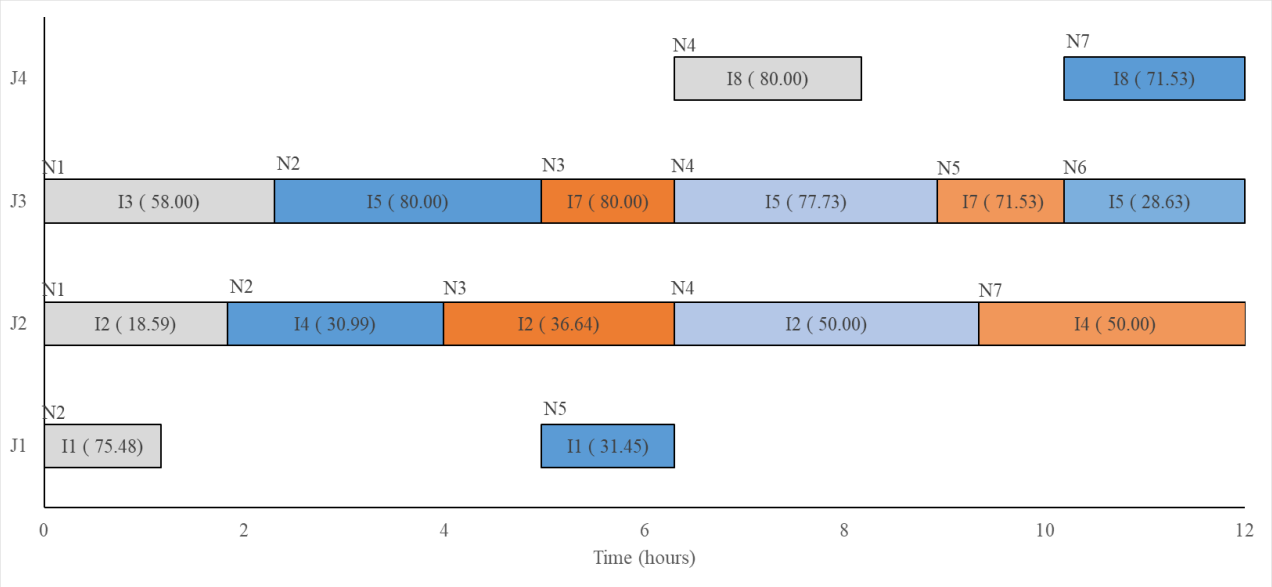

Figure 7 Optimal schedule for Motivating Example 2 using model M1

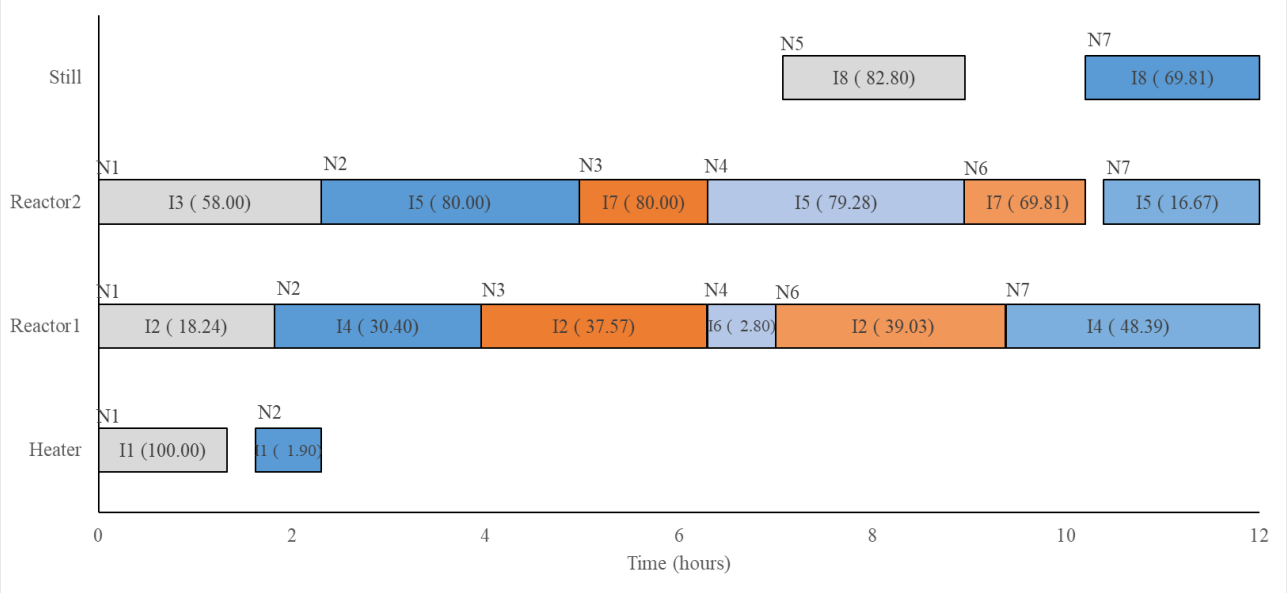

Figure 8 Schedule for Motivating Example 2 using the model of Vooradi and Shaik[15]

Motivating Example 3

Motivating example 3 is quite similar to the Example 3c from Li *et al.*[16]. The maximum capacity for states S5, S6 and S7 is changed 10 mu. Additionally, the initial amount of materials for states S6 and S7 is changed 0 mu. Similar to Motivating Example 2, we use the model of Li and Floudas[12], Vooradi and Shaik[15], Mostafaei and Harjunkoski[21] and the proposed model **M1** and **M2** to solve this motivating example. The computational results are provided in Table 3. From Table 3, both proposed models **M1** and **M2** are able to generate a better solution than the models from the literature (1033.60 cu). This can be explained by examining the optimal schedule generated by using model **M1** (see Figure 9) and the model of Vooradi and Shaik[15] (see Figure 10). Since there is small storage capacity

for states S6 and S7, unit J4 is able to process batches with small sizes when the model of Vooradi and Shaik[15] is used. More specifically, I9 is processed in unit J4 in event points N3, N5 and N7 with batch sizes of 20.00 mu, 38.40 mu and 53.73 mu respectively. On the other hand, model **M1** can produce significantly larger amounts of states S6 and S7, since those excessive amounts can be temporarily stored in the processing units, before transferred to another unit. For instance, with model **M1** unit J4 also processes three batches of I9 at event points N3, N6 and N8 with batch sizes 25.00 mu, 43.00 mu and 90.00 mu respectively.

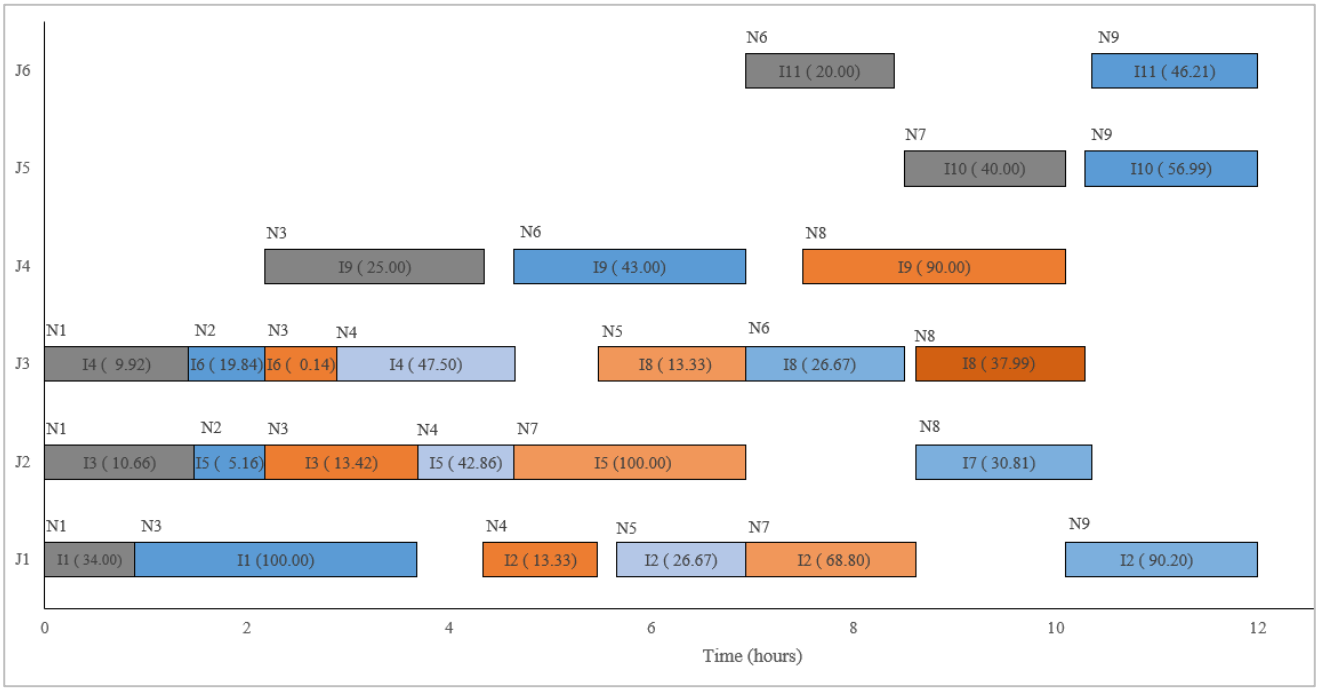

Figure 9 Optimal schedule for Motivating Example 3 using model M1

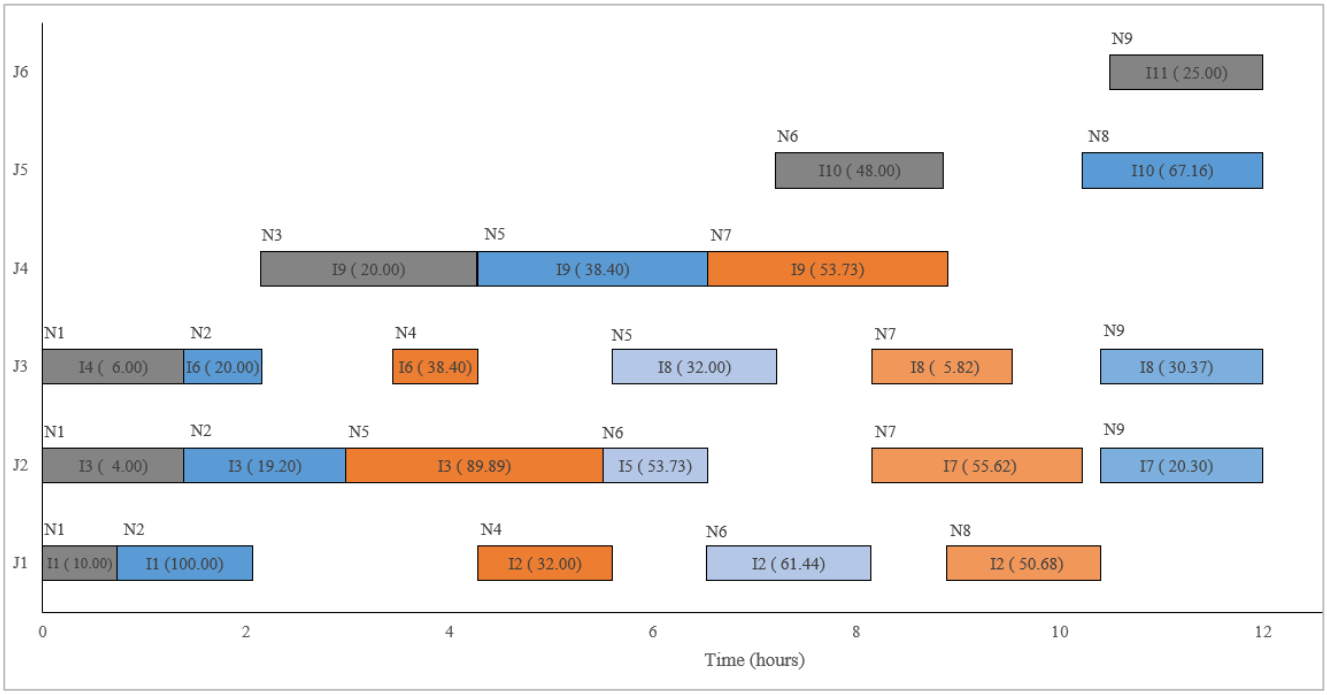

Figure 10 Optimal schedule for Motivating Example 3 using the model of Vooradi and Shaik[15]

Motivating Example 4

This example is also quite similar to the Example 3c from Li *et al.*[16]. The maximum capacity for state S7 is 10 mu. Additionally, it is assumed that in the first event point 40 mu of S7 are stored in unit J4 at the first event point. Since the models of Vooradi and Shaik[15] and Mostafaei and Harjunkoski[21] do not allow materials to be stored for multiple event points, they fail to generate a feasible solution. On the other hand, the proposed models **M1** and **M2** allows materials to be stored in processing units for multiple event points and as a result they are able to generate the optimum solution of 2503.15 mu in less than 30 s.

Benchmark Examples

To further examine the performance of the proposed mathematical models **M1** and **M2**, we solve in total 9 examples from the literature[1, 8, 16]. The data as well as the STN representations for all examples are presented in the **Supplementary Material**. The maximum computational time is one hour for all examples. The optimality gap is set to zero. All examples are solved using CPLEX 12/GAMS 24.6.1. on a desktop computer with Intel® Core™ i5-2500 3.3 GHz and 8 GB RAM running Windows 7. It should also be noted that we only compare our models with the model of Vooradi and Shaik[15] (denoted as **VS2013**) since they incorporate similar features. The model of Mostafaei and Harjunkoski[21] is very similar to the model of Shaik and Floudas[11] which requires more event points in some examples as demonstrated in the Motivating Example 1 and in Vooradi and Shaik[15]. Detailed comparison of our models with Shaik and Floudas[11], Li and Floudas[12], Susarla et al.[8], and Mostafaei and Harjunkoski[21] will be presented in our next contribution in order to reduce the length.

The computational results for Examples 1-9 with UIS policy for maximization of productivity, are presented in Tables 4 and 5. From Tables 4 and 5, it seems that both the model of Vooradi and Shaik[15] and the model **M2** require the same number of event points to generate the optimal solution. This is because both models do not allow related production and consumption tasks to take place at the same event point. Nevertheless, it seems that model **M2** requires fewer binary variables in some cases. For instance, in Example 3d, the model of Vooradi and Shaik[15] requires 263 binary variables, while model **M2** requires 245 binary variables. This is because **M2** only examines if there is a material transfer between processing units, whist the model of Vooradi and Shaik[15] examines if there is a material transfer from a production task to a related consumption task. In a multipurpose batch process facility, a processing unit can process more than one tasks. Therefore, two processing units can process two or more tasks which are related to the same state. In such a case, the proposed model **M2** only requires one binary decision variable, while the model of Vooradi and Shaik[15] requires two or more binary decision variables. As a result, the proposed model **M2** can lead to a smaller model size and less computational time. For instance, **M2** requires 36% less computational time for Example 2d (15.1 s vs 23.6 s), 87.7% less computational time for Example 3b (146.4 s vs 1191 s) and 62.2%

less computational time for Example 3d (41.7 s vs 110.3 s) than the model of Vooradi and Shaik[15]. Furthermore, additional constraints (33)-(44) can improve the performance of the proposed models. For instance, both models **M2** and the model of Vooradi and Shaik[15] lead to the same model size for Example 1d. However, the model **M2** requires 51.8% less computational time to generate the optimal solution (10.5s vs 21.8 s).

Table 4 Computational results for Examples 1-3 with maximization of productivity (UIS policy)

Example	Model	Event points	CPU Time (s)	RMILP (cu)	MILP (cu)	Binary Variables	Continuous Variables	Constraints
Ex1a	**VS2013**	4	**0.125**	2000.00	1840.17	32	90	177
(H = 8 h)	**M1**	2	**0.031**	2000.00	1840.17	18	54	105
	M2	4	**0.031**	2000.00	1840.17	32	102	198
Ex1b	**VS2013**	5	**0.125**	3000.00	2628.19	41	113	226
(H = 10h)	**M1**	3	**0.046**	3000.00	2628.19	27	80	163
	M2	5	**0.047**	3000.00	2628.19	41	128	256
Ex1c	**VS2013**	6	**0.250**	4000.00	3463.62	50	136	275
(H = 12h)	**M1**	4	**0.062**	4000.00	3463.62	36	106	221
	M2	6	**0.109**	4000.00	3463.62	50	154	314
Ex1d	**VS2013**	9	21.8	6601.65	5038.05	77	205	422
(H = 16h)	**M1**	7	12.9	6601.65	5038.05	63	184	395
	M2	9	10.5	6601.65	5038.05	77	232	488
Ex2a	**VS2013**	4	**0.125**	1730.87	1498.57	62	178	384
(H = 8 h)	**M1**	4	**0.063**	1730.87	1498.57	64	180	396
	M2	4	**0.078**	1730.87	1498.57	56	178	377
Ex2b	**VS2013**	5	0.17	2436.69	1962.69	80	225	496
(H = 10h)	**M1**	5	0.22	2436.69	1962.69	80	227	510
	M2	5	0.20	2436.69	1962.68	72	225	491
Ex2c	**VS2013**	6	0.48	3076.62	2658.52	98	272	608
(H = 12h)	**M1**	6	0.42	3076.62	2658.52	96	274	624
	M2	6	0.42	3076.62	2658.52	88	272	605
Ex2d	**VS2013**	8	23.6	4291.67	3738.38	134	366	832
(H = 16h)	**M1**	8	15.6	4291.67	3738.38	128	368	852
	M2	8	15.1	4291.67	3738.38	120	366	833
Ex3a	**VS2013**	5	**1.92**	2100.00	1583.44	123	311	741
(H = 8h)	**M1**	5	**0.90**	2100.00	1583.44	130	316	793

	M2	5	**0.86**	2100.00	1583.44	115	316	776
Ex3b	**VS2013**	7	**1191**	3369.69	2358.20	179	441	1077
(H = 10h)	**M1**	7	**146.4**	3369.69	2358.20	182	448	1155
	M2	7	**157.0**	3369.69	2358.20	167	448	1138
Ex3c	**VS2013**	7	1.31	3465.63	3041.27	179	441	1077
(H = 12h)	**M1**	7	1.22	3465.63	3041.27	182	448	1155
	M2	7	1.14	3465.63	3041.27	167	448	1138
Ex3d	**VS2013**	10	**110.3**	5225.86	4262.80	263	636	1581
(H = 16h)	**M1**	10	**42.8**	5225.86	4262.80	260	646	1698
	M2	10	**41.7**	5225.86	4262.80	245	646	1681

Note. $\Delta n = 0$ for all examples. VS2013: Vooradi and Shaik[15] model.

Table 5 Computational results for Examples 4-9 with maximization of productivity (UIS policy)

Example	Model	Event Points	CPU Time (s)	RMILP (cu)	MILP (cu)	Binary Variables	Continuous Variables	Constraints
Ex4	**VS2013**	6	**0.124**	7.5000	5.3225	65	157	358
(H=15 h)	**M1**	4	**0.078**	7.5000	5.3225	48	127	298
	M2	6	**0.078**	7.5000	5.3225	65	187	434
Ex5	**VS2013**	5	**0.109**	14.00	10.00	36	98	201
(H=6 h)	**M1**	3	**0.032**	14.00	10.00	24	71	152
	M2	5	**0.031**	14.00	10.00	36	113	237
Ex6	**VS2013**	5	**0.141**	300.00	210.00	49	138	283
(H=9 h)	**M1**	3	**0.047**	300.00	210.00	33	100	211
	M2	5	**0.031**	300.00	210.00	49	158	327
Ex7	**VS2013**	5	**0.125**	80.00	58.99	54	147	301
(H=76 h)	**M1**	2	**0.031**	80.00	58.99	24	71	145
	M2	5	**0.046**	80.00	58.99	54	167	351
Ex8	**VS2013**	6	**0.093**	400.00	400.00	44	130	289
(H=10 h)	**M1**	4	**0.032**	400.00	400.00	32	106	237
	M2	6	**0.047**	400.00	400.00	44	154	337
Ex9	**VS2013**	10	**0.109**	400.00	400.00	76	218	497
(H=10 h)	**M1**	8	**0.062**	400.00	400.00	64	210	485
	M2	10	**0.032**	400.00	400.00	76	258	585

Note. $\Delta n = 0$ for all examples. VS2013: Vooradi and Shaik[15] model.

Mathematical model **M1** requires a smaller number of event points in most cases since related production and consumption tasks are allowed to take place at the same event point. For instance, the model **M1** requires two event points less than the models **M2** and the model of Vooradi and Shaik[15]for Examples 1a-d, 8 and 9. As a result, model **M1** leads to the smallest model size with less number of binary variables, continuous variables and constraints, which makes it more efficient than the mathematical model Vooradi and Shaik[15]. Nevertheless, it seems that both mathematical models **M1** and **M2** require similar computational time to generate the optimal solution, mainly because they both models can solve all examples in less than three minutes. From Tables 4 and 5, it can be concluded that the models **M1** and **M2** reduced the computational time by one order of magnitude for most examples in comparison to **VS2013**.

Tables 6 and 7 present the computational results for Examples 1-10 with FIS policy for maximization of productivity. Both mathematical models **M2** and the model of Vooradi and Shaik[15] require the same number of event points for all examples to generate the optimal solution. As we introduce additional binary variables to allow processing units to store materials for multiple event points, model **M2** leads to a big larger model size for Examples 1a-1d, 4, 5, and 7 where the model of Vooradi and Shaik[15] does not need to allow tasks to span over multiple event points (i.e., $\Delta n = 0$) to generate the optimal solution. However, model **M2** requires similar computational time as the model of Vooradi and Shaik[15] for these examples. On the other hand, model **M2** requires 15.5%-16.5% fewer binary variables for Examples 2a-2d, 3a-3d due to fact that the proposed model only uses binary variables to examine whether there is a material transfer between two units. More importantly, both models **M2** and **M1** do not require to allow tasks to span over multiple event points in any case due to allowing processing units to store materials over multiple event points. Therefore, both proposed models lead to significantly smaller model size with less binary and continuous variables and constraints in Examples 6, 8 and 9. For instance, both models **M2** and **M1** require 61.9% (128 vs 336) and 53.5% (156 vs 336) less binary variables than the model of Vooradi and Shaik[15] to generate the optimal solution for Example 9 respectively. Such reduction in the model size leads to one magnitude less computational time required for both proposed models **M1** and **M2** in comparison to the model of Vooradi and Shaik[15].

Table 6 Computational results for Examples 1-3 with maximization of productivity (FIS policy)

Example	Model	Event Points	CPU Time (s)	RMILP (cu)	MILP (cu)	Binary Variables	Continuous Variables	Constraints
Ex1a	**VS2013**	4	0.094	2000.00	1840.17	64	102	273
(H = 8 h)	**M1**	2	0.031	2000.00	1840.17	38	72	175
	M2	4	0.047	2000.00	1840.17	74	126	358
Ex1b	**VS2013**	5	**0.234**	3000.00	2628.19	81	129	352
(H = 10h)	**M1**	3	**0.046**	3000.00	2628.19	58	107	279
	M2	5	**0.062**	3000.00	2628.19	94	159	462
Ex1c	**VS2013**	6	0.23	4000.00	3463.62	98	156	431
(H = 12h)	**M1**	4	0.17	4000.00	3463.62	78	142	383
	M2	6	0.25	4000.00	3463.62	114	192	566
Ex1d	**VS2013**	9	45.3	6601.65	5038.05	149	240	668
(H = 16h)	**M1**	7	40.3	6601.65	5038.05	138	247	695
	M2	9	43.5	6601.65	5038.05	174	291	878
Ex2a	**VS2013**	4	**0.125**	1730.87	1498.57	142	220	665
(H = 8 h)	**M1**	4	**0.078**	1730.87	1498.57	122	256	774
	M2	4	**0.078**	1730.87	1498.57	120	252	755
Ex2b	**VS2013**	5	0.45	2436.69	1962.69	180	281	866
(H = 10h)	**M1**	5	0.30	2436.69	1962.69	154	325	999
	M2	5	0.20	2436.69	1962.69	152	321	980
Ex2c	**VS2013**	6	0.66	3076.62	2658.52	218	342	1067
(H = 12h)	**M1**	6	0.50	3076.62	2658.52	186	394	1224
	M2	6	0.47	3076.62	2658.52	184	390	1205
Ex2d	**VS2013**	8	34.6	4291.67	3738.38	294	464	1469
(H = 16h)	**M1**	8	22.2	4291.67	3738.38	250	532	1674
	M2	8	23.7	4291.67	3738.38	248	528	1655
Ex3a	**VS2013**	5	3.32	2100.00	1583.44	293	387	1321
(H = 8h)	**M1**	5	1.80	2100.00	1583.44	245	437	1542
	M2	5	1.92	2100.00	1583.44	245	437	1531
Ex3b	**VS2013**	7	**976.3**	3369.69	2358.20	417	555	1935
(H = 10h)	**M1**	7	**383.8**	3369.69	2358.20	349	625	2233
	M2	7	**364.9**	3369.69	2358.20	349	625	2233
Ex3c	**VS2013**	7	2.90	3465.63	3041.27	417	555	1935

(H = 12h)	**M1**	7	1.47	3465.63	3041.27	349	625	2244
	M2	7	1.42	3465.63	3041.27	349	625	2233
Ex3d	**VS2013**	10	**155.6**	5225.86	4262.80	603	807	2856
(H = 16h)	**M1**	10	**85.5**	5225.86	4262.80	505	907	3297
	M2	10	**82.2**	5225.86	4262.80	505	907	3286

$\Delta n = 0$ for all examples. VS2013: Vooradi and Shaik[15] model.

Table 7 Computational results for Examples 4-10 with maximization of productivity (FIS policy)

Example	Model	Event Points	CPU Time (s)	RMILP (cu)	MILP (cu)	Binary Variables	Continuous Variables	Constraints
Ex4	**VS2013**	6 (Δn=0)	0.312	7.5000	5.3225	149	341	618
(H=15 h)	**M1**	4 (Δn=0)	0.218	7.5000	5.3225	105	168	560
	M2	6 (Δn=0)	0.172	7.5000	5.3225	157	246	845
Ex5	**VS2013**	5 (Δn=0)	**0.141**	14.00	10.00	76	114	327
(H=6 h)	**M1**	3 (Δn=0)	**0.062**	14.00	10.00	52	92	265
	M2	5 (Δn=0)	**0.047**	14.00	10.00	84	144	438
Ex6	**VS2013**	5 (Δn=1)	**0.265**	300.00	210.00	144	182	547
(H=9 h)	**M1**	3 (Δn=0)	**0.078**	300.00	210.00	78	130	380
	M2	5 (Δn=0)	**0.078**	300.00	210.00	128	202	636
Ex7	**VS2013**	5 (Δn=0)	**0.125**	80.00	58.99	114	171	490
(H=76 h)	**M1**	2 (Δn=0)	**0.047**	80.00	58.99	50	91	243
	M2	5 (Δn=0)	**0.062**	80.00	58.99	122	211	638
Ex8	**VS2013**	6 (Δn=3)	**0.343**	400.00	400.00	152	198	665
(H=10 h)	**M1**	4 (Δn=0)	**0.047**	400.00	400.00	64	134	409
	M2	6 (Δn=0)	**0.062**	400.00	400.00	92	192	597
Ex9	**VS2013**	10 (Δn=7)	**2.10**	400.00	400.00	336	422	1453
(H=10 h)	**M1**	8 (Δn=0)	**0.23**	400.00	400.00	128	266	849
	M2	10 (Δn=0)	**0.11**	400.00	400.00	156	324	1037

VS2013: Vooradi and Shaik[15] model.

The computational results for examples using minimization of makespan as objective are presented Tables 8 and 9. While Table 8 presents the results with UIS policy, Table 9 gives the results with FIS policy. From Table 8, it seems that mathematical models **M1** and **M2** both lead to tighter MILP relaxation and smaller model sizes. For instance, the MILP relaxation from both M1 and M2 are 18.68 h for Example 2a, which is improved by 73.2% compared to 10.78 from the model of Vooradi and

Shaik[15]. The number of binary variables is reduced from 152 to 138 by 9%. As a result, they can successfully solve all examples except Example 2b to global within one hour. On the other hand, the model of Vooradi and Shaik[15] can only solve for Examples 2a, 3a and 3b to optimality, whilst both models **M1** and **M2** require similar or less computational time to solve Examples 2a, 3a and 3b to optimality. The maximum reduction in the computational time can reach 43% for Example 2a (174 vs. 99 and 174 vs. 106). By comparing models **M1** and **M2** in Table 8, it seems that allowing related production and consumption tasks at the same event point can also lead to less computational times. For instance, model **M1** requires 34.8% for Example 1a (412 s vs 639 s) and 49.2% (1004 s vs 1978 s) less computational time for Example 1b compared to model **M2**.

Table 8 Computational results for Examples 1-3 with minimization of makespan (UIS policy)

Example	Model	Event points	CPU Time (s)	RMILP (h)	MILP (h)	Binary Variables	Continuous Variables	Constraints
Ex1a	**VS2013**	14	**>3600[a]**	24.24	27.88	122	320	672
(D_{S4}=2000 cu)	**M1**	12	**412**	24.24	27.88	108	314	690
	M2	14	**639**	24.24	27.88	122	362	783
Ex1b	**VS2013**	23	**>3600[b]**	48.47	52.07	203	527	1113
(D_{S4}=4000 cu)	**M1**	21	**1004**	48.47	52.07	189	548	1212
	M2	23	**1978**	48.47	52.07	203	596	1305
Ex2a	**VS2013**	9	173.4	10.78	19.34	152	413	953
(D_{S8}=200 cu)	**M1**	9	99.6	18.68	19.34	138	415	963
(D_{S9}=200 cu)	**M2**	9	106.1	18.68	19.34	136	413	952
Ex2b	**VS2013**	20	>3600[c]	26.12	46.11	350	930	2185
(D_{S8}=500 cu)	**M1**	20	>3600[d]	45.57	46.11	312	930	2206
(D_{S9}=400 cu)	**M2**	20	>3600[e]	45.57	46.11	314	932	2217
Ex3a	**VS2013**	7	0.578	10.00	13.37	179	441	1089
(D_{S12}=100 cu)	**M1**	7	0.546	11.25	13.37	167	448	1145
(D_{S13}=200 cu)	**M2**	7	0.702	11.25	13.37	167	448	1145
Ex3b	**VS2013**	10	0.889	12.50	17.02	263	636	1593
(D_{S12}=250 cu)	**M1**	10	0.873	14.27	17.02	245	646	1688
(D_{S13}=250 cu)	**M2**	10	0.874	14.27	17.02	245	646	1688

Note that $\Delta n = 0$ in all cases. [a] Relative Gap 0.19%. [b] Relative Gap 0.01%. [c] Relative Gap 17.3%. [d] Relative Gap 1.17% [e] Relative Gap 1.17%. VS2013: Vooradi and Shaik[15] model.

From Table 9, we can observe that models **M1** and **M2** lead to tighter MILP relaxation and

smaller model size. For instance, the MILP relaxation from both **M1** and **M2** are 45.57 h for Example 2b, which is improved by 73% compared to 26.40 from the model of Vooradi and Shaik[15] . The number of binary variables is reduced by 16.2% (788 vs. 660). As a result, the models **M1** and **M2** can solve Examples 2a, 3a and 3b to optimality within 1 hour and solve Examples 1a, 1b, and 2b with smaller optimality gap within 1 hour compared to the model of Vooradi and Shaik[15]. It should also be noted that models **M1** and **M2** find a better solution of 52.07 within 1 hour compared to the model of Vooradi and Shaik[15] (52.07 vs. 52.23), which has not been found in the literature. By comparing models **M1** and **M2** in Table 9**,** it seems that allowing related production and consumption tasks at the same event point can also lead to less computational times. For instance, model **M1** requires 12.5% (125 s vs 143 s) less computational time for Example 2a. In brief, we can conclude that the mathematical model **M1** is the most efficient for makespan minimization.

Table 9 Computational results for Examples 1-3 with minimization of makespan (FIS policy)

Example	Model	Event points	CPU time (s)	RMILP (h)	MILP (h)	Binary Variables	Continuous Variables	Constraints
Ex1a	**VS2013**	14	>3600[a]	24.24	27.88	234	372	1068
(D_{S4}=2000 cu)	**M1**	12	>3600[b]	24.24	27.88	214	398	1196
	M2	14	>3600[c]	24.24	27.88	242	456	1371
Ex1b	**VS2013**	23	>3600[d]	48.47	**52.23**	387	615	1779
(D_{S4}=4000 cu)	**M1**	21	>3600[e]	48.47	**52.07**	376	695	2114
	M2	23	>3600[f]	48.47	**52.07**	404	753	2289
Ex2a	**VS2013**	9	241.7	10.78	19.34	332	525	1679
(D_{S8}=200 cu)	**M1**	9	125.3	18.68	19.34	276	601	1889
(D_{S9}=200 cu)	**M2**	9	142.7	18.68	19.34	272	597	1875
Ex2b	**VS2013**	21	>3600[g]	26.40	47.68	788	1257	4091
(D_{S8}=500 cu)	**M1**	21	>3600[h]	45.57	47.68	660	1429	4589
(D_{S9}=400 cu)	**M2**	21	>3600[i]	45.57	47.68	656	1425	4575
Ex3a	**VS2013**	7	0.780	10.00	13.37	417	555	1947
(D_{S12}=100 cu)	**M1**	7	1.841	11.25	13.37	320	625	2209
(D_{S13}=200 cu)	**M2**	7	1.311	11.25	13.37	320	625	2209
Ex3b	**VS2013**	10	1.545	12.50	17.02	603	807	2868
(D_{S12}=250 cu)	**M1**	10	1.092	14.27	17.02	470	907	3256
(D_{S13}=250 cu)	**M2**	10	1.513	14.27	17.02	470	907	3256

Note $\Delta n = 0$ in all cases. [a] Relative Gap 1.75%. [b] Relative Gap 1.40%. [c] Relative Gap 1.67%. [d] Relative Gap 0.39%. [e] Relative Gap 0.15%. [f] Relative Gap 0.08%. [g] Relative Gap 19.4%. [h] Relative Gap 0.64%. [i] Relative Gap 1.07%. VS2013: Vooradi and Shaik[15] model.

Large-scale example

We also solve a large-scale industrial batch plant example from Janak et al.[25] to further illustrate the capabilities of models **M1** as model **M1** performs slightly better than **M2** based on the above computational results. Figure 11 depicts the STN representation of this batch plant. The facility produces 87 different products by processing 17 raw materials in 8 different processing paths. There is a total of 6 different types of processing tasks. 20 different processing units are available to process these tasks. Each processing unit can only process one type of tasks. The batch plant has to fulfil 402 orders within 19 days. For more information, it can be referred to Janak et al.[25].

We first use the proposed model **M1** to solve this problem directly. It fails to generate a feasible schedule within 12 hours due to intractable problem size. We then employ he rolling-horizon decomposition approach of Janak et al.[25] with the proposed model **M1** as the short-term scheduling model to solve this problem, which are denoted as **RH-M1**. The level-1 model and the modified short-term scheduling model **M1** are provided in the **Supplementary Material**. Each subproblem is solved to zero optimality gap using CPLEX 12/GAMS 24.6.1. on a desktop computer with Intel® Core™ i5-2500 3.3 GHz and 8 GB RAM running Windows 7. The maximum computational time is set as 3 hours for each level. An integer solution limit of 40 is also imposed.

The computational results are provided in Table 10. From Table 10, **RH-M1** can generate a better solution with productivity of 6880.2 mu, which is increased by 26.7% in comparison to that of 5427.8 mu from the model of Janak et al.[25]. More interestingly, **RH-M1** requires 11.5 h to generate such an improved solution, which is approximately half of the CPU time required by the model of Janak et al.[25] (22.4 h). Since the same rolling horizon decomposition approach is used in both cases, such improvement solely derives from the improved efficiency of the short-term model.

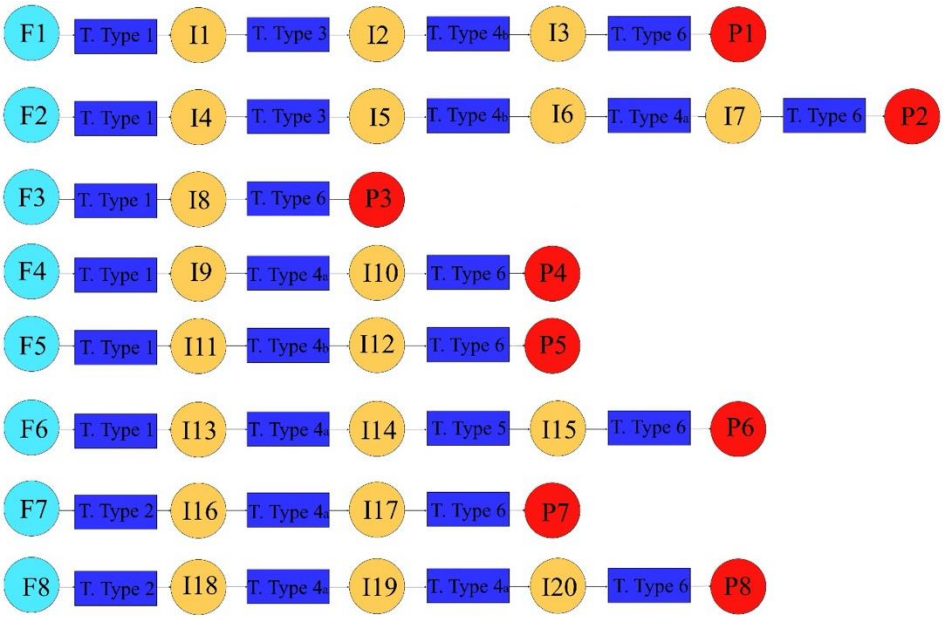

Figure 11 STN representation of large-scale industrial plant example

Table 10 Computational results for the industrial plant example

Model	Total production (mu)	Total CPU time (h)
RH-JF	5427.8	22.4
RH-M1	6880.2	11.5

Table 11 Computational results for each subproblem for industrial plant example

Sub-problem	Model	Days	Production (mu)	CPU time (s)	Binary Variables	Continuous Variables	Constraints
1	JF	0-2	857.7	3315	4880	35384	187833
	M1	0-2	853.5	1972	15465	60213	127010
2	JF	3-4	758.5	7202	3834	27053	135916
	M1	3-4	790.1	10200	11021	42382	94200
3	JF	5-6	697.0	9878	5406	30545	248663
	M1-J	5-6	777.3	3899	15388	48876	160812
4	JF	7-8	788.8	10329	5526	30729	276612
	M1-J	7-8	994.9	1355	15781	48915	172265
5	JF	9-10	634.7	6945	5406	30465	271764
	M1-J	9-10	853.5	706	15855	49310	172930
6	JF	11-12	517.9	10800	6222	32280	354410
	M1-J	11-12	779.0	10800	17323	53395	198433
7	JF	13-14	532.3	10800	6252	32318	359365
	M1-J	13-14	1114.6	1541	17228	53642	199952
8	JF	15-16	315.7	10800	6156	32085	354649
	M1-J	15-18	717.3	10800	28614	90455	350605
9	JF	17-18	335.3	10800	5976	31664	344065

The computational results for each subproblem from **RH-M1** and **RH-JF** are depicted in Table 11. While **RH-JF** divides the entire scheduling problem into 9 subproblems, **RH-M1** divides into 8 subproblems. **RH-M1** can solve all subproblems except the subproblems 6 and 8 to optimality within 3 hours. However, **RH-JF** reaches the maximum time of 3 hours for 4 subproblems out of 9. **RH-M1** leads to higher productivity in comparison to **RH-JF** for all subproblems except the subproblem 1. The difference in productivity for the subproblem 1 between **RH-M1** and **RH-JF** is 0.5% only. Since processing units overproduce some materials in **RH-M1**, which do not fulfil any order at the current scheduling horizon, they can be stored and used for order delivery directly at a later sub-problem

without the need of using the facility to produce. As a result, processing units require to process fewer tasks in the successive sub-problems. Consequently, **RH-M1** can successfully generate the schedule of subproblem 8, which contains days 15-18 without the need of further dividing into smaller sub-problems. On the other hand, **RH-JF** needs to produce significantly more materials to fulfil the demand within days 15-18. Therefore, **RH-JF** divides this sub-horizon into sub-problem 8 with days 15-16, and sub-problem 9 with days 17-18 to successfully develop a schedule for this period.

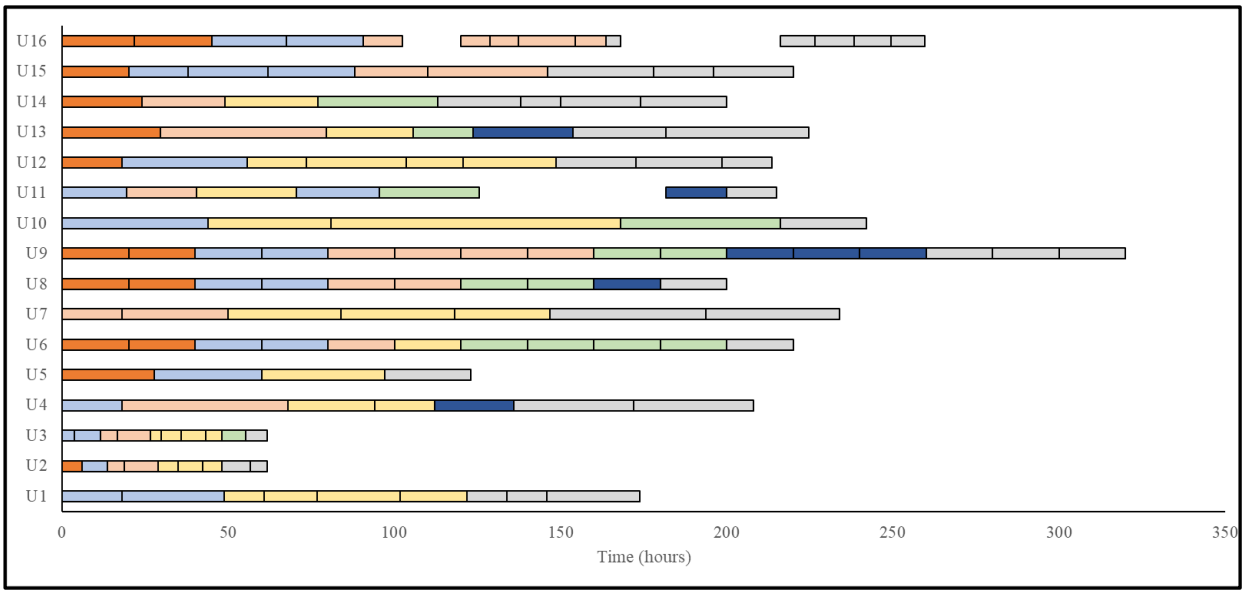

Figure 12 Optimal schedule for the large-scale industrial plant example using **RH-M1**

Table 12 Utilisation efficiency of processing units from RH-M1 and RH-JF

	RH-M1			RH-JF		
	Time used	Time left	% utilised	Time used	Time left	% utilised
U1	173.8	282.2	40.2	187.6	268.4	41.1
U2	61.8	394.2	14.3	92.4	363.6	20.3
U3	61.6	394.4	14.3	118.0	338.0	25.9
U4	208.0	248.0	48.1	200.0	256.0	43.9
U5	123.0	333.0	28.5	49.6	406.4	10.9
U6	123.0	333.0	28.5	296.2	159.8	65.0
U7	233.8	222.2	54.1	174.1	281.9	38.2
U8	200.0	256.0	46.3	233.0	223.0	51.1
U9	340.0	116.0	78.7	311.4	144.6	68.3
U10	242.0	214.0	56.0	170.4	285.6	37.4
U11	158.9	297.1	36.8	90.0	366.0	19.7
U12	213.6	242.4	49.4	129.2	326.8	28.3
U13	224.6	231.4	52.0	185.5	270.5	40.7
U14	200.0	256.0	46.3	162.0	294.0	35.5
U15	220.0	236.0	50.9	129.7	326.3	28.4
U16	194.0	262.0	44.9	451.9	4.1	99.1
U17	-	-	-	12.0	444.0	2.6

The feasible schedule from RH-M1 is illustrated in Figure 12. Table 12 depicts the utilization efficiency for all processing units for both models. **RH-M1** utilizes most of the processing unit for larger periods in order to produce a larger amount of materials and fulfill more orders than **RH-JF**. Additionally, **RH-M1** utilizes one processing unit less during the whole scheduling horizon. In other words, **RH-M1** utilizes the processing units more efficiently.

6 Conclusions

In this work, we presented two generic unit-specific event-based models for scheduling of multipurpose batch processes using the unit-specific event-based modelling approach. While we followed the approach of Rakovitis et al.[17] to allow all related production and consumption tasks to take place at the same event points but in different real times in the first model, we did not allow in the second model. We introduced the concept of indirect and direct material transfer, which allows us to conditionally align the operational sequence of related production and consumption tasks. The processing units were allowed to hold materials that are previously produced over multiple event points. Nonsimultaneous material transfer[8] was also allowed in both models. The computational results demonstrated that both models require a smaller number of binary variables in most cases, especially in the cases where a processing unit can process multiple tasks, compared to the existing mathematical formulation[15]. The proposed models did not need to allow a task to span over multiple event points in order to generate the optimal solution. As a result, the computational time was significantly reduced by one order of magnitude in most cases. More importantly, the proposed models were able to generate better solutions than Vooradi and Shaik[15] and Mostafaei and Harjunkoski[21]. In addition, the first model allowing related production and consumption tasks to take place at the same event points was slightly more efficient than the second one. Finally, we used the proposed model to solve a large-scale industrial batch plant scheduling problem from Janak *et al.*[25] using the rolling-horizon decomposition algorithm. The results demonstrated that the proposed model is able to improve the productivity by 26.7% in significantly less computational time compared to that from Janak et al.[25]. The future work will extend the proposed models to consider other immediately storage policy and wait policy. Detailed comparison with all existing models in the literature especially the model of Mostafaei and Harjunkoski[21] will also be conducted.

Acknowledgments

Nikolaos Rakovitis would like to acknowledge financial support from the postgraduate award by The University of Manchester.

Literature Cited

1. Kondili E, Pantelides CC, Sargent RWH. A general algorithm for short-term scheduling of batch operations-I MILP formulation. *Comput Chem Eng.* 1993;17(2):211-227. https://doi.org/10.1016/0098-1354(93)80015-F.

2. Pantelides C. Unified frameworks for optimal process planning and scheduling. *Proceedings of the Second Conference on Foundations of Computer Aided Operations*. 1994:253-274.

3. Lee H. Maravelias CT. Discrete-time mixed integer programming models for short-term scheduling in multipurpose environments. *Comput Chem Eng.* 2017;107:171-183. https://doi.org/10.1016/j.compchemeng.2017.06.013.

4. Zhang X. Sargent RWH. 1996. The optimal operation of mixed production facilities-A general formulation and some approaches for the solution. *Comput Chem Eng.* 1996;20(6-7):897-904. https://doi.org/10.1016/0098-1354(95)00186-7.

5. Castro P. Barbosa-Póvoa APFD. Matos H. An improved RTN continuous-time formulation for the short-term scheduling of multipurpose batch plants. *Ind Eng Chem Res*. 2001;40(9):2059-2068. https://doi.org/10.1021/ie000683r.

6. Maravelias CT. Grossmann IE. New General Continuous-Time State-Task Network Formulation for Short-Term Scheduling of Multipurpose Batch Plants. *Ind Eng Chem Res*. 2003;42(13):3056-3074. https://doi.org/10.1021/ie020923y.

7. Sundaramoorthy A, Karimi IA. A simpler better slot-based continuous-time formulation for short-term scheduling in multipurpose batch plants. *Chem Eng Sci*. 2005;60(10):2679-2702. https://doi.org/10.1016/j.ces.2004.12.023.

8. Susarla N, Li J, Karimi I. A. Novel approach to scheduling multipurpose batch plants using unit-slots. *AIChE J*. 2010;56(7):1859-1879. https://doi.org/10.1002/aic.12120.

9. Li J. Karimi IA. Scheduling gasoline blending operations from recipe determination to shipping using unit slots. *Ind Eng Chem Res*. 2011;50(15):9156-9174. https://doi.org/10.1021/acs.iecr.6b01930.

10. Ierapetritou MG. Floudas CA. Effective continuous-time formulation for short-term scheduling. 1. Multipurpose batch processes. *Ind Eng Chem Res.* 1998;37(11):4341-4359. https://doi.org/10.1021/ie970927g.

11. Shaik MA. Floudas CA. Novel Unified Modeling Approach for Short-Term Scheduling. *Ind Eng Chem Res.* 2009;48(6):2947-2964. https://doi.org/10.1021/ie8010726.

12. Li J. Floudas CA. Optimal Event Point Determination for Short-Term Scheduling of Multipurpose Batch Plants via Unit-Specific Event-Based Continuous-Time Approaches, *Ind Eng Chem Res.* 2010;49(16):7446-7469. https://doi.org/10.1021/ie901842k.

13. Tang QH. Li J. Floudas CA. et al. Optimization framework for process scheduling of operation-dependent automobile assembly lines. *Optim Lett*. 2012;6(4):797-824.

https://doi.org/10.1007/s11590-011-0303-5.

14. Seid R. Majozi T. A robust mathematical formulation for multipurpose batch plants. *Chem Eng Sci*. 2012;68(1):36-53. https://doi.org/10.1016/j.ces.2011.08.050.

15. Vooradi R. Shaik MA. Rigorous unit-specific event-based model for short term scheduling of batch plants using conditional sequencing and unit-wait times. *Ind Eng Chem Res*. 2013;52(36):12950-12792. https://doi.org/ 10.1021/ie303294k.

16. Li J. Xiao X. Floudas CA. Integrated gasoline blending and order delivery operations: Part I. short-term scheduling and global optimization for single and multi-period operations, *AIChE J*. 2016;62(6):2043-2070. https://doi.org/10.1002/aic.15168.

17. Rakovitis N. Zhang N. Li J. Zhang L. A new approach for scheduling of multipurpose batch processes with unlimited intermediate storage policy. *Front Chem Sci Eng*. 2019;13:784-802. https://doi.org/10.1007/s11705-019-1858-4.

18. Méndez CA. Cerdá J. Optimal scheduling of a resource-constrained multiproduct batch plant supplying intermediates to nearby end-product facilities. *Comput Chem Eng*. 2000;24(2-7):369-376. https://doi.org/10.1016/S0098-1354(00)00482-8.

19. Hui C. Gupta A. van der Meulen HAJ. A novel MILP formulation for short-term scheduling of multi-stage multi-product batch plants with sequence-dependent constraints. *Comput Chem Eng*. 2000;24(12):2705 – 2717. https://doi.org/10.1016/S0098-1354(00)00623-2.

20. Méndez CA. Cerdá J. An MILP continuous-time framework for short-term scheduling of multipurpose batch processes under different operation strategies. *Optim Eng*. 2003;4(1-2):7-22. https://doi.org/10.1023/A:1021856229236.

21. Mostafaei H, Harjunkoski I. Continuous-time scheduling formulation for multipurpose batch plants. AIChE J. 2020;66:e16804. https://doi.org/10.1002/aic.16804.

22. Floudas CA. Lin X. Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review. *Comput Chem Eng*. 2004;28(11):2109-2129. https://doi.org/10.1016/j.compchemeng.2004.05.002.

23. Méndez CA. Cerdá, J. Grossmann IE. Harjukoski I. Fahl M. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Comput Chem Eng*. 2006;30(6-7):913-946. https://doi.org/10.1016/j.compchemeng.2006.02.008.

24. Harjunkoski I. Maravelias CT. Bongers P. et al. Scope for industrial application of production scheduling models and solution methods. *Comput Chem Eng*. 2014;62(5):161-193. https://doi.org/10.1016/j.compchemeng.2013.12.001.

25. Janak SL. Floudas CA. Kallrath J. Vormbrock N. Production scheduling of a large-scale industrial batch plant. I. Short-term and Medium-term scheduling. *Ind Eng Chem Res*. 2006;45(25):8234-8252. https://doi.org/10.1021/ie0600588.