Batch Scheduling with Quality-based Changeovers

Braulio Brunaud[1], Satyajith Amaran[2], Scott Bury[2], John Wassick[2], and Ignacio E. Grossmann[1]

[1]Carnegie Mellon University, Department of Chemical Engineering
[2]The Dow Chemical Company

November 30, 2018

Abstract

Batch scheduling is a frequent and complex operation performed in all process industries. Because of its importance, general modeling frameworks have been developed to optimize multipurpose plants. Three of the main modeling frameworks are compared and extended to include quality-based changeovers (QBC), a feature present in the chemical industry in which the cleaning operation can be avoided given that enough batches of the second product are performed in a row. The frameworks considered are: State-Task-Network (STN), Resource-Task-Network (RTN), and Unit-Operation-Port-State-Superstructure (UOPSS). A case study implemented in the three frameworks allows to compare their main features, including computational efficiency and extensibility to accommodate novel features such as quality based changeovers. The results indicate the STN formulation is limited for large-scale problems, the RTN is difficult to extend, while the UOPSS is very intuitive and has a good performance for large scale problems. Sequence-dependent changeovers are effectively managed with the use of lazy constraints, and the inclusion of QBC does not add computational burden to the STN and UOPSS models. This is the first comparison performed for these scheduling frameworks and is meant to serve as a guideline to select the right framework for the desired application.

1 Introduction

Short-term batch scheduling has been one of the key drivers of Enterprise-wide Optimization (Grossmann, 2012). Due to the high frequency in which these decisions are made, even small improvements can translate into millions of dollars in savings. The high frequency also allows to quickly test decision tools in the operation, facilitating the adoption of new technologies. The advances in modeling and solution techniques of scheduling optimization problems is an incentive to move away from manual or spreadsheet-aided scheduling. In Process Systems Engineering (Sargent, 2005), the timing and sizing of batches

must be determined for multipurpose plants. Optimization models have been proposed for a variety of applications in the chemical, pharmaceutical, petroleum, and consumer goods industries (Harjunkoski et al., 2014).

The importance of this problem in the chemical industry has motivated the development of standardized modeling frameworks that can accommodate a variety of requirements. Kondili et al. (1993) introduced the State-Task-Network (STN), representing the transformation of raw materials into products, both referred as states, through the use of tasks. Pantelides (1994) proposed the Resource-Task-Network (RTN), which extends the STN framework to account for other resources such as energy, people, and the physical units. More recently, Zyngier and Kelly (2012) proposed the Unit-Operation-Port-State-Superstructure (UOPSS) paradigm, in which the starting point is the Process Flowsheet Diagram (PFD), augmented by an STN-like structure. The units in the PFD can perform several tasks that have inlet and outlet ports for resource flows. A large body of research has been devoted to the development of models based on STN and RTN formulations (Méndez et al., 2006). However, they have not been critically compared to unveil their advantages and disadvantages. This is the first goal of this paper. We restrict ourselves to the case of discrete-time representation. For continuous-time representations the reader is referred to Floudas and Lin (2004).

Most scheduling models with sequence-dependent changeovers (SDC) consider a specific task that must be performed when transitioning from one product to the next one. Cleaning is the most common operation in a changeover task. However, in some applications, especially in the chemical industry, there are different ways of transitioning between products. The second goal of this paper is to introduce a new type of changeover frequently present in chemical plants, the specific case of quality-based changeovers (QBC). When a transition occurs from one product to the next one without cleaning, the first batch of the second product might be contaminated with traces of the first one, generating off-spec product. However, if sufficient batches of the second product are executed in a row, the impurity can be diluted enough to bring back the resulting batches to the quality specifications. We develop formulations that consider this alternative, which also allows to evaluate the extensibility of the standardized modeling frameworks.

Both goals of the paper, the critical comparison of batch scheduling frameworks, and the extension to include quality-based changeovers are relevant contributions to enable the selection of the best scheduling models for multilevel supply-chain optimization (Brunaud and Grossmann, 2017). The paper is organized as follows: Section 2 details the concept of quality-based changeovers. Section 3 describes the general

problem that needs to be solved. The STN, RTN and UOPSS frameworks are described in detail in Section 4. A case study is presented in Section 5 with discussions in Section 6, and the main conclusions of the paper are summarized in Section 7.

2 Quality-based Changeovers

In this section we further describe the concept of quality-based changeovers, in which batches with traces of impurities can be brought to the required quality specification by means of dilution, avoiding time-consuming cleaning operations.

To illustrate the concept consider the following example, in the following referred as Example 1. A single batch reactor can perform two tasks, either produce product A from raw material RawA (reaction RxA), or produce product B from raw material RawB (reaction RxB). To keep things simple, both reactions take only one hour and have stoichiometry 1:1 (Fig. 1)

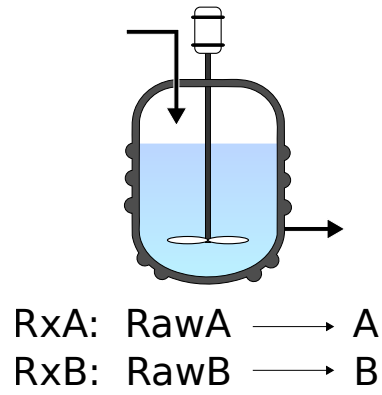

Figure 1. Example 1 reactor

First, a batch of 100 L of A is produced. Next, a batch of 100 L of B is produced without cleaning inbetween, containing a trace of 500 mL of A. To meet the quality requirements, a batch of B can contain at most 2 mL/L of A. If another batch of 100 L of B is produced and blended in the same product tank, the concentration of A (the contaminant) decreases to 2.5 mL/L. After a third batch of 100 L of B is produced the concentration of the contaminant reaches 1.7 mL/L, bringing the product back to specification. The situation in the tank of product B is shown in Fig. 2.

The traditional scheduling frameworks do not include the option to handle the situation described above. This paper contributes to bridge that gap. Schedulers need the alternative of deciding which kind of changeover to execute, whether is cleaning or a QBC. In some cases ensuring several batches in a row

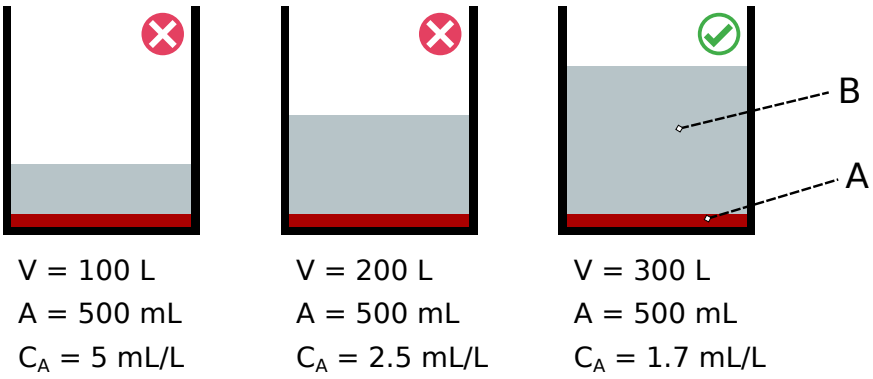

Figure 2. Filling of the tank of product B

for the next product will be the best choice, whereas in others doing a cleaning operation might be better decision, the later case will be more frequent when the demand of the next product is low.

3 Problem Description

The goal of this section is to gain a deeper understanding of scheduling modeling frameworks. Their modeling capabilities will be assessed with their extension to include quality-based changeovers.

Given a set of process equipment, their interconnections, the tasks that each can perform, the recipes for each task, and the changeover times between different products, as well as number of successive batches for each product that meet the product specifications by blending of these batches. The problem is then to determine the optimal sequence and size of each batch. The output of the model corresponds to a short-term production plan that can be described by a Gantt chart. The optimal plan depends on the objective chosen, whether to minimize makespan, maximize production, or fulfill certain orders at minimum cost.

All the discussion from the current and following sections is centered on Example 1, because it is small and simple enough for our development. The following additional considerations are required:

1. The objective is to maximize the production of both products
2. The production of product A must be 100 L, equal to the capacity of the reactor
3. The planning horizon is 4 hours
4. All processes take 1 hour, and use the same stoichiometry 1:1

5. Changeover from product B to product A is forbidden

6. A changeover from product A to product B requires a cleaning of 1 hour, unless 3 batches of B are produced in a row

When the only changeover option is cleaning, the objective value is 300 L, and the Gantt chart from Fig 3 is obtained.

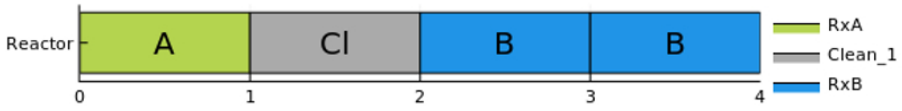

Figure 3. Optimal schedule for changeover with cleaning

When the QBC option of transitioning to 3 batches in a row of B is added, the objective value raises to 400 L and the Gantt chart from Fig 4 is obtained.

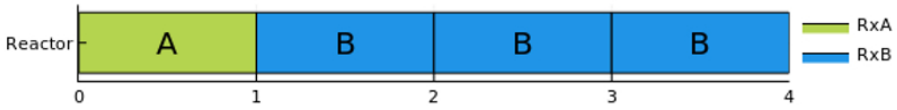

Figure 4. Optimal schedule for changeover with cleaning and QBC

We now explore how to obtain these results in the three scheduling frameworks under study, STN, RTN, and UOPSS. Our goal is to develop optimization formulations capable of selecting between these options accounting for the trade-offs present in the model.

4 Modeling Frameworks for Batch Scheduling

4.1 State-Task-Network

The State-Task-Network is a modeling framework for batch scheduling that is based on the transformation of materials into products (both called states) by the use of tasks. The STN representation for Example 1 is given in Fig. 5.

In the representation, physical plant units are absent. It is purely focused on the processing part of the process. The base variables of the model are:

The minimal STN model with discrete time representation with the objective of maximizing production

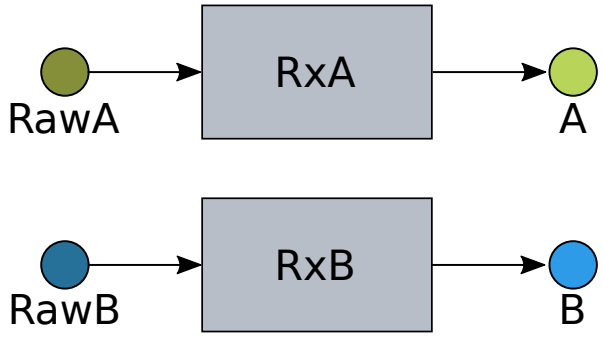

Figure 5. STN representation of the example

w_{ijt}	1 if task j is started at unit i in period t
b_{ijt}	batch size of task j produced at unit i in period t
s_{kt}	inventory of state k at the end of period t

is given by Eqs. 1–6.

$$Max \quad \sum_k \eta_k s_{kT} - \sum_i \sum_j \sum_t PC_{ij} b_{ijt} \tag{1}$$

$$s.t. \quad \sum_{i \in I_j} \sum_{\tau=t-PT_{ij}+1}^{t} w_{ij\tau} \leq 1 \qquad \forall j,t \tag{2}$$

$$s_{kt} = s_{kt-1} + \sum_{i \in I_k^P} \rho_{ik}^p \sum_{j \in J_i} b_{ij(t-PT_{ij})} - \sum_{i \in I_k^C} \rho_{ik}^c b_{ijt} \qquad \forall k,t \tag{3}$$

$$V_{ij}^{min} w_{ijt} \leq b_{ijt} \leq V_{ij}^{max} w_{ijt} \qquad \forall i,j \in J_i,t \tag{4}$$

$$C_k^{min} \leq s_{kt} \leq C_k^{max} \qquad \forall k,t \tag{5}$$

$$b_{ijt}, s_{kt} \geq 0, w_{ijt} \in \{1,0\} \tag{6}$$

Eq. (2) is required to optimize the sequencing of batches and ensure that tasks do not overlap. This constraint is proposed by Shah et al. (1993) as an improvement to the original formulation by Kondili et al. (1993). Eq. (3) is the material balance for each state. Eq. (4) relates the binary variables for batch sequencing with the batch amount variables. It also provides bounds for the batch size.

The objective value considered is to maximize the profit involving the product sales and the production costs. For simplicity, it is assumed that all product is sold at the end of the optimization horizon, thus the amount sold is represented in Eq. 1 as the final inventory s_{kT} in period T. It is trivial to incorporate intermediate withdrawals and purchases of raw materials by adding the corresponding terms to Eq. 3.

Other alternative objective functions are minimizing the number of batches, minimizing changeover costs. The parameter η_k is the price of the product, PC_{ij} is the production cost, and PT_{ij} is the processing time of task j in unit i. The parameters ρ^p_{ik} and ρ^c_{ik} define the proportion of state k produced or consumed by task i. $V^{min/max}_{ij}$ is the minimum/maximum batch size, and $C^{min/max}_k$ is the minimum/maximum inventory of state k.

To model sequence-dependent changeovers Eq (7) is added to the model. We will refer to the model defined by Eqs. (1) – (7) as STN-C.

$$\sum_{t=t_1+1}^{t_2-1} w_{imt} \geq w_{ijt_1} + w_{ilt_2} - \sum_{n} \sum_{t=t_1+1}^{t_2-1} w_{int} - 1 \qquad \forall i,j,l,j \neq l, t_1, t_2 > t_1 \tag{7}$$

The constraint from Eq. (7) indicates that if a task l is started at a time t_2 after task j started at time t_1, then the changeover task m must be started in a period between t_1 and t_2, unless a different task n is started in that period (Fig. 6). Because the constraint is defined for two time indices, it can yield a large number of constraints, $O(T^2)$, where T is the number of time periods. The constraint also leads to undesirable symmetry, because the cleaning task can be performed at any period between t_1 and t_2.

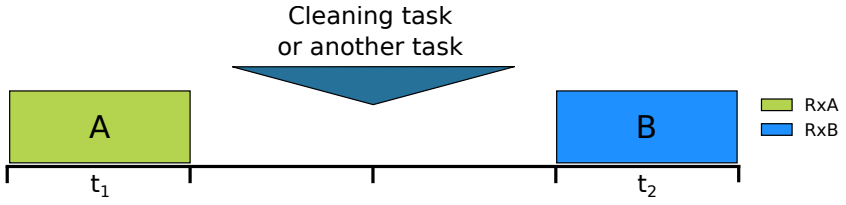

Figure 6. STN sequence-dependent changeovers constraint representation

The constraint from Eq. (7) can be extended to accommodate a QBC. The resulting constraint is defined by Eq. (8).

$$\sum_{m \in M(j,k)} \sum_{t=t_1+1}^{t_2-1} w_{imt} \geq w_{ijt_1} + w_{ikt_2} - \sum_{n} \sum_{t=t_1+1}^{t_2-1} w_{int} - 1 - \sum_{\alpha=1}^{U_{jk}-1} \frac{1}{U_{jk}-1} w_{ik(t_2+\alpha PT_{ik})} \qquad \forall i,j,k,j \neq k, t_1, t_2 > t_1 \tag{8}$$

As before, the constraint indicates that a cleaning task m must be performed to transition from task j to k, unless another task n is performed. However, in Eq. (8) the last two terms allow to override

the cleaning changeover if at least U_{jk} batches of task k are executed, after t_2. We will refer to the model including Eqs. (1) – (6), and (8) as STN-QBC. An example of the application of the constraint is presented in Appendix B.

4.2 Resource-Task-Network

The explicit absence of the processing units in the STN formulation is addressed by the Resource-Task-Network (RTN) (Pantelides, 1994). The equipment and other materials required to conduct the operation are explicitly considered. Together with states, they are given the generic name of resources. In this way, any resource including materials, equipment, utilities, and human resources, can be seamlessly incorporated into the model, requiring a single resource balance constraint. The formulation becomes more general but less intuitive. The RTN representation for Example 1 is given in Fig. 7. The double arrow notation indicates that the task consumes and produces a resource. For example, reaction A consumes a reactor and produces an empty reactor when it finishes.

The main decision variables of the model are:

y_{it} : a binary variable indicating if task i starts at the beginning of period t
b_{it} : the size of the batch processed in period t, executing task i
r_{kt} : inventory of resource k in period t

The simplest discrete time RTN model to maximize production is given by Eqs. 9–12

$$Max \quad \sum_k \eta_k r_{kT} - \sum_i \sum_t PC_i b_{it} \tag{9}$$

$$s.t. \quad r_{kt} = r_{kt-1} + \sum_{i \in I_r} \sum_{tau=0}^{PT_i} (\mu_{ir\tau} y_{i(t-\tau)} + \nu_{ir\tau} b_{i(t-\tau)}) + \Pi_{rt} \qquad \forall k, t \tag{10}$$

$$V_{ik}^{min} y_{it} \leq b_{it} \leq V_{ik}^{max} y_{it} \qquad \forall i, r \in R_i, t \tag{11}$$

$$b_{it}, r_{kt} \geq 0, y_{it} \in \{1, 0\} \tag{12}$$

The first interesting observation about the model is that it does not require a logical constraint to prevent task overlapping as Eq (2) in the STN model. The ability to start a task is controlled by the resource availability. Another observation is that the resource inventory variables might represent integer quantities, such as number of reactors or operators. However, there is no need to declare them as integer, provided that they are consumed in integer quantities. The main advantage of the RTN model is that

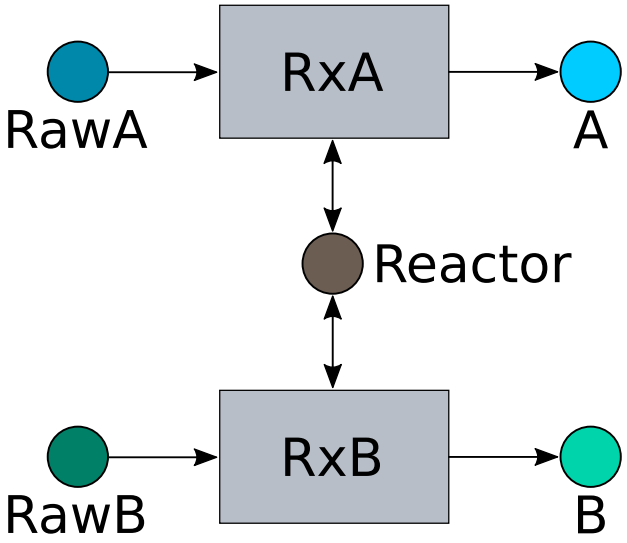

Figure 7. RTN representation of the example

processes with many identical units can be represented with a smaller number of variables. At the same time, this forces to represent most custom logical conditions with auxiliary tasks and resources. For example, sequence-dependent changeovers are modeled with the inclusion of auxiliary resources and tasks to represent the cleaning operations. The example with cleaning changeover is represented in Fig. 8, where Reactor A represents a virtual unit corresponding to a reactor ready to perform reaction A, and Reactor B a virtual unit of a reactor ready to perform reaction B.

The initial inventory of the resource Reactor must be expressed as the constraint in Eq. (13). The model defined by Eqs. (9)–(13) is referred as RTN-C.

$$r_{Reactor\,A,0} + r_{Reactor\,B,0} = 1 \tag{13}$$

The extension to include QBC is more challenging. To represent exactly the same feasible space as the STN-QBC formulation additional virtual units and tasks are required. Fig 9 shows the RTN diagram for Example 1. As for the case of cleaning changeover case, a cleaning task can be used to transform a virtual reactor A into a virtual reactor B. However, in this case there is also the alternative of taking the route through auxiliary tasks RxB1 and RxB2, ensuring that if cleaning is not performed, at least 3 batches of B are executed in a row. To make sure the three tasks are executed in-tandem, the inventory of Reactor B2 must be set to zero.

Even though this approach represents the same feasible space as STN-QBC, it is not a scalable

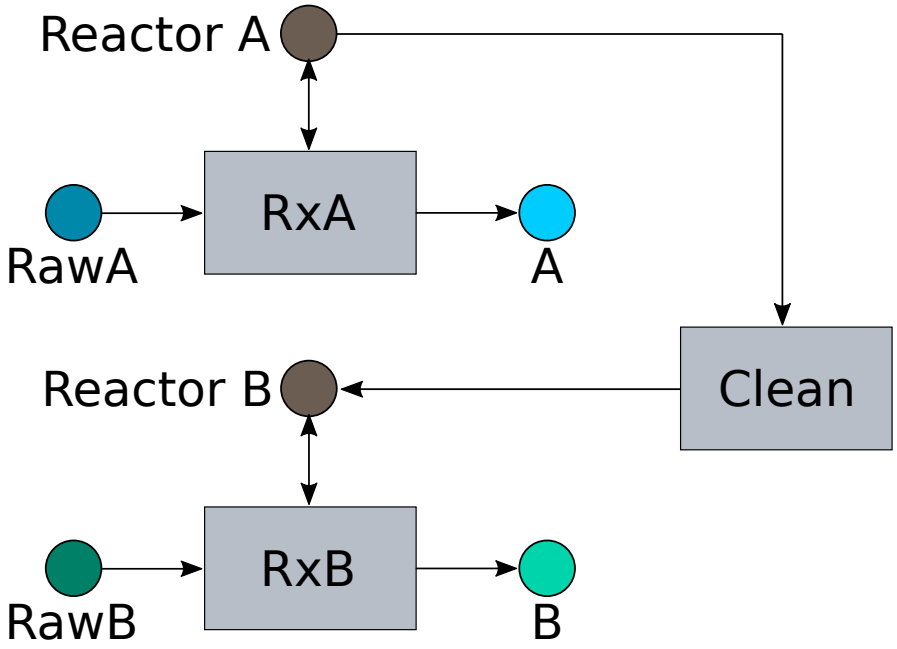

Figure 8. RTN representation of the example with cleaning changeover

modeling strategy as it grows exponentially. For each pair of QBC of length U_{jk}, the number of batches in a row of product k produced to transition from product j to product k, U_{jk} - 2 auxiliary tasks, and U_{jk} - 1 auxiliary resources are required. The RTN model for the QBC case is defined by the same constraints as the base RTN, Eqs. (9) – (12).

4.3 Unit-Operation-Port-State-Superstructure

The Unit Operation Port State Superstructure (UOPSS) (Zyngier and Kelly, 2012) is a scheduling framework focused on representing the physical interconnections in a plant, with the addition of also representing the tasks that a unit can perform. A unit has different operations, which have inlet and outlet ports to allow the filling an unloading of a unit. Connections are accounted for with flow lines between ports of different units. Storage tanks are special units that can perform the storage operation. Multipurpose tanks are also included in the framework. Fig. 10 shows the main components of UOPSS.

The UOPSS representation for the Example 1 with one reactor and two operations is shown in Fig. 11.

The emphasis on representing the physical layout of the plant can be seen in Fig. 11. In the UOPSS representation storage tanks are explicitly included, whereas in STN and RTN it is assumed that states and resources, respectively, are storage points. To consider a zero-wait (ZW) policy the tank capacity is

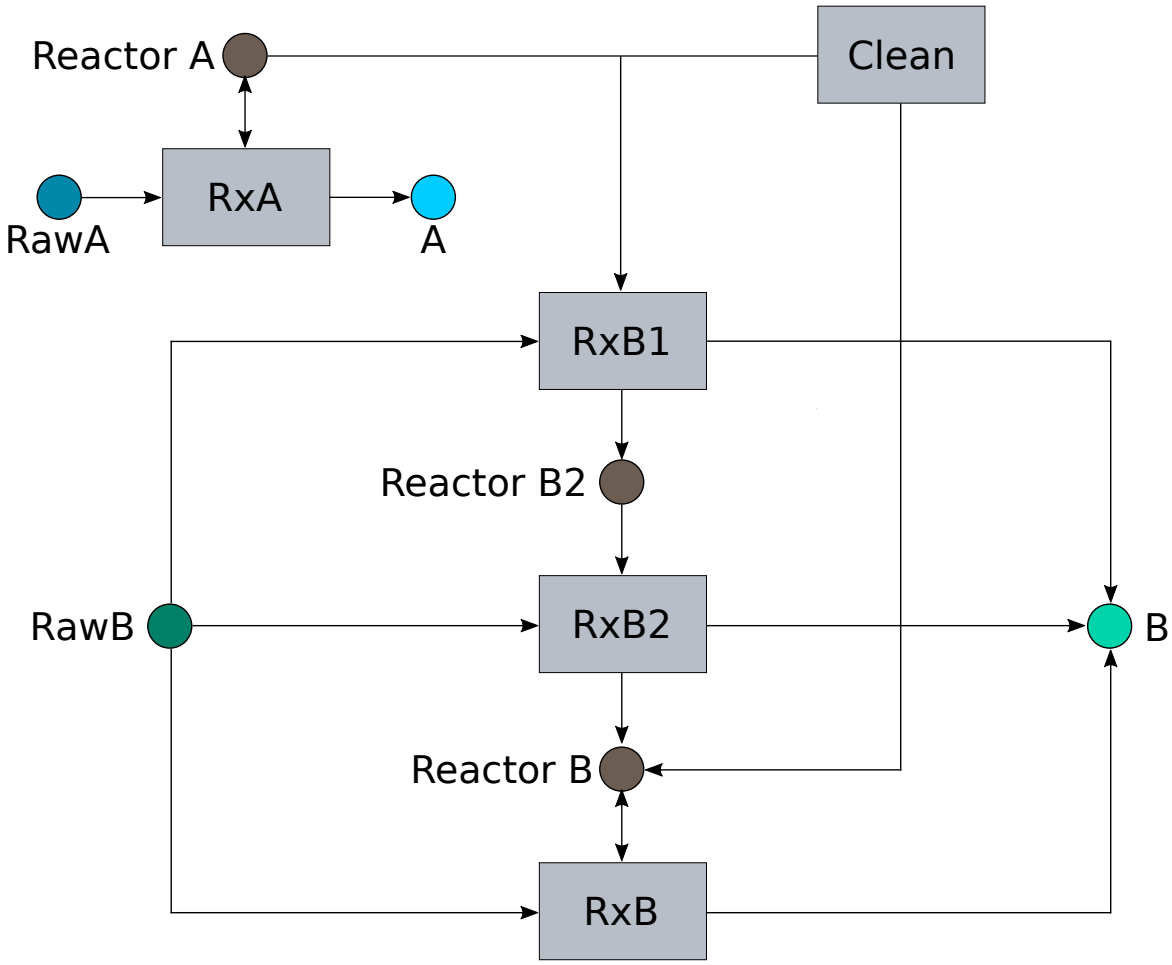

Figure 9. RTN representation of the example with cleaning and QBC

set to zero; for unlimited-intermediate-storage (UIS) policy the tank capacity is set to a large value.

The main decision variables of the model, detailed in Zyngier and Kelly (2009), are the following:

For an application in which production must be maximized, the objective is expressed by Eq. (14).

$$Max \quad \sum_i \sum_s \eta_s inv_{ipT} - \sum_i \sum_j \sum_t PC_{ij} b_{ijt} \tag{14}$$

The constraints can be classified as logical, logistic, and material balances. The first type controls the sequence of the operations, i.e. when tasks are started and stopped. The UOPSS formulation includes a larger number of logical variables that are related by Eqs. (15)–(18), where Eq. (15) ensures that tasks do not overlap, Eq. (16) sets the variable indicating when a task is active, Eq. (17) sets the switch-to-self variable, and Eq. (18) connects the start-up of a tasks with its shutdown. The start-up variable su is is analogous to the variable w in STN. Eqs 19–22 are tightening constraints that allow to declare only one of

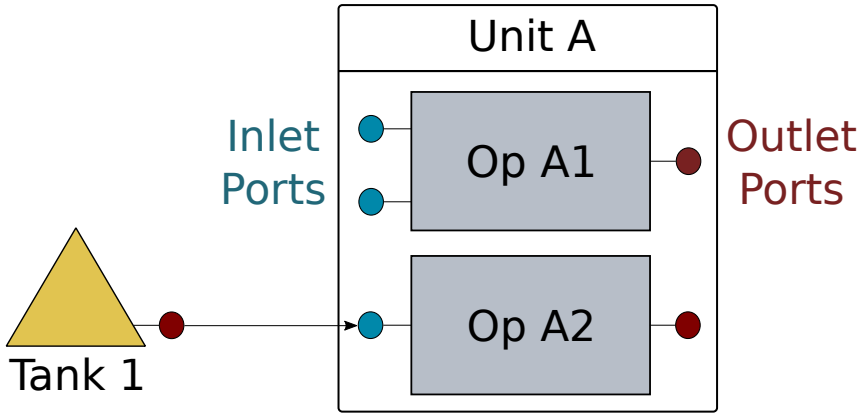

Figure 10. Main components of the UOPSS representation

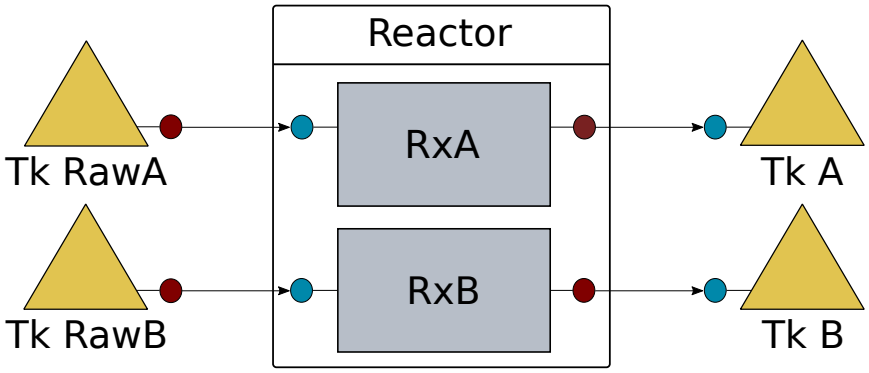

Figure 11. UOPSS representation for the example

logical variables as binary and relax the rest to continuous variables in the $[0, 1]$ range. However, declaring all the logical variables as binaries can be beneficial for some solvers, allowing to do more inference in the presolve process (Savelsbergh, 1994). This was also observed by Zyngier et al. (2018) in application of the UOPSS framework to the optimization of the operation of rail lines. The relationship between the

su_{ijt} : a binary variable indicating if task j starts in unit i at the beginning of period t
sd_{ijt} : a binary variable indicating if task j is shutdown in unit i at the beginning of period t
sw_{ijjt} : a binary variable indicating if task j continues operation in unit i at period t. Referred by the authors as "switch-to-self"
y_{ijt} : a binary variable indicating if task j is active in unit i in period t
b_{ijt} : the size of the batch processed in period t, executing task j in unit i
inv_{ist} : inventory of material s at tank i at the end of period t
xo_{pt} : flow through outlet port p at period t
xi_{pt} : flow through inlet port p at period t
$x_{pp't}$: flow between port p and port p' at period t

logical variables for a task with 3 periods of duration is shown in Fig. 12.

$$\sum_{j\in J_i}\sum_{\tau=0}^{PT_{ij}} su_{ijt-\tau} \leq 1 \qquad \forall i,t \tag{15}$$

$$y_{ijt} = \sum_{\tau=0}^{PT_{ij}-1} su_{ijt-\tau} \qquad \forall i,j\in J_i,t \tag{16}$$

$$sw_{ijjt} = \sum_{\tau=1}^{PT_{ij}-1} su_{ijt-\tau} \qquad \forall i,j\in J_i,t \tag{17}$$

$$sd_{ijt} = su_{ijt-PT_{ij}} \qquad \forall i,j\in J_i,t,t-PT_{ij}\geq 1 \tag{18}$$

$$y_{ijt} - y_{ijt-1} - su_{ijt} + sd_{ijt} = 0 \qquad \forall i,j\in J_i,t\geq 1 \tag{19}$$

$$y_{ijt} - y_{ijt-1} - su_{ijt} - sd_{ijt} - 2sw_{ijjt} = 0 \qquad \forall i,j\in J_i,t\geq 1 \tag{20}$$

$$su_{ijt} + sw_{ijjt} \leq 1 \qquad \forall i,j\in J_i,t \tag{21}$$

$$sd_{ijt} + sw_{ijjt} \leq 1 \qquad \forall i,j\in J_i,t \tag{22}$$

The next set of constraints, given by Eqs. (23)–(24) are logistic constraints relating the size of the batch with the start-up variables. At the same time, they are used to set bounds for the batch variables.

$$b_{ijt} \leq b_{ij}^{U} su_{ijt} \qquad \forall i,j\in J_i,t \tag{23}$$

$$b_{ijt} \geq b_{ij}^{L} su_{ijt} \qquad \forall i,j\in J_i,t \tag{24}$$

Finally, the flow of materials and resources is controlled by Eqs. (25)–(29). The complete UOPSS model

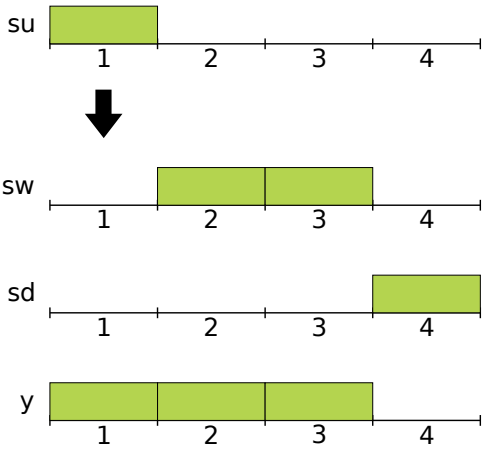

Figure 12. relationship between logic variables in UOPSS

formulation is given by Eqs. (14)–(29)

$$xi_{pt} = \rho_{ijp} b_{i,j,t+1} + \mu_{ijp} su_{i,j,t+1} \qquad \forall i, j \in J_i, p \in IP_{ij}, t \leq T \tag{25}$$

$$xo_{pt} = \rho_{ijp} b_{i,j,t-PT_{ij}-1} + \mu_{ijp} su_{i,j,t-PT_{ij}-1} \qquad \forall i, j \in J_i, p \in OP_{ij}, t \tag{26}$$

$$inv_{ist} = inv_{ist-1} + xi_{pt} - xo_{p't} \qquad \forall i, s, t > 1, p \in IP_{is}, p' \in OP_{is} \tag{27}$$

$$xi_{pt} = \sum_{p' \in O_p} x_{p'pt} \qquad \forall p, t \tag{28}$$

$$xo_{pt} = \sum_{p' \in D_{ijp}} x_{pp't} \qquad \forall p, t \tag{29}$$

To consider sequence-dependent changeover a new binary variable, yy, to capture the last operation performed is included. Also, a new set of constraints given by Eqs. 30–33 must be added.

$$\sum_{j \in PJ} yy_{ijt} = 1 \qquad \forall i, t \tag{30}$$

$$y_{ijt} \leq yy_{ijt} \qquad \forall i, j \in PJ, t \tag{31}$$

$$yy_{ijt} - yy_{ijt-1} - su_{ijt} \leq 0 \qquad \forall i, j \in PJ, t \geq 1 \tag{32}$$

$$yy_{ijt-1} + yy_{ikt} - 1 + su_{ikt} - \sum_{m \in M(j,k)} sd_{imt} \leq 1 \qquad \forall i, j, k, t > 1 \tag{33}$$

where PJ is the set of production tasks, and $M(j,k)$ is the set of maintenance tasks m (or cleaning) to go from task j to task k. Eq. (30) ensures a single memory variable for a production task is always active, Eq. (31) sets the memory variable every time a task is active, Eq. (32) indicates that if the memory variable

becomes active for a task (set to 1), the task is actually started, and Eq. (33) indicates that if there is change in the memory variable from task j to task k, task k needs to start and a changeover task m must end in the same period. Unlike the case of the STN framework, the number of sequence-dependent changeover constraints in the UOPSS framework is $O(T)$ (Eq. 33), compared to the $O(T^2)$ of the STN case (Eq. 7) (Kelly and Zyngier, 2007). Fig 13 shows the UOPSS diagram for Example 1 with a cleaning operation explicitly considering the consumption of cleaning buffer. To include QBC, Eq. (33) is replaced

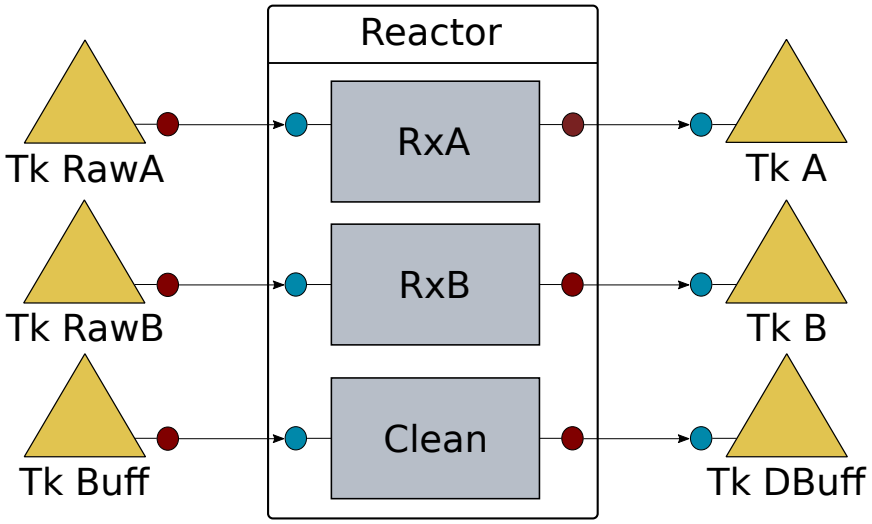

Figure 13. UOPSS representation for the example with sequence-dependent changeovers. Buff:Cleaning Buffer, DBuff:Dirty Buffer

by Eq. (34), where the last term allows to override the cleaning operation if $U_{jk}-1$ batches of task k are performed after the first ocurrence of task k. An example of the application of the constraint is presented in Appendix C.

$$yy_{ijt-1} + yy_{ikt} - 1 + su_{ikt} - \sum_{m \in MJ(j,k)} sd_{imt} - \sum_{\alpha=1}^{U_{jk}-1} \frac{1}{U_{jk}-1} su_{ik(t+\alpha PT_{ik})} \leq 1 \qquad \forall i,j,k,t>1 \qquad (34)$$

The UOPSS framework also includes more unit types such as pipelines and stacks, different constraints for the fixed batch size, variable processing time, and more complex processes with intermediate withdrawals of material and multiple feeds (fed-batch). In this paper we have chosen to present the simplest case for illustrative purposes. Additional details on the different formulations options included in UOPSS can be found in Zyngier and Kelly (2009).

5 Case Studies

In this section we explore the modeling and computational aspects the different scheduling frameworks applying them to a case study. The case study considered is the classic example of Kondili et al. (1993), presented in the paper that originally introduced the STN formulation. The problem consists on 4 units, a 100L heater that can perform a heating task, 50L and 80L reactors that can perform either of 3 reactions, and batch distillation column that can perform a separation task. The goal is to produce 2 products starting from 3 raw materials. Notice that the QBC considered is for these two products.

5.1 Representation

The first difference between the scheduling frameworks lies in the schematic representation of the problem. The diagram helps to gain understanding on the problem and facilitates the modeling task. The STN representation of the problem is given by Fig. 14.

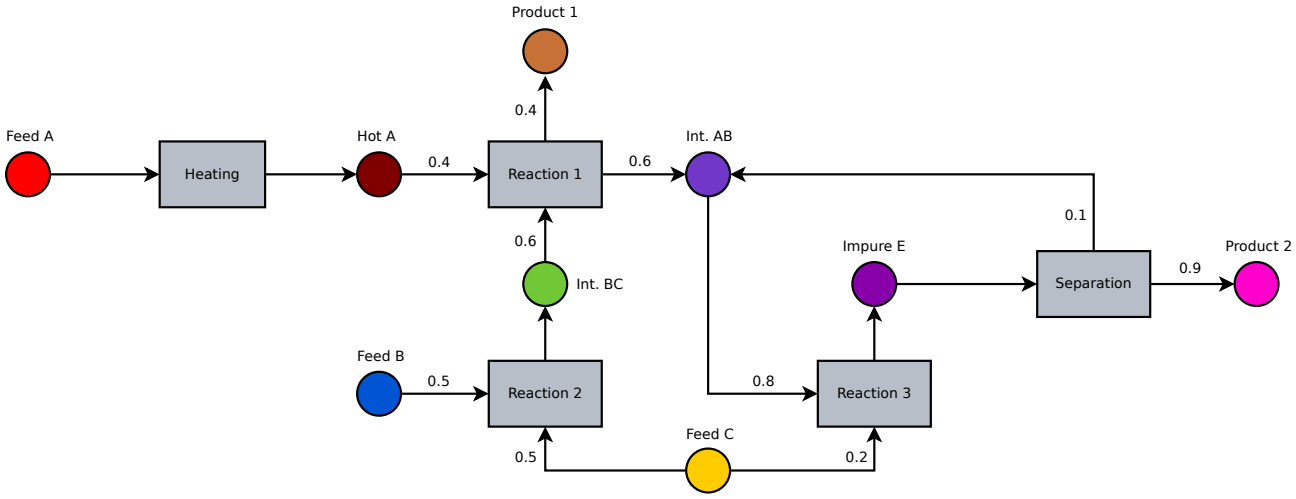

Figure 14. STN representation of the Case Study 1

The focus on processing operations of the STN framework can be seen in Fig. 14; the physical equipment is absent. In the RTN representation (Fig. 15), the processing units and other resources are explicitly captured. The equipment is present as a resource required for each reaction. Because the reactors are of different size they are represented as two separate resources. The double-arrow notation indicates that the tasks "consumes" at the beginning of the task, and "produces" an equipment at the end of it. The UOPSS representation from Fig. 16 is completely different from the previous two as it is based on the process diagram flowsheet. This could be advantageous to address industrial applications

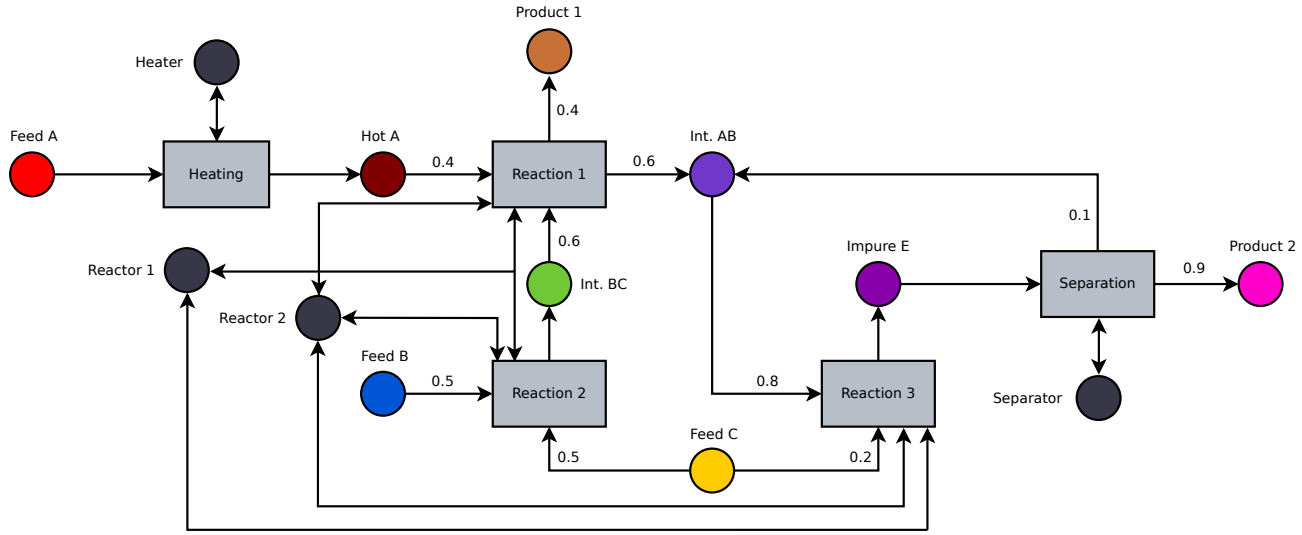

Figure 15. RTN representation of the Case Study 1

in which the PFDs are available.

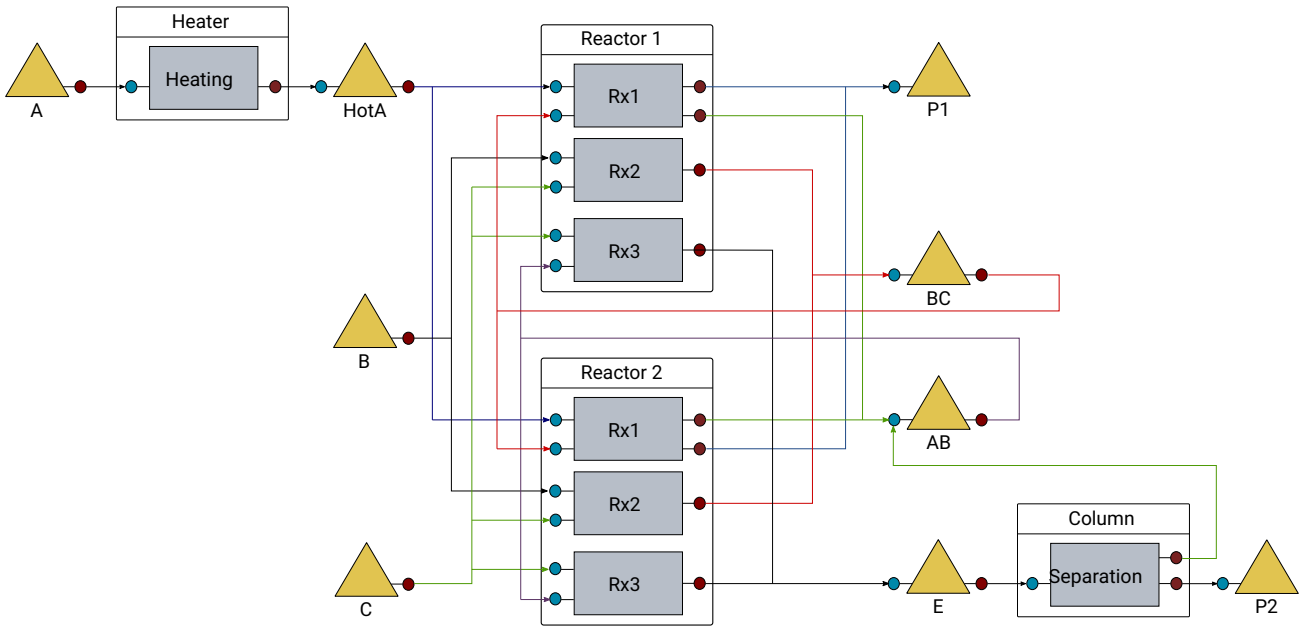

Figure 16. UOPSS representation of the Case Study 1

5.2 Performance of the Base Models

The next aspect we compare is the computational performance of the models in the absence of changeovers. To gain some insight in this issue we apply each of the formulations to the case study one for horizon lengths of 300 and 1,000 periods. Table 1 summarizes the model sizes obtained before and after presolve.

From the model sizes, the first observation that can be made is that the UOPSS formulation has a

	300 periods			1000 periods		
	Cons	Vars	Bin	Cons	Vars	Bin
Input						
STN	8,709	7,518	2,400	29,009	25,018	8,000
RTN	8,713	8,726	2,400	29,013	29,026	8,000
UOPSS	46,843	37,281	13,200	156,043	124,081	44,000
Presolve						
STN	6,821	5,928	2,372	22,921	19,928	7,972
RTN	4,148	6,509	2,961	13,952	21,913	9,961
UOPSS	9,469	7,403	3,845	31,875	24,909	12,945

Table 1. Model sizes for case study 1 in terms of constraints (Cons), variables (Vars), and binary variables (Bin)

significantly larger number of variables and constraints. However, most of them are redundant as they are removed in the presolving process. The resulting model is still larger than the STN formulation. The model sizes for STN and RTN are similar in this case study because of the absence of identical units that would help decrease the size of the RTN model. The computational performance of the models is shown in Fig. 17.

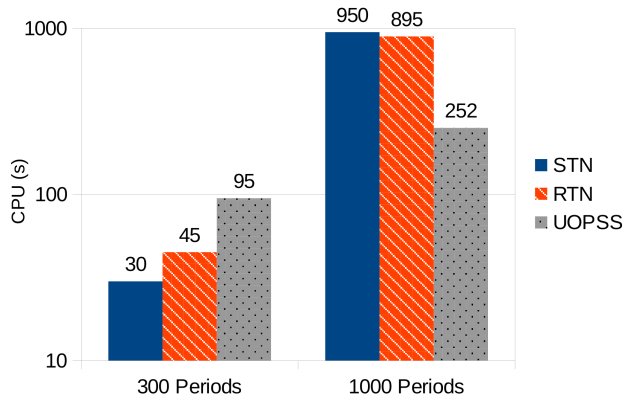

Figure 17. CPU time to reach 0.5 % gap for the models in absence of changeovers

The results from Fig. 17 show that the STN and RTN models have similar performance. The UOPSS model is the slowest for the 300 periods case, but the fastest for the 1,000 period case. This difference is indication of both the size and the tightness of the formulation. For small models, the size of the constraint matrix plays an important role in the solution time, yet for larger models a tighter formulation leads to a more efficient search procedure.

5.3 Performance with sequence-dependent changeovers and quality-based changeovers

The next aspect to explore is the performance of the models in presence of sequence-dependent changeovers (SDC). Since the three frameworks have different ways of handling this feature, the computational results are also affected. A first observation is that the number of constraints required to represent SDC can be very large, and very few of them are active at the optimal solution. Therefore, a useful strategy that can be implemented in modern solvers is to start the problem without these constraints and add them if violated in a lazy callback. This strategy is suitable for both STN, and UOPSS formulation. However, it is not suitable for the RTN case because it handles sequence-dependent changeovers through reformulation with auxiliary tasks and resources. The same concepts described are also applied to the case of QBC. Table 2 shows the model sizes of all the formulations, including the cases in which lazy constraints are considered. For the STN and UOPSS formulations, the model sizes with QBC are the same as the model size with SDC.

	300 periods			1000 periods		
	Cons	Vars	Bin	Cons	Vars	Bin
Input						
STN-C/QBC	100,211	9,322	3,000	1,034,011	31,022	10,000
STN-C/QBC lazy	10,511	9,322	3,000	35,011	31,022	10,000
RTN-C	11,717	11,730	3,600	39,017	39,030	12,000
RTN-QBC	16,521	16,538	5,400	55,021	55,038	18,000
UOPSS-C/QBC	63,039	49,895	21,600	210,039	166,095	72,000
UOPSS-C/QBC lazy	62,441	49,895	21,600	208,041	166,095	72,000
Presolve						
STN-C/QBC	94,732	6,520	2,964	out of memory		
STN-C/QBC lazy	6,843	6,550	2,978	22,980	21,978	9,978
RTN-C	5,437	8,913	3,571	17,947	29,913	11,971
RTN-QBC	7,117	11,275	4,165	23,922	37,881	13,965
UOPSS-C/QBC	11,274	8,110	4,455	37,874	26,911	14,955
UOPSS-C/QBC lazy	10,729	15,263	11,679	35,994	51,028	39,008

Table 2. Model sizes for the Kondili example with SDC and QBC in terms of constraints (Cons), variables (Vars), and binary variables (Bin)

The computational performance for the models is shown in Fig 18 for the 300 periods case, and in Fig. 19 for the 1,000 periods case.

The 300 period case shows that the STN formulation has a slow performance due to the large number of SDC constraints. This issue is solved by the use of lazy constraints, obtaining a similar performance to RTN. The UOPSS formulation has the best performance, which only improves the use of lazy constraints.

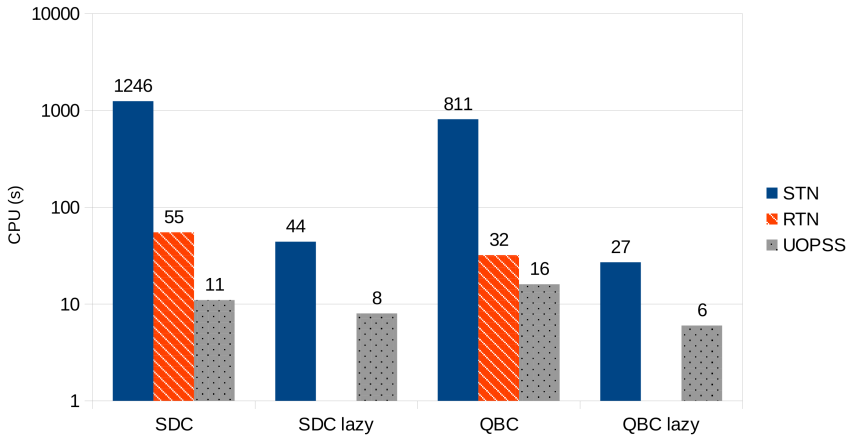

Figure 18. Time to reach 1 % gap for 300-period models with SDC and QBC

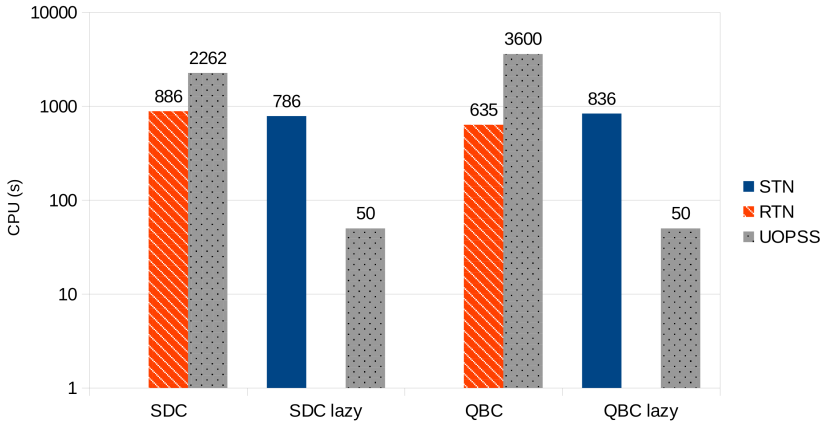

Figure 19. Time to reach 1 % gap for 1,000-period models with SDC and QBC

The results for the 1,000 periods case (Fig. 19) are similar. The STN formulation with SDC constraints runs out of memory and can not complete the solution process. When including lazy constraints it is possible to obtain solutions with a similar performance to RTN. Compared to UOPSS with SDC constraints RTN performs better. However, a single pair of changeovers was considered in the case study. The RTN formulation grows in auxiliary tasks and resources with each pair of changeovers considered. The UOPSS formulation with lazy constraints is the overall best performer. Another important observation is that the inclusion of QBC for STN and UOPSS does not bring additional computational burden to the optimization.

6 Discussion

Several interesting insights can be obtained from the computational results. The development of several frameworks for short term scheduling is a contribution to address a large variety of problems. It is important to understand the strengths and weaknesses of each formulation to select the right model for the application.

The first difference between the frameworks, the schematic representation, has a critical importance in gaining understanding of the problem, helping to simplify the modeling task. The STN representation is simple, yet very familiar for process engineers as it focuses on the processing operations to go from raw materials into finished goods. The explicit inclusion of resources in the RTN formulation has the advantage to seamlessly integrate other non-material resources, such as energy or human operators. This allows to address applications in which those resources are important, or applications not explicitly related to short-term production planning (Castro et al., 2013; Chen et al., 2010). On the other hand, this generalization of resources makes the representation less intuitive, and requires a modeler with a high level of expertise to use it. The schematic representation for the UOPSS framework is not based on task networks, but on the process flowsheet diagram. This allows to connect the physical interconnections represented by the PFD with the process network from the STN representation, which makes it very intuitive. Once the schematic representation for the problem is complete, the modeling is almost straightforward.

A second aspect we were interested in analyzing is the extensibility of the frameworks. We have chosen to evaluate this aspect through the inclusion of QBC because it is a practical feature present in chemical plants that has been neglected in the literature. Since the STN, and UOPSS formulations have some logical constraints, it was possible to add QBC without major modifications to the models. On the other hand, the RTN formulation becomes quite complex with the inclusion if this simple feature. The RTN formulation is indeed versatile for addressing a large number of features. However, these extensions require time, creativity, and a fair share of trial and error. Because of being a very explicit formulation, the UOPSS framework is more amenable to extensions.

The models resulting from applying each of the formulations vary in size and performance. The UOPSS formulation results in models with the largest number of variables and constraints, even after applying the presolve. However, its performance scales better than the STN and RTN formulations. This could be explained by the inference commercial solvers can make in a formulation as explicit as UOPSS. The STN formulation still remains simple to use, with good performance for small models, yet

the explicit inclusion of SDC constraints has a detrimental effect on its performance. We have shown that this issue can be handled by the inclusion of lazy constraints. In all the cases these kind of constraints were included the performance of the model improved. Therefore, we strongly recommend to consider them when dealing SDC (or QBC). For the largest models addressed the UOPSS formulation with lazy constraints obtained the best computational performance. To generalize these results, however, requires further experimentation with other objective functions and other applications is required. Nevertheless, the insights collected are valuable guidance for future research directions. Our findings, in particular about the UOPSS framework, indicate it is a powerful modeling tool that deserves more attention.

7 Conclusions

Three general purpose scheduling frameworks were compared and extended, with an emphasis on quality-based changeovers. They were developed for multipurpose plants of the chemical industry. However, they are general enough for a wide range of applications. The comparison performed, the first of its kind, serves to provide guidelines to optimization practitioners to select the right framework for the desired application.

The STN formulation is the simplest to implement, but it is limited in both modeling and computational performance for large problems. Extensions to the RTN formulation are challenging, because it is not as intuitive to use. Finally, the UOPSS formulation is easy to implement and extend; it also scales very good with problem size.

The STN and UOPSS formulation benefit from the use of lazy constraints to handle SDC constraints, both in the standard changeovers case and in the QBC case. The latter does not add computational burden to the problem, and represents a contribution for broader applications in the chemical industry.

8 Acknowledgments

The authors acknowledge the financial support from the Dow Chemical Company, the Center for Advanced Process Decision-making (CAPD) from Carnegie Mellon University, and the Government of Chile through its Becas Chile program.

References

Brunaud, B. and Grossmann, I. E. (2017). Perspectives in multilevel decision-making in the process industry. *Frontiers of Engineering Management*, 4(3):256–270.

Castro, P. M., Sun, L., and Harjunkoski, I. (2013). Resource–task network formulations for industrial demand side management of a steel plant. *Industrial & Engineering Chemistry Research*, 52(36):13046–13058.

Chen, C.-L., Chang, C.-Y., and Lee, J.-Y. (2010). Resource-task network approach to simultaneous scheduling and water minimization of batch plants. *Industrial & Engineering Chemistry Research*, 50(7):3660–3674.

Floudas, C. A. and Lin, X. (2004). Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review. *Computers & Chemical Engineering*, 28(11):2109–2129.

Grossmann, I. E. (2012). Advances in mathematical programming models for enterprise-wide optimization. *Computers & Chemical Engineering*, 47:2–18.

Harjunkoski, I., Maravelias, C. T., Bongers, P., Castro, P. M., Engell, S., Grossmann, I. E., Hooker, J., Méndez, C., Sand, G., and Wassick, J. (2014). Scope for industrial applications of production scheduling models and solution methods. *Computers & Chemical Engineering*, 62:161–193.

Kelly, J. D. and Zyngier, D. (2007). An improved MILP modeling of sequence-dependent switchovers for discrete-time scheduling problems. *Industrial & Engineering Chemistry Research*, 46(14):4964–4973.

Kondili, E., Pantelides, C., and Sargent, R. (1993). A general algorithm for short-term scheduling of batch operations–I. MILP formulation. *Computers & Chemical Engineering*, 17(2):211–227.

Méndez, C. A., Cerdá, J., Grossmann, I. E., Harjunkoski, I., and Fahl, M. (2006). State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers & Chemical Engineering*, 30(6):913–946.

Pantelides, C. C. (1994). Unified frameworks for optimal process planning and scheduling. In *Proceedings on the Second Conference on Foundations of Computer Aided Process Operations*, pages 253–274. CACHE Publications New York.

Sargent, R. (2005). Process Systems Engineering: A retrospective view with questions for the future. *Computers & Chemical Engineering*, 29(6):1237–1241.

Savelsbergh, M. W. (1994). Preprocessing and probing techniques for mixed integer programming problems. *ORSA Journal on Computing*, 6(4):445–454.

Shah, N., Pantelides, C., and Sargent, R. (1993). A general algorithm for short-term scheduling of batch operationsii. computational issues. *Computers & Chemical Engineering*, 17(2):229–244.

Zyngier, D. and Kelly, J. D. (2009). Multi-product inventory logistics modeling in the process industries. In *Optimization and Logistics Challenges in the Enterprise*, pages 61–95. Springer.

Zyngier, D. and Kelly, J. D. (2012). UOPSS: a new paradigm for modeling production planning & scheduling systems. In *Symposium on Computer Aided Process Engineering*, volume 17, page 20.

Zyngier, D., Lategan, J., and Furstenberg, L. (2018). A process systems approach for detailed rail planning and scheduling applications. *Computers & Chemical Engineering*, 114:273–280.

A Nomenclature

Sets	
I	All units
I_j	Units capable of executing task j
I_k^P	
I_k^C	
J	All tasks including processing and cleaning
J_i	Tasks performed by unit i
PJ	Processing tasks
$M(j,k)$	Cleaning tasks between j and k
Parameters	
η_k	Price of state/resource/material k
PT_{ij}	Processing time of task j at unit i
Variables	
w_{ijt}	1 if task j is performed at unit i in period t
b_{ijt}	size of batch of task j produced at unit i in period t
s_{kt}	inventory of state k at the end of period t
w_{it} :	a binary variable indicating if task i starts at the beginning of period t
b_{it} :	the size of the batch processed in period t, executing task i
r_{kt} :	inventory of resource k in period t
su_{ijt} :	a binary variable indicating if task j starts in unit i at the beginning of period t
sd_{ijt} :	a binary variable indicating if task j is shutdown in unit i at the beginning of period t
sw_{ijjt} :	a binary variable indicating if task j continues operation in unit i at period t. Referred by the authors as "switch-to-self"
y_{ijt} :	a binary variable indicating if task j is active in unit i in period t
b_{ijt} :	the size of the batch processed in period t, executing task j in unit i
inv_{ist} :	inventory of material s at tank i at the end of period t
xo_{pt} :	flow through outlet port p at period t
xi_{pt} :	flow through inlet port p at period t
$x_{pp't}$:	flow between port p and port p' in period t

B Example of QBC constraint for STN

To better understand the application of the QBC constraint defined by Eq. (8), consider an example with a unit R in which the task RxA, of 1 period of duration, is executed in period 1, and task RxB, also of 1 period of duration, is executed in period 4 (Fig. 20). To transition from RxA to RxB the cleaning task CL, i.e. $M(RxA, RxB) = \{CL\}$; or alternatively 3 batches of B must be produced in a row executing task RxB.

The corresponding QBC constraint for $t_1 = 1$ and $t_2 = 4$ is given by Eq. (35). To simplify the

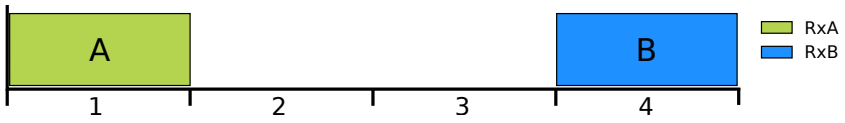

Figure 20. QBC constraints application example

notation, RxA and RxB are simply referred as A and B.

$$w_{R,CL,2} + w_{R,CL,3} \geq w_{R,A,1} + w_{R,B,4} - 1 - \sum_{n}(w_{R,n,2} + w_{R,n,3}) - \frac{1}{2}w_{R,B,5} - \frac{1}{2}w_{R,B,6} \tag{35}$$

If no other tasks besides the ones shown in Fig. (20) are started, the constraint is violated ($0 \geq 1$). On the other hand, the constraint is satisfied if any of the following options are met:

1. The cleaning task CL is executed either in period 2 or period 3
2. Any task n is executed either in period 2 or period 3
3. The task RxB is executed in both periods 5 and 6

Note that the case where RxB is executed in periods 2 and/or 3 is covered by another constraint.

C Example of QBC constraint for UOPSS

The QBC constraints in the UOPSS formulation are use a single time index and take advantage of the memory variable yy. To understand the application of Eq. (34) consider the same example described in Appendix B. The value of the memory variables for each period in the example is given in Table 3. The

Table 3. Values of yy variable for each period in the example

Period	1	2	3	4
yy_A	1	1	1	0
yy_B	0	0	0	1

corresponding constraint for $t = 4$ is given by Eq. 36.

$$yy_{R,A,3} + yy_{R,B,4} - 1 + su_{R,B,4} - sd_{R,CL,4} - \frac{1}{2}su_{R,B,5} - \frac{1}{2}su_{R,B,6} \leq 1 \tag{36}$$

If no changeover task or QBC is executed the constraint is violated ($2 \leq 1$). The constraint can be satisfied if either the cleaning task is executed in period 3 and finished in period 4 ($sd_{R,CL,4} = 1$), or if

batches of B are produced in both period 5 and 6 ($su_{R,B,5} = su_{R,B,6} = 1$).