Short Note

Strengthening Discrete-Time Scheduling Formulations by Introducing the Concept of Campaigns

Diego C. Cafaro[a] & Ignacio E. Grossmann[b]

[a] *INTEC(UNL-CONICET), Güemes 3450, 3000 Santa Fe, ARGENTINA*
[b] *Dept. of Chem. Eng., Carnegie Mellon University, 5000 Forbes Ave, 15213 Pittsburgh, PA*

Abstract

Discrete-time, Mixed-Integer Linear Programming (MILP) models are frequently used for scheduling problems in which the utilization of resources and/or the determination of costs and revenues imply complex time-dependent functions, directly related to the development of the tasks. One of the main challenges in these models is to deal with changeovers, which are one of the most complicating features. In this short note, we provide some insight to effectively model the scheduling of operations by means of the concept of campaigns. This allows to reduce the computational burden by three orders of magnitude when compared to conventional approaches.

Introduction

Time-indexed Mixed-Integer Linear Programming (MILP) optimization approaches are the first choice to model scheduling problems in which the consumption of resources, the determination of costs and/or the calculation of revenues imply complex time-dependent functions, directly related to the implementation of tasks (Kondili, Pantelides & Sargent, 1993; Shah, Pantelides & Sargent, 1993; Castro, Barbosa-Povoa & Matos, 2001; Moniz, Barbosa-Póvoa & de Sousa, 2013). In discrete-time formulations, holding and backlogging costs can be modeled with linear terms (Velez, Dong & Maravelias, 2017), hyperbolic decline functions can also be accounted for (Cafaro & Grossmann, 2014), and piecewise, nonlinear, non-differentiable functions can be also tracked linearly (Zeng & Cremaschi, 2017; Drouven, Cafaro & Grossmann, 2017). Although these MILP models lead to tight linear (LP) relaxations, most of the complexity of the discrete-time models comes from their large number of constraints, especially when sequence-dependent changeovers are considered. Ondeck et al. (2019) prove that most of the computational burden when solving the optimal planning of wellpad development operations comes from the need to consider resource mobilization costs. However, under certain conditions, changeover costs can be modeled more effectively as is shown in this short note. Specifically, we address scheduling problems in which changeover (or resource mobilization) costs are paid every time a new sequence of jobs of the same class is started. We propose to model sequences of operations by means of the concept of campaigns, which yields very tight formulations. This reduces the computational burden by about three orders of magnitude, when compared to conventional discrete-time approaches.

Motivating Example

Assume we are given a set of n jobs (k = 1…n) that must be executed in a single machine. Jobs are of different classes and there are sequence-dependent changeover costs, but depending only on the classes. More precisely, every time a new job of class i starts, a fixed changeover cost c_i must be paid, except for the case when a job of the same class is being completed, and the job sequence is continued with no interruption. A real-world problem with these characteristics is the shale gas well development planning problem, in which a fracturing crew (machine) has to complete a set of wells (jobs), grouped in different wellpads (classes). Many wells of the same pad might be completed during a single visit of the fracturing crew to the pad. Longer stays, completing more wells in a row, reduce mobilization costs. However, the longer the crew stays at a pad, the later revenues are obtained from gas production, since wells do not produce gas until they are turned-in-line, and turn-in-line operations need to be postponed after long fracturing campaigns. The problem is very hard to solve to optimality (Ondeck et al., 2019), and one of the main reasons is the way changeover or mobilization costs are modeled. From now on, we focus on how to model changeover costs.

Suppose we need to schedule an even number of jobs (n) in a single machine, with half of the jobs belonging to class A, and the other half to class B. For simplicity, suppose that every job has a deterministic duration of q time periods, no matter the class. Regarding changeovers, we account for three different situations:

(a) Changeover costs are null for every pair of subsequent jobs involving the same class (sequence A-A or B-B), as shown in Figure 1.
(b) Changeover costs are equal to c_i if a job of class i follows a job of a different class (sequence A-B or B-A).
(c) Changeover costs are equal to c_i if a job of class i is performed after an idle period.

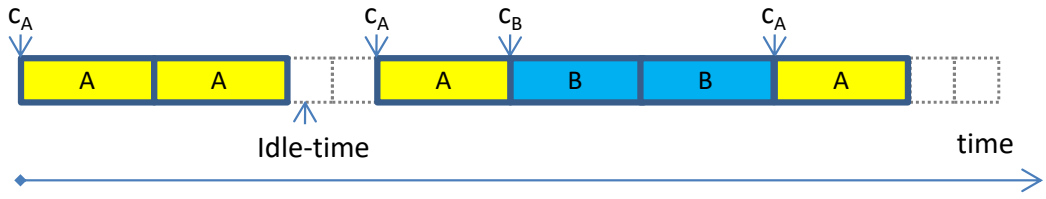

Figure 1. Gantt chart illustrating changeover costs for a sequence of jobs of two classes in a single machine scheduling problem

In the simplest case, idle times are not allowed, and alternative (c) is not an option. Besides, changeover times are neglected, meaning that the total number of time periods in the time horizon is simply T = 1… $q \cdot n$. The aim is to sequence every job in the single machine so that the total profit is maximized. If revenues are independent of the production sequence, the optimal solution is straightforward: perform all jobs of class A and then all of class B, or vice-versa. However, to investigate the computational performance of two different MILP formulations to solve this problem, we consider: (a) a conventional discrete, time-indexed scheduling formulation (Kondili, Pantelides & Sargent, 1993; later improved by Shah, Pantelides & Sargent, 1993; Wolsey, 1997; and Velez, Dong & Maravelias, 2017), and (b) a tighter, higher-dimensional formulation, by introducing the concept of campaigns. Later in the paper, we elaborate on the strength

of both approaches by focusing on the model LP relaxations, and provide some insights on the use of discrete-time approaches for scheduling problems.

Conventional Discrete-Time Formulation

By assuming that all the jobs of the same class are identical, the discrete-time MILP formulation can be defined in terms of the index of classes *i*. The variable $y_{i,t}$ indicates that a job of class *i* starts at time period *t*. Since every job has to be performed once, an aggregate constraint can be imposed for every index *i*, as shown in (1). Moreover, one job must be performed at a time, leading to the non-overlapping constraint of the STN approach, as expressed in (2). This constraint is usually referred to as the backward aggregating constraint (Shah, Pantelides & Sargent., 1993).

$$\sum_{t=1}^{T} y_{i,t} = n_i \qquad \forall i = A, B \tag{1}$$

$$\sum_{i=A,B} \sum_{t'=t-q+1}^{t} y_{i,t'} \leq 1 \qquad \forall t = 1 \dots T \tag{2}$$

To leave no incomplete jobs at the end of the time horizon ($t = T$), we also impose constraint (3).

$$\sum_{i=A,B} \sum_{t=T-q+2}^{T} y_{i,t} = 0 \tag{3}$$

In this problem, an effective formulation to account for the changeovers can be derived from propositional logic (see Eq. 4). As proposed by Ondeck et al. (2019), the changeover (in their case, the resource mobilization) condition is implied by the start of a new job of class *i* at a certain time period *t*, and the absence of a job of the same class being completed at the previous time period *t*-1.

$$Y_{i,t} \wedge \neg Z_{i,t-1} \Longrightarrow X_{i,t} \qquad \forall i = A, B;\ t = 1 \dots T \tag{4}$$

Variable $Y_{i,t}$ is equivalent to the condition $y_{i,t} = 1$, while $Z_{i,t-1}$ is true if and only if a job of class *i* is being performed at time *t*-1. In propositional logic terms, it can be expressed as in Eq. (5).

$$Z_{i,t} \Leftrightarrow \bigvee_{t'=t-q+1}^{t} Y_{i,t'} \qquad \forall i = A, B;\ t = 1 \dots T \tag{5}$$

After obtaining the Conjunctive Normal Form (CNF) of the logic constraints and converting them into algebraic inequalities (Raman & Grossmann, 1994), proper arrangement of the terms yields constraint (6) (Ondeck et al., 2019).

$$x_{i,t} \geq y_{i,t} - \sum_{t'=t-q}^{t-1} y_{i,t'} \qquad \forall i = A, B;\ t = 1 \dots T \tag{6}$$

Variable $x_{i,t}$, which may not necessarily be defined as 0-1, takes value 1 whenever a job of class *i* starts at period *t* and the machine was not performing another job of the same class right before the one being started at *t*.

Moreover, note that the non-overlapping constraint (2) yields the implication: $Y_{i,t} \Longrightarrow \bigwedge_{t'=t-q+1}^{t-1} \neg Y_{i,t'}$. In other words, $y_{i,t} = 1 \Longrightarrow \sum_{t'=t-q+1}^{t-1} y_{i,t'} = 0$, from which we derive the simplified expression (7).

$$x_{i,t} \geq y_{i,t} - y_{i,t-q} \qquad \forall i = A, B;\ t = 1 \dots T \tag{7}$$

Finally, the objective function seeks to minimize the total changeover cost, given by Eq. (8).

$$Min\ TC = \sum_{i=A,B} \sum_{t=1}^{T} c_i \cdot x_{i,t} \tag{8}$$

In summary, the conventional MILP formulation is given by the minimization of (8), subject to constraints (1), (2), (3) and (7).

If *n* is the total number of jobs, and n_i is the number of jobs of class *i*, the LP relaxation of this problem yields the solution $y_{i,t} = n_i/n$ for periods $t = 1, q+1, 2q+1, \dots (n-1)q+1$, and $y_{i,t} = 0$ in any other case. In the illustrative example, where $n_i = n/2$, $y_{i,t} = 0.5$ for both classes A and B at the periods $t = 1, q+1, 2q+1, \dots (n-1)q+1$, and $y_{i,t} = 0$ in any other case. The total changeover cost amounts to *TC* = 1, since in our simplified case c_i = 1 for any class *i*. Figure 2 illustrates the relaxed solution.

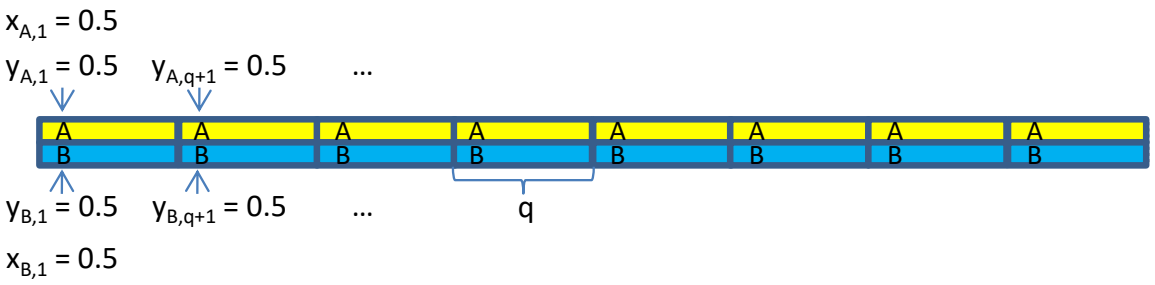

Figure 2. Illustration of the solution yielded by the relaxed LP of the conventional time-indexed scheduling formulation

In the relaxed LP solution, the jobs only pay for changeover costs at the beginning of the time horizon because for any subsequent starting period the single machine performs the same "fraction" (0.5) of each job, yielding $y_{i,t} - y_{i,t-q} = 0$ for all *t* > 1. Note that the solution at the root node is far from integrality, and certainly below the actual optimal value of the MILP ($TC^* = 2$).

Branching on any of the variables $y_{i,t}$ of the first time period yields a still fractional solution, whose optimal value depends on the number of jobs *n*. For instance, if *n* = 8, by imposing $y_{A,1} = 1$ we obtain the solution $y_{A,t} = 3/7$ and $y_{B,t} = 4/7$ for the periods $t = q + 1, 2q + 1, 3q + 1$, and $y_{i,t} = 0$ in any other case, with TC = 1.571. The solution after the first branching is depicted in Figure 3. In total, we need at least *n*/2 depth-first branching to obtain an integer solution of our illustrative example. There may be more efficient ways of branching like, for instance, by partitioning the time domain in the summation of constraints (1). However, as we show later, it is still not easy for a modern MILP solver to come up with the optimal solution even using all the battery of current branch-and-cut strategies.

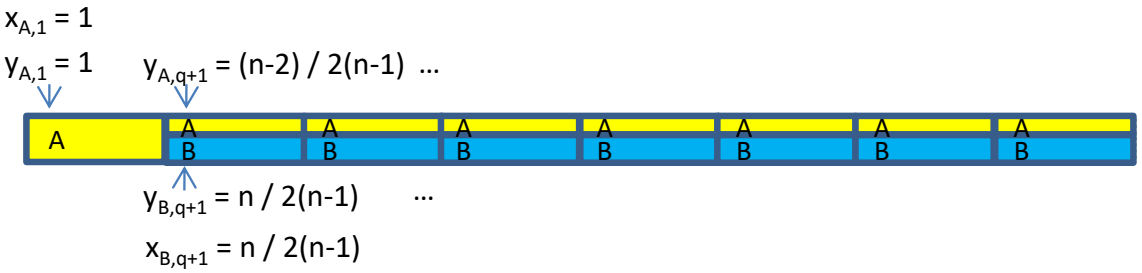

Figure 3. Illustration of the solution yielded by the branch and bound algorithm applied to the conventional time-indexed scheduling formulation, after the first branching

It can be easily observed that solution degeneracy increases with the number of classes *i* and the number of items *n*, making it harder to find the optimal solution. Kelly & Zyngier (2007) also note degeneracy in the time domain when introducing changeover constraints in discrete-time formulations, highlighting that such inequalities are not strong for effective branching. It is quite clear that there is a need for a stronger formulation, and with that purpose we introduce the dimension of campaigns, as next explained.

Campaign-Based Formulation

By definition, a "campaign" is regarded as an uninterrupted sequence of jobs of the same class. By introducing the new dimension *r* = 1 … n_i we account for all possible campaign lengths (meaning the number of consecutive jobs) one may find in any feasible solution of the scheduling problem. Next, we introduce an extended set of binary variables $y_{i,t,r}$ which take value 1 whenever a campaign of *r* consecutive jobs of class *i* starts at period *t*, and zero otherwise. Therefore, the so-called campaign-based formulation of the scheduling problem can be modeled in terms of constraints (9) and (10), which follow directly from Eqs. (1) and (2).

$$\sum_{t=1}^{T} \sum_{r=1}^{n_i} r \cdot y_{i,t,r} = n_i \qquad \forall i = A, B \tag{9}$$

$$\sum_{i=A,B} \sum_{r=1}^{n_i} \sum_{t'=t-q\cdot r+1}^{t} y_{i,t',r} \leq 1 \qquad \forall t = 1 \ldots T \tag{10}$$

The set of possible campaign lengths for every class *i* is denoted R_i. In the most general case, $R_i = \{1, 2, \ldots, n_i\}$, where n_i is the number of jobs of class *i*. The optimization model might decide on the one hand to develop all the jobs of the same class in a single campaign of length n_i, or on the other hand, develop the jobs one by one, in n_i different campaigns of length 1.

As in the previous model, we also impose a boundary condition to prevent incomplete campaigns, which is enforced by Eq. (11).

$$\sum_{i=A,B} \sum_{r=1}^{n_i} \sum_{t=T-q\cdot r+2}^{T} y_{i,t,r} = 0 \tag{11}$$

What is certainly unique in this alternative formulation is the straightforward calculation of changeover costs. From the definition of campaigns, we can simply impose a changeover cost for every selected campaign, as stated by Eq. (12).

$$x_{i,t} = \sum_{r=1}^{n_i} y_{i,t,r} \qquad \forall i = A, B; t = 1 \ldots T \tag{12}$$

We notice that the condition $x_{i,t} \leq 1$ is guaranteed by constraint (10). Finally, the campaign-based formulation of our single machine scheduling problem is given by the minimization of function (8), subject to constraints (9), (10), (11) and (12).

When compared to the conventional formulation, the number of variables is much larger, but the number of constraints remains exactly the same, with one important difference; the last constraints (12), the only whose number grows with both the number of classes and jobs, turn into equalities. Moreover, the right hand side (RHS) of Eq. (12) is at least as large as the RHS of its analogous constraint (7), as shown in the disaggregated version of constraint (12). In Eq. (12'), variables $y_{i,t,1}$ (accounting for campaigns of length *r* = 1) have been replaced by their analogous variables $y_{i,t}$ of the conventional formulation.

$$x_{i,t} = y_{i,t} + \sum_{r=2}^{n_i} y_{i,t,r} \qquad \forall i = A, B; t = 1 \ldots T \tag{12'}$$

The strength of the campaign-based approach is also explained by comparing Eqs. (2) and (10). The non-overlapping condition (10) is more stringent than (2), since all the terms in the left hand side (LHS) of (2) are also in the LHS of (10), as shown in the disaggregated constraint (10').

$$\sum_{i=A,B} \sum_{t'=t-q+1}^{t} y_{i,t'} + \sum_{i=A,B} \sum_{r=2}^{n_i} \sum_{t'=t-q\cdot r+1}^{t} y_{i,t',r} \leq 1 \qquad \forall t = 1 \ldots q \cdot n \tag{10'}$$

More specifically, there are many more building blocks (all the campaigns with length *r* > 1) to be potentially arranged in the list of assignments of the single machine, within a fixed time horizon, as

illustrated in Figure 4. And more important, each building block permits by itself to precisely account for the changeover cost.

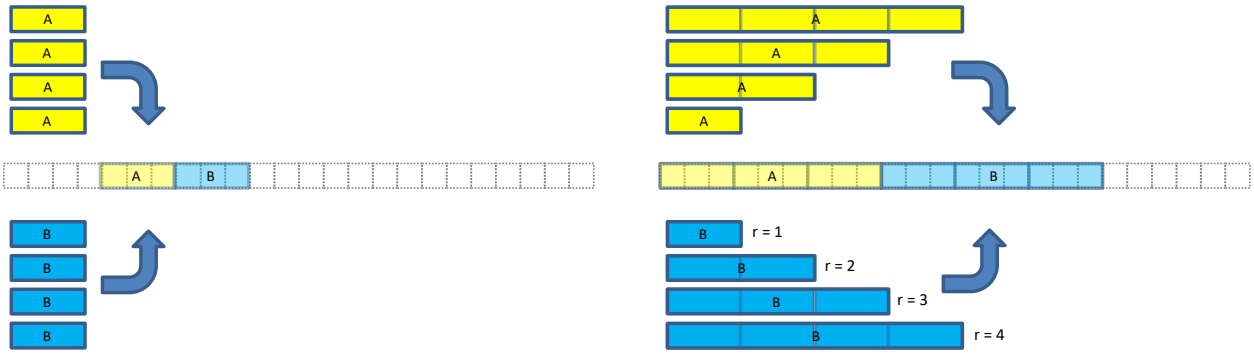

Figure 4. Comparison of the scheduling frameworks from the conventional (left) and the campaign-based (right) formulations

In logic terms, the existence of an extended campaign (of length $r > 1$) in the new model implies condition (13), which the conventional formulation has to infer by itself.

$$Y_{i,t,r} \Longrightarrow \bigwedge_{k=1}^{r} Y_{i,t+(k-1)q} \qquad \forall i = A,B; t = 1 \dots T; r = 1 \dots n_i \tag{13}$$

Note that the converse implication is not enforced by the constraints in the campaign-based model, meaning that an uninterrupted sequence of campaigns of shorter length is still a feasible solution, not necessarily constituting a single campaign. However, this is clearly discouraged by the objective function.

The CNF of (13) yields constraints (14) (Raman & Grossmann, 1994), which can be interpreted as the bridge constraints between both formulations.

$$y_{i,t+(k-1)q} \geq y_{i,t,r} \qquad \forall i = A,B; t = 1 \dots T; r = 1 \dots n_i; k = 1 \dots r \tag{14}$$

Overall, although the number of binary variables has increased by the dimension of campaign lengths r, the new formulation proves to be very tight, as explained next.

Numerical Results

Assume we are concerned with sequencing 40 jobs divided into 10 classes (4 jobs per class), each job having 5 periods of length. Changeover costs meet the assumption described in the Motivating Example and are equal to one for any job class. Idle times are not allowed and the time horizon comprises exactly 200 periods. The conventional STN formulation of this problem involves 4,001 variables (of which 2,000 are necessarily 0-1) and 2,212 equations. Even though the optimal solution is once again straightforward, i.e. perform all the jobs of the same class in a single campaign, with the campaigns in any order, CPLEX 29.1 (on an Intel Core i7 CPU, 2.60 GHz, 12 Gb RAM, single-thread mode) takes 52,500 iterations (475

nodes) and about 10 s to converge from the LP relaxed solution of $TC = 1$ to the optimal value of $TC^* = 10$. In contrast, the campaign-based approach finds the optimal solution at the root node, in 0.11 s. If the number of items per class is increased to 10, the conventional model comprises 10,001 variables and 5,512 equations, and CPLEX takes around 1 million iterations and 10 minutes to find the optimal solution, even though the lower bound hits the optimal value 2 minutes earlier. Interestingly, if we use the campaign-based formulation for this larger instance the optimal solution is once again found at the root node in less than one second (1500 iterations), although the number of variables is 5.5 times larger when compared to the conventional approach.

More complex examples are presented as Supplementary Material, with jobs of different duration and time dependent changeover costs, also allowing for idle times. Specifically, the models considered are generalizations of the models presented in this paper, accounting for any job duration d_i, time horizon length T, and time-dependent changeover costs $c_{i,t}$. In summary, we demonstrate that the campaign-based formulation is able to reduce the computational time and the number of iterations by three orders of magnitude, always finding the optimal integer solution in the relaxed LP.

Conclusion

Introducing the concept of campaigns in the scheduling of jobs or operations of different classes proves to be an effective strategy for strengthening discrete-time formulations for particular problems. The tightness of the model mainly relies on the ability to account for changeover costs in a very straightforward way. Instead of checking the occurrence of a changeover at every single period of the time horizon, changeover costs are charged on every single campaign. In general terms, the key is to define larger building blocks instead of isolated jobs to be conveniently arranged in the planning timeframe. If a certain price or benefit can be associated to each building block, even at the expense of more binary variables, the modeling effort and the associate computational burden can be reduced significantly. We note that the results found in this work are not limited by the number of job classes, the length of the operations, or the existence of idle time periods. The key assumption is that class-dependent changeover costs are paid every time a new job starts, except for the case when a job of the same class is being completed, and the campaign is continued with no interruption.

Acknowledgments

Financial support from the Wilton E. Scott Institute for Energy Innovation at Carnegie Mellon University, the Center for Advanced Process Decision Making, Dept. of Chemical Engineering (Carnegie Mellon University), CONICET and UNL is gratefully acknowledged.

References

1. *Cafaro DC, Grossmann IE (2014). Strategic planning, design, and development of the shale gas supply chain network. AIChE J., 60(6): 2122-2142*
2. *Castro P, Barbosa-Póvoa AP, Matos H (2001). An improved RTN continuous-time formulation for the short-term scheduling of multipurpose batch plants. Ind. Eng. Chem. Res., 40: 2059-2068.*

3. *Drouven M, Cafaro DC, Grossmann IE (2017). Stochastic programming models for optimal shale well development and refracturing planning under uncertainty AIChE J., 63(11): 4799-4813.*
4. *Kelly JD, Zyngier D. An improved MILP modeling of sequence-dependent switchovers for discrete-time scheduling problems. Ind. Eng. Chem. Res., 46(14): 4964-4973.*
5. *Kondili E, Pantelides CC, Sargent RWH (1993). A general algorithm for short-term scheduling of batch operations—I. MILP formulation. Comput. Chem. Eng., 17(2): 211-227.*
6. *Moniz S, Barbosa-Póvoa AP, de Sousa JP (2013). New general discrete-time scheduling model for multipurpose batch plants. Ind. Eng. Chem. Res., 52 (48): 17206-17220.*
7. *Ondeck A, Drouven M, Blandino N, Grossmann IE (2019). Multi-operational planning of shale gas pad development Comput. Chem. Eng., 126: 83-101.*
8. *Shah N, Pantelides CC, Sargent RWH (1993). A general algorithm for short-term scheduling of batch operations—II. Computational issues. Comput. Chem. Eng., 17(2): 229-244.*
9. *Velez S, Dong Y, Maravelias CT (2017). Changeover formulations for discrete-time mixed-integer programming scheduling models. European Journal of Operational Research, 260(3): 949-963.*
10. *Wolsey LA. (1997). MIP modelling of changeovers in production planning and scheduling problems. European Journal of Operational Research, 99(1): 154-165.*
11. *Zeng Z, Cremaschi S (2018) Multistage Stochastic Models for Shale Gas Artificial Lift Infrastructure Planning. Computer Aided Chemical Engineering, 44: 1285–1290.*
12. *Raman R, Grossmann IE (1994) Modelling and computational techniques for logic based integer programming. Comput. Chem. Eng., 18: 563-578.*

Supplementary Material for

Strengthening Discrete-Time Scheduling Formulations by Introducing the Concept of Campaigns

Diego C. Cafaro[a] & Ignacio E. Grossmann[b]

[a] *INTEC(UNL-CONICET), Güemes 3450, 3000 Santa Fe, ARGENTINA*

[b] *Dept. of Chem. Eng., Carnegie Mellon University, 5000 Forbes Ave, 15213 Pittsburgh, PA*

Generalized Formulations

The following formulations are generalizations of the models presented in the main text, accounting for any job duration d_i, time horizon length T, and time-dependent changeover costs $c_{i,t}$.

Discrete-Time Conventional Formulation

$$Min \qquad TC = \sum_{i=A,B} \sum_{t=1}^{T} c_{i,t} \cdot x_{i,t} \tag{S1}$$

$$s.t. \qquad \sum_{t=1}^{T} y_{i,t} = n_i \qquad \forall i = A, B \tag{S2}$$

$$\sum_{i=A,B} \sum_{t'=t-d_i+1}^{t} y_{i,t'} \leq 1 \qquad \forall t = 1 \dots T \tag{S3}$$

$$\sum_{i=A,B} \sum_{t=T-d_i+2}^{T} y_{i,t} = 0 \tag{S4}$$

$$x_{i,t} \geq y_{i,t} - y_{i,t-d_i} \qquad \forall i = A, B;\ t = 1 \dots T \tag{S5}$$

Campaign-Based Formulation

$$Min \qquad TC = \sum_{i=A,B} \sum_{t=1}^{T} c_{i,t} \cdot x_{i,t} \tag{S6}$$

$$s.t. \qquad \sum_{t=1}^{T} \sum_{r=1}^{n_i} r \cdot y_{i,t,r} = n_i \qquad \forall i = A, B \tag{S7}$$

$$\sum_{i=A,B} \sum_{r=1}^{n_i} \sum_{t'=t-d_i \cdot r+1}^{t} y_{i,t',r} \leq 1 \qquad \forall t = 1 \dots T \tag{S8}$$

$$\sum_{i=A,B} \sum_{r=1}^{n_i} \sum_{t=T-d_i \cdot r+2}^{T} y_{i,t,r} = 0 \tag{S9}$$

$$x_{i,t} = \sum_{r=1}^{n_i} y_{i,t,r} \qquad \forall i = A, B; t = 1 \dots T \tag{S10}$$

Computational Experiments

We present the results for a more complex example in which the duration of the tasks and the number of jobs vary with the job class. Furthermore, changeover costs are time-dependent and idle times are also allowed. Table S1 presents the number of jobs per class and the jobs duration for all the problem instances. Note that the total time required to process all the jobs is 180 time periods.

Table S1. Job classes, number of jobs per class, and job duration

Job Class	*p1*	*p2*	*p3*	*p4*	*p5*	*p6*	*p7*	*p8*	*p9*	*p10*
No of Jobs per Class	4	6	4	6	4	6	4	6	4	6
Job Duration (periods)	2	4	5	2	4	5	2	4	5	3

Changeover costs depend on both the job class and the time period when a new campaign starts. To determine the value of the changeover cost $c_{i,t}$ we use the equation given in (S11).

$$c_{i,t} = \alpha_i + \beta_i \cdot t + \gamma_i \cdot t^2 + \delta_i \cdot t^{0.5} + \varepsilon_i \cdot \sin(t \cdot \pi/4) \qquad \forall i, t \tag{S11}$$

Table S2. Parameters determining the changeover costs for different job classes at different time periods

Job Class	*p1*	*p2*	*p3*	*p4*	*p5*	*p6*	*p7*	*p8*	*p9*	*p10*
α_i	0	0	3	20	0	20	10	5	10	10
β_i	0.1	0.1	0	-0.1	0	0	0.1	0	-0.1	0
γ_i	0	0.001	0	0.001	0	0	-0.0001	0	0.001	0
δ_i	0	0	0	0	1	-1	-1	0.5	-0.5	0
ε_i	0	0	0	0	0	0	0	0	0	5

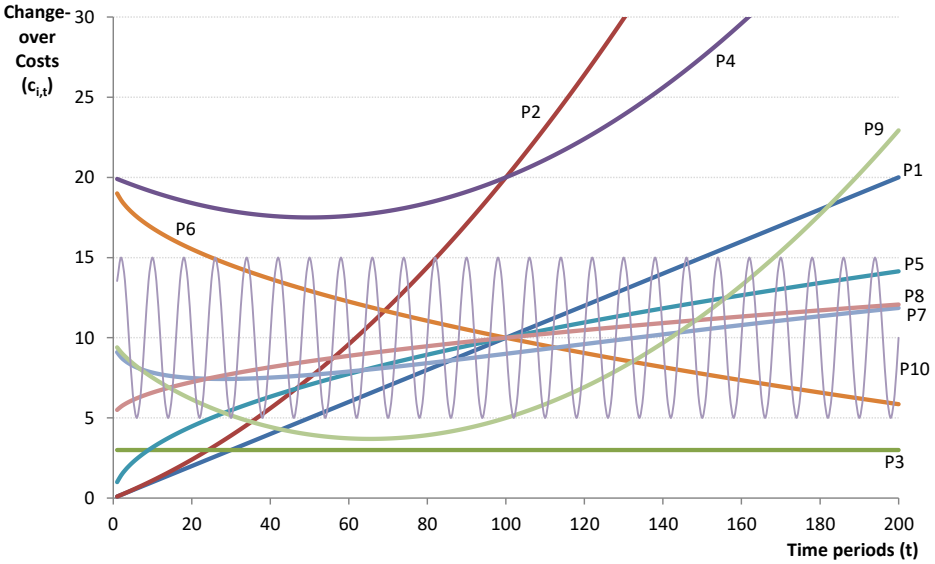

Figure S1. Changeover costs over time for different job classes in the case study

Values of coefficients α_i, β_i, γ_i, δ_i, ε_i for all problem instances are given in Table S2. Note that constant, increasing, decreasing, convex, concave and oscillating cost functions are all accounted for, as shown in Figure S1.

We solve this example using both the conventional and the proposed campaign-based formulation, with different time horizon lengths. In Table S3 we summarize the computational results (model sizes, number of iterations and CPU times), while Table S4 presents the optimal solutions found (0% optimality gap). We note that all the solutions yielded by the proposed approach are found at the relaxed LP node (0% integrality gap), while conventional formulations average 89.2% of integrality gap.

Table S3. Model statistics and computational performance of both models for different instances

Time Horizon Length	*Conventional Model*					*Campaign-Based Model*				
	Eqs.	*Vars.*	*0-1 Vars.*	*CPU Time (s)*	*Iter. x10^3*	*Eqs.*	*Vars.*	*0-1 Vars.*	*CPU Time (s)*	*Iter. x10^3*
180	1,992	3,601	1,800	5287.2	21,492	1,992	10,801	9,000	0.562	3.8
190	2,102	3,801	1,900	310.4	1,446	2,102	11,401	9,500	0.405	3.7
200	2,212	4,001	2,000	187.8	885	2,212	12,001	10,000	0.312	3.6

Table S4. Optimal solution of every problem instance

Time Horizon Length	*Jobs in Campaign / Campaign Starting Period*										*Min TC*
	p1	*p2*	*p3*	*p4*	*p5*	*p6*	*p7*	*p8*	*p9*	*p10*	
180	4 / 17	6 / 25	4 / 113	6 / 57	4 / 1	6 / 151	4 / 49	6 / 89	4 / 69	6 / 133	61.634
190	4 / 17	6 / 25	4 / 114	6 / 57	4 / 1	6 / 161	4 / 49	6 / 89	4 / 69	6 / 134	59.770
200	4 / 17	6 / 25	4 / 114	6 / 57	4 / 1	6 / 171	4 / 49	6 / 89	4 / 69	6 / 150	59.382