BATCH-CENTRIC CONTINUOUS-TIME FORMULATION FOR PIPELINE SCHEDULING

Pedro M. Castro*[,a], and Hossein Mostafaei[a,b]
[a]CMAFCIO, Faculdade de Ciências, Universidade de Lisboa
1749-016 Lisboa, Portugal
[b]Department of Applied Mathematics, Azarbaijan Shahid Madani University
Tabriz, Iran

Abstract

We propose a new mixed-integer linear programming (MILP) formulation for straight pipelines with multiple intermediate dual purpose nodes. Products enter the pipeline as batches, making this a batch-centric approach. As such, and before solving the model, it is required to convert the products initially inside the pipeline into batches and assign them left and right coordinates. Furthermore, we need to leave empty batches in between to allow for injections at intermediate nodes. We will show that these decisions, together with the number of time slots in the single grid continuous-time formulation, affect solution quality. The model features segment-dependent coordinates and allows for interacting pumping runs. It is thus more general than previous work, leading to a better utilization of the pipeline capacity.

Keywords

Petroleum industry, mathematical modeling, optimization, mixed-integer linear programming.

Introduction

Pipelines are frequently used to send refined petroleum products over long distances, from refineries to distribution centers. Different configurations can be encountered, ranging from a single pipeline with a refinery feeding a depot at the other end, to tree- and mesh-like structures. The flow is unidirectional in most cases, but systems with reversible flow can also be found.

Planning and scheduling of multiproduct pipelines can be quite challenging since the liquid fuels will typically take different routes, a batch may increase/decrease in size while passing through input/output nodes, pipeline segments vary in diameter (leading to changes in the preferred flowrate), some product sequences are forbidden, etc. This has motivated researchers to developed optimization approaches and a few contributions have appeared over the last decade. They can be divided into product and batch centric.

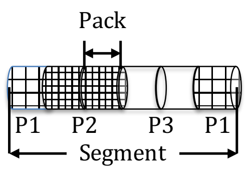

Figure 1. Pipeline segment discretized into packs of known capacity (single product/pack)

The main modeling difficulty concerns allowing a product to be present in different places of a segment with other products in between, e.g. P1 in Figure 1. One way to overcome it, is to rely on a discrete volume representation (Rejowski and Pinto, 2003, 2008; Herran et al., 2010), where each pack holds exactly one product. Discretizing the volume increases model size, affects solution quality and may degrade computational performance. The alternative is

* To whom all correspondence should be addressed (pmcastro@fc.ul.pt)

to use a continuous volume representation. An example, is the Resource-Task Network (RTN) model of Castro (2010), which is also product centric but may return suboptimal solutions since it does not allow for the case in Figure 1.

Batch centric approaches overcome such problem while using a continuous volume representation. The first is due to Cafaro and Cerdá (2004) and tackled a system with a single refinery and multiple output nodes. Products initially in the pipeline are converted into batches with the modeler postulating additional elements so as to meet product demand. It is a simple process that becomes more complex in systems with multiple input nodes. More specifically, Cafaro and Cerdá (2009) highlight the importance of defining empty batches as part of the initial characterization of the pipeline. However, to the best of our knowledge, no work has made such decisions part of the optimization. One of the goals of this paper is to motivate researchers to do so, by testing alternative assignments.

The straight pipeline configuration considered in this paper was also tackled by Cafaro and Cerdá (2010) and Cafaro et al. (2015) but they used a 2-stage decomposition approach to generate the schedule. As a consequence, it is not possible to enforce flowrate constrains on the segments. The higher planning level handles decisions involving the sequence of product injections and the destination for each batch. The lower scheduling level then finds the sequence and timing of product deliveries. In contrast, the models in Ghaffari-Hadigheh and Mostafaei (2015) and Mostafaei et al. (2015, 2016) generate the detailed schedule in one step, leading to improved solution quality. By allowing a segment to receive a product both from its input node and the immediate upstream segment (interacting pumping runs), the model discussed in this work can do even better. The comparison below, considers makespan minimization to better highlight the more efficient use of the pipeline capacity. Different objectives can naturally be included.

Problem Statement

We consider a straight pipeline with multiple intermediate single or dual purpose nodes, see Figure 2. A segment $s \in S$ identifies the part of the pipeline located between consecutive nodes, of volume v_s^S (m^3). The location of input nodes $r \in R$, output nodes $d \in D$ and dual purpose nodes $dp \in DP$ is known. More specifically, subset R_s, if $\neq \emptyset$, indicates the refinery at the start of segment s, while subset D_s holds the depot at the end of s.

Lower and upper bounds on the aggregated storage capacity are known for each product $p \in P$, e.g. $v_{r,p}^{R,\min}$ and $v_{r,p}^{R,\max}$, and so are the initial volumes in storage, e.g. $v_{d,p}^{D,0}$. For simplicity, it is assumed that the initial volume at the refineries is enough to meet the product demand at the depots $f_{d,p}^{D,\text{end}}$, which is removed all at once at the end of the last pumping run. Additional data includes minimum and maximum pumping rates, which are product specific for nodes, e.g. $\rho_{d,p}^{D,\min}$, but not for segments, e.g. $\rho_s^{S,\min}$.

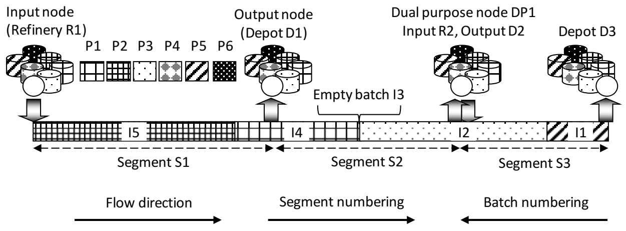

Figure 2. Straight pipeline with multiple intermediate nodes

Assigning initial products to batches

The initial product sequence and volumes inside the pipeline are also known. However, this is not enough to proceed, since batch-centric models require the assignment of each of these products to one or more batches $i \in I$. Take Figure 2 as an example. There are 4 products in the pipeline, arranged, from right to left, in a P5-P3-P1-P2 sequence. The standard approach is to consider each position in the sequence as a batch and so, the number of batches initially in the pipeline, I^{old}, is set to four. It does not necessarily mean that the old batches will go from one to four since we may need empty batches in between to allow for new product injections at intermediate nodes. More specifically, empty batch I3 allows input node R2 (part of dual purpose node DP1) to pump another product between P3 (I2) and P4 (I3). Empty batches will become new batches I^{new} during operation. Thus, for $|I|$=7, I^{old}={I1,I2,I4,I5} and I^{new}={I3,I6,I7}.

The batch-product assignment for old batches involves making parameter $y_{i,p}$=1. The initial volumes inside the pipeline segments are given by $v_{s,i}^{S,0}$ (m^3), with the right-coordinates $rc_{s,i}^0$ defining their exact location, see Figure 3. In contrast, the batch-product assignment for new batches will be determined by the optimization (variables $Y_{i,p}$).

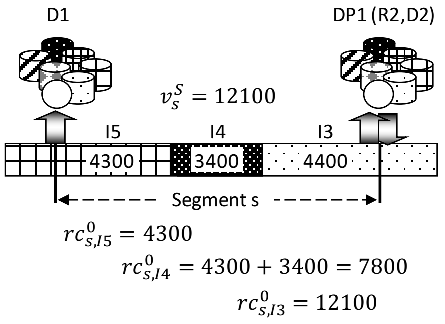

Figure 3. Computing initial right coordinates

Time Representation

One key element of scheduling formulations is the underlying time representation. The modeler has to decide between discrete and continuous, based on experience. For the specific case of pipeline scheduling, the continuous-time model of Cafaro and Cerdá (2004) improved performance by up to 3 orders of magnitude compared to the discrete-time and -volume model of Rejowski and Pinto (2003). A fairer comparison involving the same RTN continuous-volume model also favored continuous time (Castro, 2010) and so, it is the option taken.

Batch-centric models use the notion of pumping runs. They can be viewed as the time slots of a continuous-time representation relying on a single grid, which was also used by the product-centric formulation of Castro (2010).

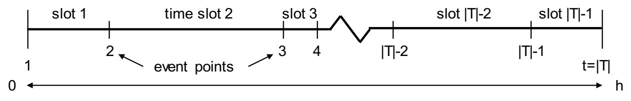

Figure 4. Underlying continuous-time grid

The adopted reference grid is show in Figure 4. There are $t \in T$ event points ($|T|$-1 time slots) ranging from time zero to the given horizon h. Let T_t give the absolute time of event point t and L_t the duration of slot t (h). The objective will be to minimize the makespan, the time of the last event point, Eq. (1). The difference in time of two consecutive event points equals the slot duration, Eq. (2).

$$\min T_{|T|} \tag{1}$$

$$T_{t+1} = T_t + L_t \ \forall t \neq |T| \tag{2}$$

Decisions affecting solution quality

The number of event points $|T|$, number of batches $|I|$ and product-batch assignments for old batches, all affect solution quality. It reflects the higher complexity of pipeline scheduling compared to other process scheduling problems, which only need to be iterated over $|T|$ to find the real optimal solution.

Rather than performing an exhaustive search over all possible combinations, we do the following: (i) choose the number of batches and initial assignments based on product demand; (ii) iterate over $|T|$, starting with a low number and stopping when the makespan stops improving; (iii) increase the number of batches by one and/or change the location of empty batches to see if a better solution can be found. Since this is a heuristic search procedure, it is perfectly possible that the reported solutions are not globally optimal.

Pipeline Model

The mathematical formulation to be presented next is divided into modules. The pipeline system features input and output nodes (dual purpose nodes have one input and one output node), and pipeline segments, see Figure 5. Nodes are product centric since all its model parameters are related to products. Nevertheless, the volumetric balances feature disaggregated variables with a batch index to make the connection to the immediate segment. On the other hand, the segment module is batch centric. Batches can be viewed as virtual entities that facilitate the writing of some model constraints. Ideally, the complete model should be product centric to forbid certain product sequences.

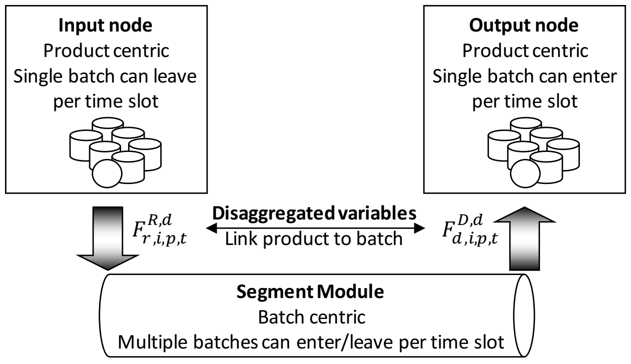

Figure 5. Main elements of pipeline system

It is important to highlight at this point that the complete mixed-integer linear programming (MILP) model was derived from Generalized Disjunctive Programming (GDP) (Balas, 1979; Raman and Grossmann, 1994; Castro and Grossmann, 2012). The timing constraints are of the big-M type since a slightly better performance was observed compared to their convex hull counterparts. Further details are given in Mostafaei and Castro (2017).

Batch-product assignment

The linking variables in Figure 5 can be different than zero only if batch i is associated to product p, i.e. $Y_{i,p}$=1. In Eqs. (3)-(4), $f_p^{P,\max}$ is an upper bound on the maximum volume that can be transferred. Batch-product assignments are already known for old batches, while new batches have to select exactly one product, Eqs. (5)-(6).

$$\sum_r \sum_{t\neq|T|} F^{R,d}_{r,i,p,t} \leq f_p^{P,\max} Y_{i,p} \ \forall i,p \tag{3}$$

$$\sum_d \sum_{t\neq|T|} F^{D,d}_{d,i,p,t} \leq f_p^{P,\max} Y_{i,p} \ \forall i,p \tag{4}$$

$$Y_{i,p} = y_{i,p} \ \forall i \in I^{old}, p \tag{5}$$

$$\sum_p Y_{i,p} = 1 \ \forall i \in I^{new} \tag{6}$$

Input node

The mass balance in Eq. (7) states that the volume $V^R_{r,p,t}$ at event point t is equal to the volume at the previous event point (or the initial capacity $v^{R,0}_{r,p}$) minus the volume entering the pipeline during slot t-1. Eq. (8) enforces lower and upper limits on the storage capacity. The volume of batch i entering the pipeline is equal to the sum of the disaggregated product variables, see Eq. (9).

$$V^R_{r,p,t} = v^{R,0}_{r,p}\big|_{t=1} + V^R_{r,p,t-1} - \sum_{i\in I_r} F^{R,d}_{r,i,p,t-1} \ \forall r,p,t \tag{7}$$

$$v_{r,p}^{R,\min} \leq V_{r,p,t}^{R} \leq v_{r,p}^{R,\max}\ \forall r,p,t \tag{8}$$

$$F_{r,i,t}^{R} = \sum_p F_{r,i,p,t}^{R,d}\ \forall r,i,t \neq |T| \tag{9}$$

Let binary variable $X_{r,i,t}^{R}$ indicate if input node r is pumping batch i during slot t. As highlighted in Figure 5, at most one batch can leave the refinery during slot t, see Eq. (10). If $X_{r,i,t}^{R}$=1, then $F_{r,i,t}^{R}$≥0, else $F_{r,i,t}^{R}$=0 (Eq. (11), where $f_r^{R,\min}$ and $f_r^{R,\max}$ are bounding parameters). In addition, the duration of the pumping run must respect the minimum and maximum flowrates, see big-M Eq. (12).

$$\sum_i X_{r,i,t}^{R} \leq 1\ \forall r,t \neq |T| \tag{10}$$

$$f_r^{R,\min} X_{r,i,t}^{R} \leq F_{r,i,t}^{R} \leq f_r^{R,\max} X_{r,i,t}^{R}\ \forall r,i,t \neq |T| \tag{11}$$

$$\sum_i \sum_p \frac{F_{r,i,p,t}^{R,d}}{\rho_{r,p}^{R,\max}} \leq L_t \leq \sum_i \sum_p \frac{F_{r,i,p,t}^{R,d}}{\rho_{r,p}^{R,\min}} + h \cdot \left(1 - \sum_{i \in I_r} X_{r,i,t}^{R}\right)\ \forall r,t \neq |T| \tag{12}$$

Output node

Similar constraints can be obtained for the output node. The main difference is that the volume is now entering the output node and there is an instantaneous product removal at the end of the time horizon that represents the product demand, see Eq. (13).

$$V_{d,p,t}^{D} = v_{d,p}^{D,0}\Big|_{t=1} + V_{d,p,t-1}^{D} + \sum_i F_{d,i,p,t-1}^{D,d} - f_{d,p}^{D,\text{end}}\Big|_{t=|T|}\ \forall d,p,t \tag{13}$$

Dual purpose node

Dual purpose node dp can act simultaneously as input and output node provided that the same batch is not involved. In Eq. (14), subsets R_{dp} and D_{dp} hold the refinery and depot associated to the node. Equation (15) ensures that product arrivals and departures report to the same tank.

$$X_{r,i,t}^{R} + X_{d,i,t}^{D} \leq 1\ \forall dp, r \in R_{dp}, d \in D_{dp}, i, t \neq |T| \tag{14}$$

$$V_{dp,p,t}^{DP} = v_{dp,p}^{DP,0}\Big|_{t=1} + V_{dp,p,t-1}^{DP} - \sum_d f_{d,p}^{D,\text{end}}\Big|_{t=|T|} + \sum_i\left(\sum_d F_{d,i,p,t-1}^{D,d} - \sum_r F_{r,i,p,t-1}^{R,d}\right)\ \forall dp,p,t \tag{15}$$

Node-segment junction

According to Figure 2, nodes are located between consecutive segments. Equation (16) ensures that the volume $F_{s,i,t}^{S,in}$ of batch i entering segment s during slot t must be equal to the volume leaving the previous segment plus the volume coming from the refinery node minus the volume entering the depot. Then, the volume leaving the last segment must enter the last depot, Eq. (17). The model allows for the input node at the start of segment s and segment s-1 to simultaneous send material to s but only if the same batch is involved. Similarly, if batch i is entering output node d, only i can be leaving segment s feeding d.

$$F_{s,i,t}^{S,in} = F_{s-1,i,t}^{S,out} + \sum_{r \in R_s} F_{r,i,t}^{R} - \sum_{d \in D_{s-1}} F_{d,i,t}^{D}\ \forall s,i,t \tag{16}$$

$$F_{s,i,t}^{S,out} = \sum_{d \in D_s} F_{d,i,t}^{D}\ \forall s = |S|, i, t \neq |T| \tag{17}$$

$$\sum_i X_{s,i,t}^{S,in} \leq 1 + (|I| - 1) \cdot \left(1 - \sum_i X_{r,i,t}^{R}\right)\ \forall s, r \in R_s, t \tag{18}$$

$$\sum_i X_{s,i,t}^{S,out} \leq 1 + (|I| - 1) \cdot \left(1 - \sum_i X_{d,i,t}^{D}\right)\ \forall s, d \in D_s, t \tag{19}$$

Segment

The volumetric balance for batch i in segment s is shown in Eq. (20). The liquid fuels are incompressible and so the sum of the batch volumes inside must match the segment volume, Eq. (21). Events triggering the entrance and withdrawal of batches into/from the segment are related to their left $LC_{s,i,t}$ and right $RC_{s,i,t}$ coordinates. These are related by Eqs. (22)-(23) and illustrated in Figure 3.

$$V_{s,i,t}^{S} = v_{s,i}^{S,0}\Big|_{t=1} + V_{s,i,t-1}^{S} + F_{s,i,t-1}^{S,in} - F_{s,i,t-1}^{S,out}\ \forall s,i,t \tag{20}$$

$$\sum_i V_{s,i,t}^{S} = v_s^{S}\ \forall s,t \tag{21}$$

$$RC_{s,i,t} = rc_{s,i}^{0}\Big|_{t=1} + \left(\sum_{i' \geq i} V_{s,i',t}^{S}\right)\Big|_{t>1}\ \forall s,i,t \tag{22}$$

$$LC_{s,i,t} = RC_{s,i,t} - V_{s,i,t}^{S}\ \forall s,i,t \tag{23}$$

Let binary variable $X_{s,i,t}^{S,in}$ indicate if batch i is entering segment s during slot t and $X_{s,t}^{S,no\ i}$ if no batch is entering. Clearly, at most one of the two cases can happen, see Eq. (24). If the batch is entering the segment, then its left coordinate must be equal to zero, otherwise it can take any value lower than the volume segment, see Eq. (25). The bounding and timing constraints are in Eqs. (26)-(27).

$$X_{s,i,t}^{S,in} + X_{s,t}^{S,no\ i} \leq 1\ \forall s,i,t \neq |T| \tag{24}$$

$$LC_{s,i,t} \leq v_s^{S} \cdot \left(1 - X_{s,i,t}^{S,in}\right)\ \forall s,i,t \neq |T| \tag{25}$$

$$f_s^{S,\min} X_{s,i,t}^{S,in} \leq F_{s,i,t}^{S,in} \leq f_s^{S,\max} X_{s,i,t}^{S,in}\ \forall s,i,t \neq |T| \tag{26}$$

$$\frac{\sum_{i \in I_s} F_{s,i,t}^{S,in}}{\rho_s^{S,\max}} \leq L_t \leq \frac{\sum_{i \in I_s} F_{s,i,t}^{S,in}}{\rho_s^{S,\min}} + h \cdot X_{s,t}^{S,no\ i}\ \forall s,t \neq |T| \tag{27}$$

For a batch to leave segment s during slot t, the right coordinate at the end of the slot (event point t+1) must be equal to the segment volume, see Eqs. (28)-(29).

$$RC_{s,i,t+1} \geq v_s^{S} X_{s,i,t}^{S,out}\ \forall s,i,t \neq |T| \tag{28}$$

$$f_s^{S,\min} X_{s,i,t}^{S,out} \leq F_{s,i,t}^{S,out} \leq f_s^{S,\max} X_{s,i,t}^{S,out}\ \forall s,i,t \neq |T| \tag{29}$$

Computational Results

The new formulation has been implemented in GAMS 24.6.1 and solved using CPLEX 12.6.3 running in parallel deterministic mode using up to eight threads. The termination criteria were either a relative optimality tolerance of 10^{-6} or a maximum wall time limit of 7200 CPUs. The hardware consisted of a Windows 10, 64-bit desktop with an Intel i7-6700K (4.0 GHz) processor and 16 GB of RAM.

We consider three example problems, Ex1-Ex3, taken respectively from Mostafaei et al. (2016) (Examples 4 and 2) and Cafaro and Cerdá (2010) (Example 2). Compared to our previous model in Mostafaei et al. (2016), we are now able to reduce the makespan (see Table 1) and the number of slots that ensures feasibility. Two novel features in our model are responsible for this: (i) the new model allows for interacting pumping runs (IPR), in which a segment of the pipeline simultaneous receives material from its input node and upstream segment; (ii) left and right coordinates of batches are no longer global, but segment dependent.

Table 1. Optimal makespan (h) assuming single empty batch

Example	Mostafaei et al. (2016)	This work	Reduction
Ex1	207.62	197.44	4.9%
Ex2	143.65	136.90	4.7%
Ex3	237.20	231.97	2.2%

Table 2. Computational statistics for new model

Example	$\lvert I\rvert$	$\lvert T\rvert$	Makespan (h)	CPUs
Ex1	8	10	Infeasible	172
		11	215.81	235
		12	203.06	403
		13	197.44	675
		14	203.06	*7200*[a]
Ex2	8	7	Infeasible	72.1
		8	140.47	175
		9	138.10	559
		10	136.90	1269
Ex3	9	10	Infeasible	1911
		11	237.60	3794
		12	231.97	1786

[a] Optimality gap at termination=8.6%

Table 2 shows the computational results for the individual iterations in the search for the global optimal solution. The number of batches $|I|$ is fixed, while single increments are adopted for the number of event points $|T|$. It can be seen that solution quality typically improves, whereas the computational time increases. The exceptions occur for: (i) Ex1 for $|T|$=14, which cannot be solved to optimality in the given time; (ii) Ex3 for $|T|$=12, which is faster than $|T|$=11 because the optimal solution of 231.97 h becomes equal to the LP relaxation (zero integrality gap). Note that the integrality gap reduces when allowing IPR.

Locating empty batches

For a given number of batches and event points, there are still degrees of freedom left, linked to the location of the empty batches. Assuming a single empty batch, we show in Table 3 the base case for Ex1-Ex3 (results in Table 2) together with the alternatives in the rows below. I3 is the empty batch in the base case of Ex1, changing then to I4 and I2. I3 is actually the only option that ensures feasibility.

Picking I4 as the empty batch for Ex2, also leads to a feasible solution for $|T|$=10, but the 137.60 h makespan is 0.6% higher. Interestingly, the relative difference to the base case is actually zero for $|T|$=9 (138.10 h) and 1.7% for $|T|$=8 (142.86 h), suggesting that the best assignment might change with the number of events. In contrast, moving the empty batch to I2 makes the problem infeasible.

The results for Ex3 show that there is also a significant impact on computational time. While the 237.60 h makespan for the base case and $|T|$=11 is proven optimal in roughly one hour, two hours are not enough when moving the empty batch from I3 to I2, I4 or I5. For I6, we don't even know if the problem is feasible.

Table 3. Results for different assignments of old batches (with single empty batch)

Example	I^{old}	$\lvert T\rvert$	Makespan (h)	Change
Ex1	I1,I2,I4	11	215.81	-
	I1,I2,I3	11,12	Infeasible	-
	I1,I3,I4	11,12	Infeasible	-
Ex2	I1,I2,I4	10	136.90	-
	I1,I2,I3	10	137.70	+0.6%
	I1,I3,I4	8,9	Infeasible	-
Ex3	I1,I2,I4,I5,I6	11	237.60	-
	I1,I3,I4,I5,I6	11	240.19[a]	+1.1%
	I1,I2,I3,I5,I6	11	238.13[a]	+0.2%
	I1,I2,I3,I4,I6	11	238.13[a]	+0.2%
	I1,I2,I3,I4,I5	11	No solution[a]	-

[a] Up to maximum resource limit of 7200 CPUs

Let us now increase the number of empty batches to two while focusing on Ex2 and $|T|$=10. If the number of batches stays at eight, then the problem is infeasible (found in just 6.54 CPUs). However, raising to $|I|$=9, lowers the makespan by 1.6% to 134.72 h (4823 CPUs) compared to Table 2. The optimal schedule is given in Figure 6 and three aspects are worth highlighting.

The first, is that segments are almost always operating, except for S1 in [16.67, 30.95] and S2 in [66.67, 75.00]. The second is the advantage of segment-dependent coordinates, observed in the last two runs. Notice in [109.72, 122.22] that batch I8 is being pumped from R4 into the last segment (S4), while I6 is still in S3. This is allowed since I6 (and I7) will never enter S4. It is also possible for I8 to enter segment

S3 in [122.22, 134.72] while I7 is still being removed from S2. The third aspect is that the inlet and outlet flows from DP1 in interval [0, 16.67] both involve P3 but do not conflict with Eq. (14) since they contain batches I6 and I4.

Conclusions

This paper has presented a new continuous-time formulation for the scheduling of straight pipelines with multiple intermediate nodes. It is batch centric, in the sense the user needs to convert the products initially in the pipeline into batches, while the optimization assigns batches to pumping runs. We have seen that the initial assignments, the number of batches and the number of event points in the time grid, all affect solution quality. Future work will thus look into including the initial decisions as part of the optimization, as well as extending the model to other pipeline configurations. The results have also shown that the new model is more general than previous work.

Acknowledgments

Financial support from Fundação para a Ciência e Tecnologia (UID/MAT/04561/2013 and Investigador FCT 2013) and Iranian Ministry of Science and Technology.

References

Balas, E. (1979). Disjunctive programming. *Annals of Discrete Mathematics*, 5, 3.

Cafaro, V.G., Cafaro, D.C., Mendéz, C.A., Cerdá, J. (2015). Optimization model for the detailed scheduling of multi-source pipelines. *Comput. Ind. Eng.*, 88, 395.

Cafaro, D.C., Cerdá, J. (2004). Optimal scheduling of multiproduct pipeline systems using a non-discrete MILP formulation. *Comput. Chem. Eng.*, 28, 2053.

Cafaro, D.C., Cerdá, J. (2009). Optimal scheduling of refined products pipelines with multiple sources. *Ind. Eng. Chem. Res.*, 48, 6675.

Cafaro, D.C., Cerdá, J. (2010). Operational scheduling of refined products pipeline networks with simultaneous batch injections. *Comput. Chem. Eng.*, 34, 1687.

Castro, P.M. (2010). Optimal scheduling of pipeline systems with a resource-task network continuous- time formulation. *Ind. Eng. Chem. Res.*, 49, 11491.

Castro, P.M., Grossmann, I.E. (2012). Generalized disjunctive programming as a systematic modeling framework to derive scheduling formulations. *Ind. Eng. Chem. Res.*, 51, 5781.

Ghaffari-Hadigheh, A., Mostafaei, H. (2015). On the scheduling of real world multiproduct pipelines with simultaneous delivery. *Optimization and Engineering*, 16, 571.

Herran, A., Cruz, J.M., Andres, B. (2010). Mathematical model for planning transportation of multiple petroleum products in a multi-pipeline system. *Comput. Chem. Eng.*, 34, 401.

Mostafaei, H., Castro, P.M. (2017). Continuous-time scheduling formulation for straight pipelines. AIChE J. doi: 10.1002/aic.15563.

Mostafaei, H., Castro, P.M., Ghaffari-Hadigheh, A. (2015). A novel monolithic MILP framework for lot- sizing and scheduling of multiproduct treelike pipeline networks. *Ind. Eng. Chem. Res.*, 54, 9202.

Mostafaei, H., Castro, P.M., Ghaffari-Hadigheh, A. (2016). Short-term scheduling of multiple source pipelines with simultaneous injections and deliveries. *Comput. Oper. Res.*, 73, 27.

Raman, R., Grossmann, I.E. (1994). Modeling and computational techniques for logic based integer programming. *Comput. Chem. Eng.*, 18, 563.

Rejowski, R., Pinto, J.M. (2003). Scheduling of a multiproduct pipeline system. *Comput. Chem. Eng.*, 27, 1229.

Rejowski, R., Pinto, J.M. (2008). A novel continuous time representation for the scheduling of pipeline systems with pumping yield rate constraints. *Comput. Chem. Eng.*, 32, 1042.

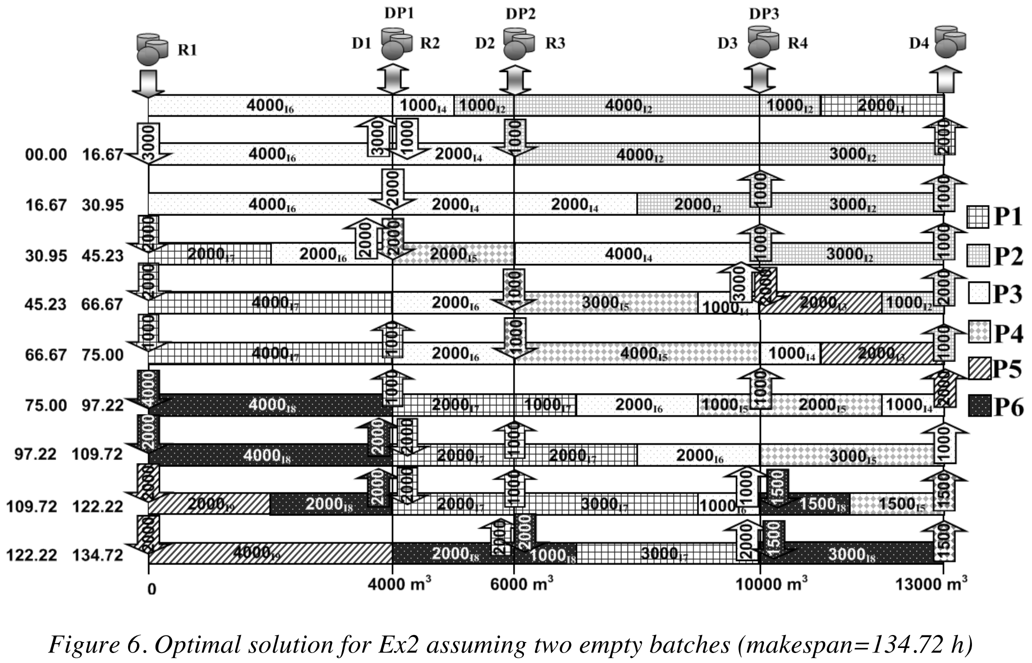

Figure 6. Optimal solution for Ex2 assuming two empty batches (makespan=134.72 h)