Planning of multi-product pipelines by economic lot scheduling models

Thomas Kirschstein

PII: S0377-2217(17)30532-5
DOI: 10.1016/j.ejor.2017.06.014
Reference: EOR 14496

To appear in: *European Journal of Operational Research*

Received date: 4 August 2016
Revised date: 13 April 2017
Accepted date: 5 June 2017

Please cite this article as: Thomas Kirschstein, Planning of multi-product pipelines by economic lot scheduling models, *European Journal of Operational Research* (2017), doi: 10.1016/j.ejor.2017.06.014

Highlights

- First study on multi-product pipeline scheduling with transition technologies.
- Modeling of product transitions by interfaces and physical product separation.
- Variant of Economic Lot Scheduling Problem applied for pipeline scheduling. Proposal of a heuristic for pipeline scheduling with batch size limits.

Planning of multi-product pipelines by economic lot scheduling models

Thomas Kirschstein[a,*]

[a]*School of Economics and Business, Martin-Luther-University, Gr. Steinstr. 73, 06108 Halle, Germany*

Abstract

In chemical and petroleum industry pipelines are one of the most important means of transportation. However, flexibility of pipeline transport systems is limited by many restrictions. Therefore, the planning of pipeline operations is a crucial part of logistics management in these industries. A particularly challenging problem is the pipeline scheduling which is concerned with finding the sequences, times, and sizes of batch injections in pipeline systems. This paper specifically studies the underlying core scheduling problem by assuming a simple multi-product pipeline system. It is shown that finding a sequence of batches which minimizes stock holding and setup costs in the long run is an $\mathcal{NP}$-hard scheduling problem, namely a variant of the economic lot scheduling problem (ELSP) with additional constraints. Therefore, a powerful heuristic for the sequence-dependent ELSP is adapted and extended to meet the requirements of the outlined pipeline scheduling problem. The application of the heuristic is illustrated by case studies from chemical and petroleum industry.

Keywords: scheduling; economic lot scheduling problem; pipeline management

1. Introduction

Pipelines are one of the most efficient modes transport w.r.t. energy consumption and, hence, variable transportation cost (van Essen et al., 2003). However, building pipelines incurs high investment costs. Moreover, origins and destinations are fixed once a pipeline is build and pipelines can only be used to transport liquefiable products. Hence, pipeline systems are highly inflexible and mostly used to serve unidirectional transport needs with high and steady transport demands. Traditionally, pipelines are designed for one product only, e.g. in the case of crude oil or natural gas transports. This allows optimizing the pipeline's technological configuration e.g. with respect to the energy consumption for transport. However, in chemical and petroleum industry multi-product pipelines are commonly used if the products intended to be transported are chemically similar. A necessary condition is, for example, that chemicals to be transported do not react with each other. This implies an increased flexibility, a (potentially) increased pipeline utilization and, hence, increased attractiveness of pipeline transports.

However, the more products are to be transported the more complex planning becomes. Apart from pure technical restrictions like storage or pumping capacities, also other restrictions need to be considered. This includes, for instance, safety stock requirements or pipeline inspection

*corresponding author, tel.: +49 (0) 345 5523 424, fax.: +49 (0) 345 5527 198
Email address: thomas.kirschstein@wiwi.uni-halle.de (Thomas Kirschstein)

routines. In case of multi-product pipelines, restrictions for different products interfere and increase the complexity of pipeline operations planning. Important in managing pipeline supply systems is to balance stock holding cost and transport costs as the lack of flexibility of pipeline systems is typically encountered by keeping (safety) stocks. In case of multi-product pipeline systems, additionally set-up or product-transition times as well as costs have to be considered when switching from one product to another. Hence, besides traditional trade-offs between stock holding and transportation costs, managing multi-product pipeline systems also has to cope with the sequencing of product batches on transport systems with limited capacity. In combination, this states a highly challenging planning problem which is encountered on a daily basis e.g. in petroleum and chemical industry.

In this paper we focus on a comparatively simple problem outline in order to highlight the basic complexity of managing any multi-product pipeline system. We assume a unidirectional one-to-one pipeline system for multiple products with constant demands per time unit is considered as it appears e.g. in chemical industry to supply large-scaled production sites (Kirschstein, 2015). It is shown that the optimal sequencing and scheduling of product batches w.r.t. to stock holding and setup cost can be obtained by solving a variant of the economic lot scheduling problem (ELSP, Narro Lopez and Kingsman (1991)) which has to be adapted according to the product separation technology of the pipeline system under study. As the ELSP is known to be hard to solve and available heuristics cannot be applied directly to the proposed ELSP variant, a heuristic is proposed based on the powerful ELSP heuristic published in Dobson (1992).

In the next section a literature review is provided and the current work is categorized. In section 3 mathematical formulations for objectives and technological constraints in pipeline management are derived. Afterwards, the derived formulations are used to formulate a ELSP-type optimization model. In section 5, a heuristic is described which adapts the approach of Dobson (1992). The application of the heuristic is illustrated by means of a case studies in section 6. The paper finishes with a conclusion.

2. Literature review

While one-product pipeline systems are comparatively easy to manage, multi-product supply system typically imply a serious level of complexity due to setup costs as well as a commonly used processing asset with limited capacity. Literature on pipeline planning for multi-product systems is primarily focused on specific applications in chemical or petroleum industry. Table 1 shows the literature on hand categorized according to certain criteria.

The column "sys. type" categorizes the papers according to the network structures one-to-one (1t1), one-to-many (1tm), or many-to-many (mtm), i.e. referring to the number of injection and retraction points. Column "flow" indicates whether a unidirectional or bidirectional material flow is considered. Column "scale" refers to the modeling of time and pipeline which are either sub-divided into a number of discrete slices (d) or handled continuously (c). "Objective" summarizes the aspects modeled as the planning objectives whereby p indicates pumping cost, h holding cost, b backlogging cost, i costs for interface processing, u pipeline utilization, and c product changeovers. Directly related to the objectives is column "model" where entries l and n refer to

reference	sys. type	scale	flow	objective	model	solution meth.
Magatão et al. (2004)	1t1	(d,d)	bi.	(i,c)	(l,l)	solver
Magatão et al. (2005)	1t1	(d,d)	bi.	(i,c)	(l,l)	decomp.
Magatão et al. (2011)	1t1	(c,d)	bi.	(i,c,h)	(l,l,l)	decomp.
Relvas et al. (2006)	1t1	(c,c)	uni.	(u)	(l)	solver
Relvas et al. (2009)	1t1	(c,c)	uni.	multiple	(l)	heur.
Relvas et al. (2013)	1t1	$(d,d/c)$	uni.	(u,h)	(l,l)	solver
Moradi and MirHassani (2015)	1t1	(d,c)	uni.	(h,u,i,b)	(l,l,l)	solver
Moradi and MirHassani (2016)	1t1	(d,c)	uni.	(p,i)	(l,l)	solver
Rejowski and Pinto (2003)	1tm	(d,d)	uni.	(p,h,i)	(l,l,l)	solver
Rejowski et al. (2004)	1tm	(d,d)	uni.	(p,h,i)	(l,l,l)	B+C
Rejowski and Pinto (2008)	1tm	(c,d)	uni.	(p,h,i)	(n,n,l)	solver
Cafaro and Cerdá (2004)	1tm	(c,c)	uni.	(p,h,i)	(l,l,l)	solver
Cafaro and Cerdá (2008)	1tm	(d,c)	uni.	(p,h,i,u,b)	(l,l,l)	solver
MirHassani (2008)	1tm	(d,d)	uni.	(i)	(l)	solver
MirHassani and Fani Jahromi (2011)	1tm	(c,d)	uni.	(p,h,i)	(l,l,l)	solver
Mostafaei et al. (2015)	1tm	(c,d)	uni.	(p,i,b)	(l,l,l)	solver
Moura et al. (2008)	mtm	(c,c)	bi.	—	—	CP
Cafaro and Cerdá (2009)	mtm	(c,c)	uni.	(p,b,u,i)	(l,l,l,l)	solver
Cafaro and Cerdá (2010)	mtm	(c,c)	uni.	$(u)/(p,b,i)$	$(l)/(l,l,l)$	solver
Magataõ et al. (2015)	mtm	(c,c)	bi.	(b)	(l)	decomp.
Mostafaei et al. (2016)	mtm	(c,c)	uni.	(p,i,b)	(l,l,l)	solver

Table 1: Classification of literature on pipeline scheduling

linearly or non-linearly modeled objectives, respectively. Finally, the solution methodology employed in the papers is reported in the last column where "solver" refers to commercial standard solvers, "B+C" a brach-&-cut algorithm, "heur." a heuristic approach, and "decomp." indicates a decomposition of the planning problem.

The literature review shows a heterogeneous set of papers on pipeline operations management. Historically, pipeline supply planning started with discrete models formulating time and pipeline as a sequence of slices which are passed successively. However, this type of formulation has the disadvantage that model complexity increases with increasing number of time periods (i.e. time horizon). Continuous time formulations are more flexible and can be solved for large time horizons. Hence, continuous formulations are prevalent in most recent research. Most of the literature applies the proposed models to real-world case studies and solve the models with standard solvers like CPLEX. However, there are also some heuristic and decomposition procedures which rely mainly on the decomposition into 1) batch allocation, 2) batch sizing and 3) batch scheduling.

The objectives pursued also vary heavily. In rather short-term models maximizing pipeline utilization is a sufficient proxy for minimizing pumping cost. The longer the considered time horizons are, the more cost aspects are taken into account. Most prominently, this includes stock holding costs as well as setup or interface processing costs. All papers consider a limited time horizon and a set of products with consumption and production data for each product. The final outcome of the reviewed literature is a schedule for the considered time horizon providing detailed information when and where which product is to be injected into and retracted from the pipeline system in which quantity. Hence, all papers model pipeline systems from an operational

perspective.

Table 1 shows that a lot of different approaches has been published mostly tackling challenging real-world problems. Most models are formulate as MILPs, i.e. all constraints and objectives are linearized. To solve even large problem instances mostly commercial standard solvers are used. However, particularly when additional planning parameters like pump rates are variable, tailor-made solution approaches are required whereby decomposition procedure dominate. Thus, improvements of decompositions approaches w.r.t. solution quality and run times are open issues particularly for *mtm* pipeline system (see Magataõ et al. (2015) or Mostafaei et al. (2016)). Another issue is the incorporation of uncertainty in pipeline scheduling. Except for Moradi and MirHassani (2016) considering demand uncertainty for a one-to-one pipeline, this aspect has not been addressed so far in literature. In the light of highly volatile electricity prices as well as technical and supply risks, is seems reasonable to investigate the effects of stochasticity more closely in order to obtain robust schedules. All references cited in Table 1 focus on operational scheduling problems with discrete customer orders and a rather limited time horizon of up to one month are considered. Almost all planning models are applied in a rolling horizon environment in daily operation and focus on operational objectives like maximizing pipeline utilization or minimizing backlogs. Although pursuing operational objectives leads to good or even optimal schedules in the short run, this does not necessarily imply that a repeated application of an short-term optimization models also yields optimal solutions in the long run Sethi and Sorger (1991). Thus, particularly economic effects of pipeline scheduling models in the long run are so far out of the focus in literature.

In contrast to all papers reviewed above, no explicit customer orders and an infinite planning horizon are assumed in this paper. Instead, a constant transportation demand for each product is assumed as it is e.g. encountered in chemical industry where multi-product pipelines are used to connect chemical production facilities (which are typically operated continuously). All parameters of the pipeline system are considered as static and deterministic. Thus, a tactical or even strategical perspective is taken in order to find a basic pumping sequence with lowest total cost per time unit. Here, total cost per period including stock holding and setup costs. Technical details which do not severely affect stock holding and setup cost (like settling periods, inspection turnoffs, or transition times) are excluded from modeling. On the other hand, restrictions affecting stock holding and setup cost (like pumping and tank capacities as well as separation technology) are taken into account explicitly or implicitly.

As demand rates are fixed and time-invariant, the schedule to be found matches a particular configuration of production system which is deemed to be typical (i.e. the system's steady state). I.e. a steady-state schedule is to be found which can be repeated infinitely without violating the system's restrictions. Thereby, small variations of the demand and supply rates are expected to be buffered by holding safety stocks at the head and the bottom of the pipeline. If more severe changes of environmental conditions occur (such as asset breakdowns) the steady-state schedule needs to be adjusted by means of an operational scheduling model taking into account operational details like initial stocks, initial pipeline condition, or settling periods.

3. Problem description

In the following, a one-to-one pipeline system is assumed where a set of products $\mathcal{S}$ is regularly pumped from an origin to a destination. The products are handled at the head and the bottom of pipeline in at least one production facility. For intermediate storage, tanks are available for each product which are connected to the local production facilities and the pipeline. At the pipeline's head, there is a (net) surplus of each product whereas at the pipeline's bottom there is a (net) demand for each product such that a transport demand for each product exists. The transport demand per period for each product ω_s is a constant and equals the minimum of the product's demand rate $\underline{\omega}_s$ at the pipeline's sink and its production rate $\bar{\omega}_s$ at the head of pipeline. If $\underline{\omega}_s < \bar{\omega}_s$, it makes no sense to transport more than required, i.e. $\omega_s = \underline{\omega}_s$. Otherwise, it cannot be shipped more than $\omega_s = \bar{\omega}_s$. Remaining imbalances, i.e. product surpluses at the pipeline's head or product demands at the pipeline's end need to be handled by external partners. Furthermore, it is assumed that the pipeline is either operated at a constant pump rate ρ for all products or turned off completely. Hence, $\sum_{s\in\mathcal{S}} \omega_s < \rho$ must hold for the pipeline and the ratio $\frac{\sum_{s\in\mathcal{S}} \omega_s}{\rho}$ is often referred to as the pipeline utilization. Note that in a classic ELSP, ρ equals the production rate. Additionally, the time span between injecting and receiving a certain product batch is denoted as τ. Figure 1 illustrates the outlined pipeline system schematically.

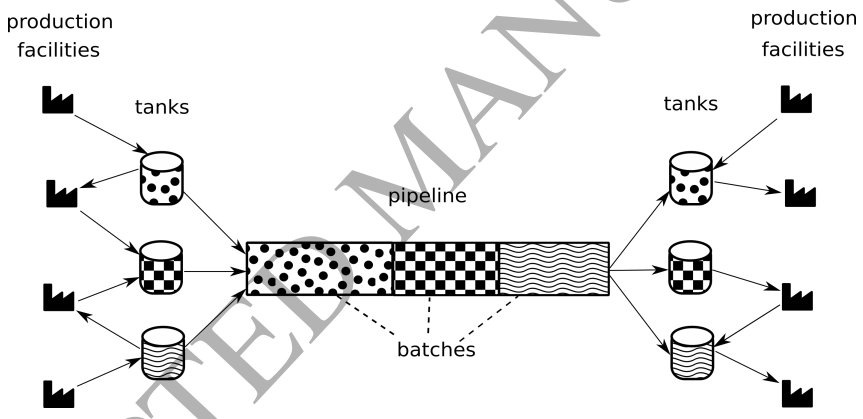

Figure 1: Schematic illustration of pipeline system under study

3.1. Storage cost modelling

For accurately calculating the storage costs of the whole pipeline system, the stock levels at origin, destination, as well as inside the pipeline have to be considered. Therefore, let T denote the cycle time between two successive batches of a certain product s. The total consumption/ production quantity is then defined by $F = T \cdot \omega_s$. The time span necessary to inject the batch is $T^{fill} = \frac{F}{\rho} = T \cdot \frac{\omega_s}{\rho}$. Hence, the stock-up time is $T_s^{stock} = T - T^{fill} = T \cdot \left(1 - \frac{\omega_s}{\rho}\right)$ and the maximum stock level L obtained in this supply system can be calculated by multiplying the stock-up time T^{stock} with the demand rate ω_s, thus, $L = \omega_s \cdot T^{stock} = \omega_s \cdot T \cdot \left(1 - \frac{\omega_s}{\rho}\right)$. Stock levels obtained inside the pipeline and at both locations are depicted in Figure 2.

Since production rates equal consumption rates, it is easy to see that the total stock in the

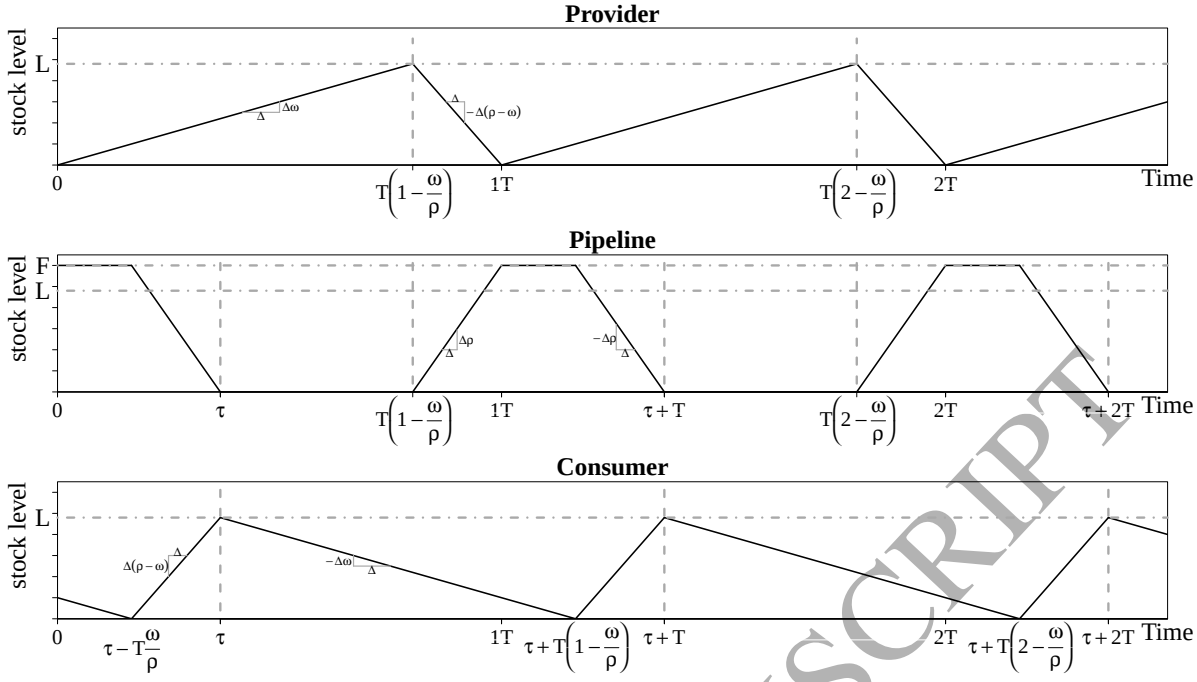

Figure 2: Inventory pattern in a batch flow system

system is constant and given by

$$\bar{L}(T) = L + \tau \cdot \omega_s = \omega_s \cdot \left(\tau + T \cdot \left(1 - \frac{\omega_s}{\rho}\right)\right). \tag{1}$$

$\bar{L}(T)$ depends only on the cycle length T. Obviously, the holding cost decrease with decreasing cycle length. I.e. for $T \to 0$ only the pipeline stock $\omega_s \cdot \tau$ is approached as the theoretical minimum for just-in-time delivery. Hence, the pipeline stock is constant and independent of the cycle time. Note that the stock level assumed in the classical ELSP is given by $\frac{L}{2}$, i.e. with a focus on one storage only.

3.2. Product changeover handling

The costs for product setups depend on the changeover handling. Basically, there are two approaches: a) A technical separation of products is applied by injecting a separation device (so-called *pipeline injection gauge*, PIG) before injecting the successive product batch or b) there is no product separation at all such that products mix inside the pipeline.

In the first case, a product setup incurs PIG injection costs which are negligible in most cases. However, once injected, each PIG arrives at the destination and must be shipped back to the origin in order to be reused for separation. I.e. a product setup incurs fixed PIG repositioning cost c^{PIG} which equals setup cost in the classic, sequence-independent ESLP.

In the second case, the products of two subsequent batches mix inside the pipeline. These product mixtures (or *interfaces*) have to be handled somehow either by reprocessing (in this case the interface is often called *transmix*, see e.g. Inkpen and Moffett (2011)) or by handling them as one of the "parental" products (typically called *downgrading*). In both cases, the quantities of the "parental" batches are affected and must be corrected accordingly in the stock balances. Let $d_{s,t}$ denote the quantity added or subtracted to/from product s. Without loss of generality, let

$d_{s,t} < 0$ indicate a loss of product s and $d_{s,t} > 0$ a surplus, respectively. Hence, for downgrading holds $sign(d_{s,t}) = -sign(d_{t,s})$ while for interface reprocessing $d_{s,t} < 0$ for all $s, t \in \mathcal{S}$. Figure 3 displays the definition graphically.

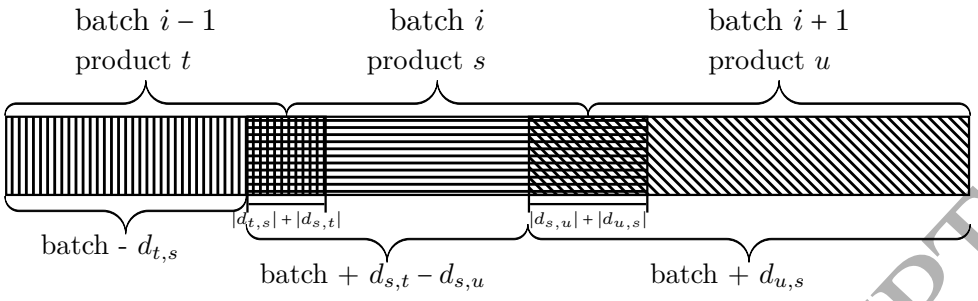

Figure 3: Illustration of interface and batch size calculation in pipeline systems with downgrading

As Figure 3 shows, the interface between product t and s is added to the batch of product s (i.e. $d_{t,s} < 0$ and $d_{s,t} > 0$). In contrast, the interface between product s and u is added to the batch of product u (i.e. $d_{u,s} > 0$ and $d_{s,u} < 0$).

No matter how the interfaces are handled, they determine setup costs. Typically, reprocessing costs are proportional to the interface quantity in the transmix case. I.e. $c^{setup}_{s,t}$ are constant and depend on $|d_{s,t} + d_{t,s}|$. In the downgrading case, the downgraded material is typically reprocessed in subsequent production processes. Hence, there are additional processing costs depending on the downgraded product quantity and the downgraded product. For any product transition $s \to t$, either product s or product t is downgraded such that $c^{setup}_{s,t}$ reflects the reprocessing cost depending on $d_{s,t}$ in this case.

For both cases of interface handling, the sequence of product batches typically matters as the interface quantity often depends on the products mixed. Hence, setup costs depend on the pumping cycle which resembles the so-called sequence-dependent ELSP (sELSP, see e.g. Dobson (1992) or Wagner and Davis (2002)). Additionally to the classic sELSP, for pipeline sequencing the stock calculations need to be adjusted accordingly. Therefore, let q^{Δ}_i denote the total batch size correction of batch i in the pumping schedule defined by

$$q^{\Delta}_i = \begin{cases} \sum_{s \in \mathcal{S}} y_{s,i} \cdot \left(\sum_{\substack{t \in \mathcal{S} \\ t \neq s}} d_{s,t} \cdot y_{t,|\mathcal{I}|} + \sum_{\substack{u \in \mathcal{S} \\ u \neq s}} d_{s,u} \cdot y_{u,(i+1)} \right) & i = 1 \\ \sum_{s \in \mathcal{S}} y_{s,i} \cdot \left(\sum_{\substack{t \in \mathcal{S} \\ t \neq s}} d_{s,t} \cdot y_{t,i-1} + \sum_{\substack{u \in \mathcal{S} \\ u \neq s}} d_{s,u} \cdot y_{u,(i+1)} \right) & i = 2, \ldots, |\mathcal{I}| - 1 \\ \sum_{s \in \mathcal{S}} y_{s,i} \cdot \left(\sum_{\substack{t \in \mathcal{S} \\ t \neq s}} d_{s,t} \cdot y_{t,i-1} + \sum_{\substack{u \in \mathcal{S} \\ u \neq s}} d_{s,u} \cdot y_{u,1} \right) & i = |\mathcal{I}| \end{cases} \quad (2)$$

where $y_{s,i}$ is a binary variable which is 1 if product s is assigned to batch i. In case of downgrading, the specific structure of $d_{s,t}$ implies that $\sum_{i \in \mathcal{I}} q^{\Delta}_i = 0$, i.e. there is no material loss or surplus in total. In case of interface reprocessing, $q^{\Delta}_i < 0$ for all batches $i \in \mathcal{I}$ such that there is a total loss of material.

Note that batches often have a minimum size in order to receive a sufficient quantity of the intended material at the consumer location. Obviously, $q_i \geq q^{\Delta}_i$ must hold at minimum, but in practice often much larger minimal batch sizes are set (see e.g. Inkpen and Moffett (2011) or

Company (2015)). Therefore, $\underline{q}_s$ indicates the minimum batch size of product s and it holds $\underline{q}_s \geq \max_{t \in \mathcal{S} | t \neq s} (|d_{s,t}|)$. Note that also other technical restrictions imply minimum batch sizes. For instance, let p_s denote the settling period of product s in the consumption tank (i.e. the minimum time span before a new batch can be inserted in the tank). Then, it follows that $\underline{q}_s \geq p_s \cdot \omega_s$. Likewise, batches are often also restricted to an upper limit, i.e. let $\bar{q}_s$ denote the upper limit of batches of product $s \in \mathcal{S}$. Upper batch size limits can be used to model storage capacities. E.g. let l_s^{cap} denote the storage capacity of product s. Then, it has to hold that $L_s \leq l_s^{cap}$ where L_s is the maximum stock level of product s as defined above. Hence, for $\bar{q}_s$ follows $\bar{q}_s \leq \frac{\rho \cdot l_s^{cap}}{\rho - \omega_s}$.

4. Sequencing and scheduling of batch flow pipelines

Based on the explanations in the previous section, in this section a mixed integer non-linear program is formulated for determining an optimal recurring pumping schedule. The necessary notation is summarized in Table 2.

Sets	
$\mathcal{S} = 1, ..., S$	set of products
$\mathcal{I} = 1, ..., I$	set of batches
Parameters	
τ	pumping time
ρ	pump rate
ω_s	demand rate of product $s \in \mathcal{S}$
$c_{s,t}^{setup}$	transition cost for a transition from product $s \in \mathcal{S}$ to $t \in \mathcal{S}$
c_s^{stock}	holding cost for product $s \in \mathcal{S}$
$d_{s,t}$	interface quantity of product s in an interface of products s and t ($d_{s,t} < 0$ indicates a loss, $d_{s,t} > 0$ a suplus)
$\bar{q}_s$	maximum batch size of product $s \in \mathcal{S}$
$\underline{q}_s$	minimum batch size of product $s \in \mathcal{S}$
Variables	
$y_{s,i}$	binary, 1 if product $s \in \mathcal{S}$ is assigned to batch $i \in \mathcal{I}$
q_s^0	starting inventory of product $s \in \mathcal{S}$
u_i	idle time for batch $i \in \mathcal{I}$
q_i	production quantity of batch $i \in \mathcal{I}$
T	total cycle time
T_i	cycle time of batch $i \in \mathcal{I}$
q_i^{Δ}	batch correction quantity for $i \in \mathcal{I}$

Table 2: Set of parameters and decision variables for the sELSP

It is assumed that a set of products $\mathcal{S}$ is assigned to a set of batches $\mathcal{I}$. Each batch $i \in \mathcal{I}$ is characterized by batch time T_i consisting of two time periods: the pumping time (i.e. the batch is inserted) and the idle time u_i (i.e. the pipeline is idle and awaits the next batch injection). The pumping time is the ratio of the corresponding batch quantity q_i and the (constant) pump rate ρ. The total pumping sequence has total cycle time $T = \sum_{i \in \mathcal{I}} T_i$. The pumping sequence is repeated infinitely. I.e. a cyclic schedule approach is assumed which has shown superiority over

the fundamental cycle approach (Wagner and Davis, 2002). As a consequence, at the beginning of each cycle product stocks are at hand for all products except the product assigned to the first batch. These implicit initial stocks are denoted by q_s^0. The formulations presented in the following are similar to the ones given by e.g. Wagner and Davis (2002) and Dobson (1992). However, here an explicit representation of the product allocation decisions is modeled by introducing binary variables $y_{s,i}$.

4.1. Objective functions

The objective function calculates total cost per time unit consisting of stock holding and setup costs. To calculate the average stock holding cost per time unit over the complete pumping cycle, (1) must be weighted with the relative duration of each batch. Hence, $\frac{T_i}{T} \cdot \bar{L}(T_i)$ gives the contribution of the average stock level of batch i to the overall average stock level. Hence, it holds

$$\frac{T_i}{T} \cdot \bar{L}(T_i) = \frac{T_i \cdot \omega \cdot \tau}{T} + \frac{\omega \cdot T_i^2 \cdot \left(1 - \frac{\omega}{\rho}\right)}{T}. \tag{3}$$

For each product $s \in \mathcal{S}$, the sum of cycle times of its associated batches must equal the total cycle time for all products. Otherwise, a permanent deficit or surplus of material would result. Hence, it follows $\sum_{i \in \mathcal{I}} \frac{T_i \cdot \omega_s \cdot \tau}{T} = \omega_s \cdot \tau$ for each product $s \in \mathcal{S}$ such that the first term in (3) is independent of the batch size.

In order to ease the batch quantity correction in the interface case, the second term in (3) is reformulated be replacing T_i with $\frac{q_i}{\omega}$ (i.e. the batch quantity is cycle time times the demand rate of the product). I.e. (3) can be reformulated as

$$\frac{\omega \cdot T_i^2 \cdot \left(1 - \frac{\omega}{\rho}\right)}{T} = \frac{q_i^2 \cdot \left(1 - \frac{\omega}{\rho}\right)}{\omega \cdot T}. \tag{4}$$

Equipped with these components the objective function for the pipeline sequencing problem in the PIG case can be formulated as follows

$$TC^{PIG} = \sum_{s \in \mathcal{S}} c_s^{stock} \cdot \omega_s \cdot \tau + \frac{1}{T} \cdot \sum_{s \in \mathcal{S}} \sum_{i \in \mathcal{I}} y_{s,i} \cdot \frac{c_s^{stock} \cdot q_i^2}{\omega_s} \cdot \left(1 - \frac{\omega_s}{\rho}\right) + \frac{c^{PIG} \cdot \sum_{i \in \mathcal{I}, s \in \mathcal{S}} y_{s,i}}{T}. \tag{5}$$

Note that the first term in (5) is a constant and, hence, can be omitted. The second term calculates average storage cost per time period depending on the batch quantities q_i and the assigned products. The last term in (5) calculates average repositioning cost per time unit by dividing the total repositioning cost $c^{PIG} \cdot \sum_{i \in \mathcal{I}, s \in \mathcal{S}} y_{si}$ by the total cycle time T.

In the interface case, the objective function has to be adjusted for the batch size corrections q_i^{Δ} and sequence-dependent setup costs. Hence, with $y_{s,0} = y_{s,I}$ it follows

$$\begin{aligned} TC^{IF} = & \sum_{s \in \mathcal{S}} c_s^{stock} \cdot \omega_s \cdot \tau + \frac{1}{T} \cdot \sum_{s \in \mathcal{S}} \sum_{i \in \mathcal{I}} c_s^{stock} \cdot y_{s,i} \cdot \frac{(q_i + q_i^{\Delta})^2}{\omega_s} \cdot \left(1 - \frac{\omega_s}{\rho}\right) + \\ & \frac{1}{T} \cdot \sum_{s \in \mathcal{S}} \sum_{\substack{t \in \mathcal{S} \\ t \neq s}} c_{s,t}^{setup} \cdot \left(\sum_{i=1}^{I} y_{s,(i-1)} \cdot y_{t,i}\right). \end{aligned} \tag{6}$$

Note that in (6) the expression of setup cost per period (third term) is only correct if at all lot

positions batches are scheduled.

4.2. Constraints

The following constraints assure sequence feasibility, i.e. for all batches the batch sizes must suffice to cover the time span until the next batch of the same product is scheduled without any scheduling interferences.

$$T = \sum_{i \in \mathcal{I}} u_i + q_i/\rho \tag{7}$$

$$\sum_{i \in \mathcal{I}} y_{si} \cdot \left(q_i + q_i^{\Delta}\right) = \omega_s \cdot T \qquad \forall s \in \mathcal{S} \tag{8}$$

$$y_{s,i} \cdot \left(q_s^0 + \sum_{j \in \mathcal{I}: j<i} y_{sj} \cdot \left(q_j + q_j^{\Delta}\right)\right) = y_{s,i} \cdot \omega_s \cdot \left(\sum_{j \in \mathcal{I} | j<i} u_j + q_j/\rho\right) \qquad \forall s \in \mathcal{S}, i \in \mathcal{I} \tag{9}$$

$$\sum_{s \in \mathcal{S}} y_{si} = 1 \qquad \forall s \in \mathcal{S} \tag{10}$$

$$q_i \geq \sum_{s \in \mathcal{S}} y_{si} \cdot \bar{q}_s \qquad \forall i \in \mathcal{I} \tag{11}$$

$$q_i \leq \sum_{s \in \mathcal{S}} y_{si} \cdot \bar{q}_s \qquad \forall i \in \mathcal{I} \tag{12}$$

$$T, q_s^0, q_i^{\Delta}, q_i, u_i \geq 0 \qquad \forall i \in \mathcal{I} \tag{13}$$

$$y_{s,i} \in \{0, 1\} \qquad \forall i \in \mathcal{I}, s \in \mathcal{S} \tag{14}$$

Constraint (7) assigns the total cycle time as the sum of all pumping and idle times. Constraint set (8) implies that the sum of batch quantities for each product s equal the total demand during the whole cycle. In order to assure sequence feasibility at all times, (9) force an empty stock of product s before a new batch of product s is scheduled. Note that equality in (9) is too restrictive. In fact, instances can be constructed where relaxed constraints ($\geq$ instead of $=$) yield better solutions. However, such situations are rather academic (Wagner and Davis, 2002). Therefore, we stay with the zero-inventory rule here, as otherwise the storage cost calculation would be much more complicated. In the PIG case, q_i^{Δ} can be dropped from (8) and (9). In the interface case, (2) must be considered in order to define q_i^{Δ}.

Constraints (10) enforces that exactly one product is assigned to each batch. Constraints (11) and (12) assure that upper and lower batch size limits are met. Note that batch size limits may lead to infeasible problem instances in combination with the 0-switch-rule. In such a case, either a larger number of batch positions or a relaxation of batch size limits is required in order to achieve feasibility.

Note that in general, it might be possible that not all batches are required to obtain the optimal solution. In these cases, in (10) equality should be replaced by $\leq$. In the PIG case, the outlined MINLP still remains valid, but in the interface cases, the setup cost assessment is no longer correct. Therefore, additional variables and an adjustment of (6) would be needed. In Kirschstein (2015) a similar MINLP is proposed where empty batches are possible but constant batch quantities per product are assumed.

5. Heuristical solution approaches

5.1. Brief literature review on sELSP heuristics

The difficulty of solving the sELSP to optimality is basically caused by the combinatorial complexity of assigning S products to I batches. Once an assignment of products to batches is made, the remaining problem is a simple NLP which can be solved quickly with standard solvers. Hence, the critical part is to solve the assignment problem. Basically, heuristic solution approaches decompose the problem. First for each product the number of batches is determined (so-called *product frequencies*). This defines the total number of batches $|\mathcal{I}|$. Afterwards, the product-batch assignment is performed (so-called *batch sequencing*).

In literature, different approaches for determining product frequencies are proposed such as lower-bound based (LB) approximations (Moon et al., 2002), LB approximations with power-of-2 rounding (Dobson, 1987, 1992), structured enumeration (Wagner and Davis, 2002) or genetic algorithms (Chatfield, 2007).

For batch sequencing, the two cases, sequence-*independent* setup costs (i.e. the PIG case) and sequence-*dependent* setup costs (i.e. the interface case) must be distinguished. In the PIG case, setup cost depend only on the total number of batches scheduled in the cycle. Hence, with given product frequencies, total setup cost is fixed and does not depend on the batch sequence. The batch sequence only influences stock holding cost which eases the evaluation of possible batch sequences. Dobson (1987) applies a bin packing heuristic seeking for a batch sequence spreading batches for each product as evenly as possible over the cycle such that average stocks for each product are small. Similar heuristics are previously introduced by e.g. Doll and Whybark (1973) and Maxwell and Singh (1983).

For cases with sequence-*dependent* setup costs, the batch sequence affects both, setup and stock holding costs. Dobson (1992) subdivides the cycle into sub-sequences which are sequenced by solving an relaxed 1-tree problem. Afterwards, the optimized sub-sequences are iteratively concatenated and remaining product batches are added at their best positions. Wagner and Davis (2002) tackles the sequencing problem by repeating 2-opt and 3-opt swaps based on an initial random batch sequence until convergence. Moon et al. (2002) uses a genetic algorithm for determining the best batch sequence. Similarly, Chatfield (2007) applies a genetic algorithm for determining both, product frequencies and batch sequence in one step utilizing an extended basic period approach (see Wagner and Davis (2002) for a detailed description of basic period approaches). Obviously, heuristics for problems with sequence-dependent setup costs can also be applied for the sequence-independent case.

When comparing the above mentioned heuristics, the focus in literature is typically on the solution quality which is evaluated by comparing the solutions of benchmark data sets such as Bomberger's stamp problem (Bomberger, 1966) or by conducting comparative computational studies with generated data sets (Wagner and Davis, 2002). The running times of the heuristics are not considered by most authors. As a noteworthy exception, Wagner and Davis (2002) showed that the heuristic of Dobson (1992) performs quickly and solved all problem instances within 3 seconds. The solution quality was only slightly inferior compared to the heuristic of Wagner and Davis (2002) with an average gap of less than 1 % and a maximum gap of 4 %. The heuristic

of Wagner and Davis (2002), however, showed an average solution time of 6.3 hours. Likewise, the genetic approaches of Moon et al. (2002) and Chatfield (2007) show slightly superior results for some complex problem instances, but are likely to be less efficient w.r.t. running times than the heuristic of Dobson (1992). Hence, the approach of Dobson (1992) appears to be the best available heuristic for the problems discussed in this paper.

5.2. Adapting Dobson's heuristic

Basically, Dobson's heuristic consists of three steps: 1) Determine the product frequencies $f_s = \sum_{i\in\mathcal{I}} y_{i,s}$ for each product, 2) Build sub-sequence of most-frequent products, 3) iteratively add remaining products. For the sake of brevity in the following bold symbols refer to sets of variables. E.g. $\mathbf{f}$ refers to the vector of product frequencies, i.e. $\mathbf{f}_s = (f_1, ..., f_{|\mathcal{S}|})$. In the following the two basic steps, product frequency and sequencing, are subsequently described.

5.2.1. Determination of product frequencies

The most important step of Dobson's heuristic is to determine f_s in a clever way. Therefore, in Dobson (1992) a series of relaxation and dualization steps is applied. In a first step, dropping (9) to (12) for the moment yields (8) as the only relevant restrictions. It is easy to see that in this case holding cost are minimized if

$$q_s^* = \frac{\omega_s \cdot T}{\sum_{j\in\mathcal{I}} y_{s,j}} = \frac{\omega_s \cdot T}{f_s} \quad \forall s \in \mathcal{S}. \tag{15}$$

I.e. for a given product-batch assignment, holding cost are minimized if all batches of a certain product have equal size. Additionally, it follows that batches are independent of one another. Substituting (15) in (5) then yields

$$\begin{aligned} TC =& C^{pipe} + \frac{C^{setup}(\mathbf{y})}{T} + \frac{1}{T} \cdot \sum_{s\in\mathcal{S}} \sum_{i\in\mathcal{I}} y_{s,i} \cdot \frac{c_s^{stock} \cdot \omega_s^2 \cdot (T)^2}{f_s^2 \cdot \omega_s} \cdot \left(1 - \frac{\omega_s}{\rho}\right) \\ =& C^{pipe} + \frac{C^{setup}(\mathbf{y})}{T} + T \cdot \sum_{s\in\mathcal{S}} \frac{\tilde{c}_s^{stock}}{f_s}. \end{aligned} \tag{16}$$

where $C^{pipe} = \sum_{s\in\mathcal{S}} c_s^{stock} \cdot \omega_s \cdot \tau$ and $\tilde{c}_s^{stock} = c_s^{stock} \cdot \left(1 - \frac{\omega_s}{\rho}\right) \cdot \omega_s$. $C^{setup}(\mathbf{y})$ represent the sequence-dependent setup cost as defined in (5).

The same result can be obtained for the interface case when q_i^{Δ} is neglected and with $C^{setup}(\mathbf{y}) = \sum_{s\in\mathcal{S}} \sum_{\substack{t\in\mathcal{S} \\ t\neq s}} c_{s,t}^{setup} \cdot \left(y_{s,I} \cdot y_{t,1} + \sum_{i=1}^{I-1} y_{s,i} \cdot y_{t,(i+1)}\right)$ which is assumed in the following.

In order to approximate $C^{setup}(\mathbf{y})$ depending on $\mathbf{f}$, the results of Dobson (1992) are adapted with only minor changes. The basic idea is outlined briefly for the sake of completeness. Given the frequencies for each product f_s, minimizing total setup cost equals solving a traveling salesman problem (TSP) with $\sum_{s\in\mathcal{S}} f_s$ nodes to optimality. Each node represents a product and for each product there are f_s corresponding nodes. The graph $G_{\mathbf{f}}$ is complete and the edge weights are denoted by $w_{i,j}$. To approximate the TSP solution, the 1-tree of $G_{\mathbf{f}}$ can be calculated. The special case of $f_s = 1$ for all $s \in \mathcal{S}$ is denoted by $G_{\mathbf{1}}$. The weights connecting the nodes in $G_{\mathbf{1}}$ can be interpreted as the changeover cost $c_{s,t}^{setup}$. Dobson (1992) showed that the value of the 1-tree

of $G_{\mathbf{f}}$ can be calculated using the 1-tree of $G_{\mathbf{1}}$ by

$$v(G_{\mathbf{f}}) = \sum_{s\in\mathcal{S}} (f_s - 1)\cdot w_{e_s} + v(G_{\mathbf{1}}) \tag{17}$$

where w_{e_s} is the least weighted edge incident to node s and $v(G_{\mathbf{1}})$ is the value of the 1-tree of $G_{\mathbf{1}}$. When dualizing the 1-tree constraints it follows that $w_{i,j} = \min\left(c_{i,j}^{setup} + \lambda_i + \mu_j, c_{j,i}^{setup} + \lambda_j + \mu_i\right)$ where λ_i and μ_i are Lagrange multipliers of the dualized 1-tree constraints. More details on the TSP approximation procedure can be found in Dobson (1992).

With $d = v(G_{\mathbf{1}}) - \sum_{s\in\mathcal{S}} w_{e_s}$ and $b_s = w_{e_s} - \lambda_s - \mu_s$, omitting C^{pipe}, and relaxing the integrality condition on f_s, the optimization problem of (16) turns out to be

$$\max_{\boldsymbol{\lambda},\boldsymbol{\mu}\in\mathbb{R}} \left\{ \min_{\mathbf{f}>0,T>0} \left\{ \frac{1}{T}\cdot\left(\sum_{s\in\mathcal{S}} b_s\cdot f_s + d\right) + T\cdot\sum_{s\in\mathcal{S}} \frac{\tilde{c}_s^{stock}}{f_s} \right\} \right\}. \tag{18}$$

As the last step, the batch limit constraints $q_s^* \geq \underline{q}_s$ and $q_s^* \leq \bar{q}_s$ (refer to (11) and (12)) must be taken into account. With $q_s = \frac{\omega_s\cdot T}{f_s}$, these constraints can be reformulated as

$$\frac{\omega_s}{\bar{q}_s} \leq \frac{f_s}{T} \leq \frac{\omega_s}{\underline{q}_s} \tag{19}$$

Dualizing (19), for the inner optimization problem of (18) yields

$$\min_{\mathbf{f}>0,T>0} \left\{ \frac{1}{T}\cdot\left(\sum_{s\in\mathcal{S}} b_s\cdot f_s + d\right) + T\cdot\sum_{s\in\mathcal{S}} \frac{\tilde{c}_s^{stock}}{f_s} + \sum_{s\in\mathcal{S}} \theta_s\cdot\left(\frac{\omega_s}{\bar{q}_s} - \frac{f}{T}\right) + \sum_{s\in\mathcal{S}} \kappa_s\cdot\left(\frac{f}{T} - \frac{\omega_s}{\underline{q}_s}\right) \right\}. \tag{20}$$

Taking the derivative of (20) for f_s and setting to zero then yields

$$f_s^* = T\cdot\sqrt{\frac{\tilde{c}_s^{stock}}{b_s - \theta_s + \kappa_s}} = T\cdot\tilde{f}_s^*. \tag{21}$$

It is easy to see that plugging in f_s^* into (20) let T diminish from all terms except $\frac{d}{T}$. Hence, when dropping $\frac{d}{T}$ the infimum of the objective is obtained and T can be eliminated as a decision variable. Finally, the outer maximization problem of the infimum reduces to

$$\max_{\boldsymbol{\lambda},\boldsymbol{\mu}\in\mathbb{R};\boldsymbol{\theta},\boldsymbol{\kappa}\in\mathbb{R}^+} \left\{ \sum_{s\in\mathcal{S}} \left(\tilde{f}_s^*\cdot(b_s - \theta_s + \kappa_s) + \frac{\tilde{c}_s^{stock}}{\tilde{f}_s^*} + \theta_s\cdot\frac{\omega_s}{\bar{q}_s} - \kappa_s\cdot\frac{\omega_s}{\underline{q}_s} \right) \right\}. \tag{22}$$

(22) can be solved quickly by standard constrained non-linear optimization routines such as sub-gradient optimization (Dobson, 1992) or Newton-type algorithms with support of box constraints (like Byrd et al. 1995) and a feasible initial solution. Hence, solving (22) yields frequency vector $\mathbf{f}^*$ which is in turn used to compose a scheduling cycle.

5.2.2. Batch sequencing

The batch sequencing follows the description of Dobson (1992) almost entirely. Therefore, it is only repeated briefly. For details the reader is referred to Dobson (1992). The procedure starts with ordering the products for decreasing frequencies. Starting with a subset consisting of the products with maximum frequency f^{max}. This subset is sequenced by solving an TSP based on the setup costs. Products with $f^{max}/2$ are added iteratively to the subsequence at the best

possible position. To determine the best position, the change in total cost is calculated consisting of the change in setup and holding cost. The change in setup cost can be calculated easily for any insertion position. The change in holding is approximated by dropping batch size constraints (11) and (12). Hence, the holding cost are determined by (9) constituting a simple system of linear equations.

Therefore, let $\mathbf{L}$ denote the indicator matrix of size $|\mathcal{I}| \times |\mathcal{I}|$ with $l_{i,j} = 1$ if batch i covers the demand of batch j. Note that $\mathbf{L}$ can be easily deduced from a given batch sequence. Furthermore, $\mathbf{q} = \left(q_1, ..., q_{|\mathcal{I}|}\right)$ denotes the vector of batch sizes, $\mathbf{q}^{\Delta} = \left(q_1^{\Delta}, ..., q_{|\mathcal{I}|}^{\Delta}\right)$ the batch size corrections, $\mathbf{u} = \left(u_1, ..., u_{|\mathcal{I}|}\right)$ the vector of idle times, $\mathbf{t}$ the vector of pumping time (whereby $t_i = \frac{q_i}{\rho}$), and $\boldsymbol{\omega} = \left(\sum_{s\in\mathcal{S}} y_{s,1} \cdot \omega_s, ..., \sum_{s\in\mathcal{S}} y_{s,|\mathcal{I}|} \cdot \omega_s\right)$ the vector of demand rates. Then, constraint set (9) can be reformulated in matrix notation as

$$\begin{aligned} & \mathbf{q} + \mathbf{q}^{\Delta} = \operatorname{diag}(\boldsymbol{\omega}) \cdot \mathbf{L} \cdot (\mathbf{u} + \mathbf{t}) \\ \iff \quad & \mathbf{q} = \left(\mathbf{I} - \operatorname{diag}\left(\frac{\boldsymbol{\omega}}{\rho}\right) \cdot \mathbf{L}\right)^{-1} \cdot \left(\operatorname{diag}(\boldsymbol{\omega}) \cdot \mathbf{L} \cdot \mathbf{u} - \mathbf{q}^{\Delta}\right). \end{aligned} \tag{23}$$

From (23), q_i can be substituted in the holding cost part of (5) or (6). Furthermore, (7) can be substituted in the objective, too. Hence, the resulting (unconstrained) holding cost function depends on $\mathbf{u}$ only and, thus, can be optimized easily e.g. by standard Newton-type optimization algorithms.

Afterwards, the previous steps are repeated until all products are scheduled. This way a batch sequence $\mathbf{p}$ is constructed whereby $p_i \in \mathcal{S}$ and $|\mathbf{p}| = \sum_{s\in\mathcal{S}} f_s^*$. Note that rounding to powers of two enables concatenation of subsequences and, hence, simplifies the sequencing procedure. Algorithm 1 describes the sequencing procedure briefly in pseudocode and can be found in the appendix.

5.3. Repair procedure

After batch sequencing, the resulting sequence $\mathbf{p}$ must be checked for batch size feasibility w.r.t. (11) and (12). Therefore, standard solvers for constraint NLPs can be used, e.g. IPOPT (Wächter and Biegler, 2006). Unfortunately, the constructed sequence may turn out to be infeasible. In order to find a feasible sequence, first pertubations of $\mathbf{p}$ are constructed and checked for feasibility. Therefore, reasonable swaps of pairs of batches are conducted. A swap of sequence $\mathbf{p}$ exchanges the batches i and j. A swap is reasonable if a) the assigned products differ (i.e. $p_i \neq p_j$), and b) the exchange does not lead to a sequence where the same product is assigned to two successive batches. If a reasonable swap leads to a feasible solution, the corresponding sequence is returned and the procedure stops. Otherwise, the product frequencies have to be changed in a second step.

Therefore, let $\bar{\delta}_i(\mathbf{p})$ and $\underline{\delta}_i(\mathbf{p})$ denote the violation of the lower and upper batch size limits of batch i, i.e. $\bar{\delta}_i(\mathbf{p}) = \max\{0, q_i - \bar{q}_{p_i}\}$ and $\underline{\delta}_i(\mathbf{p}) = \max\left\{0, \underline{q}_{p_i} - q_i\right\}$. All products assigned to batches exceeding the maximum batch size (i.e. $\bar{\delta}_i(\mathbf{p}) > 0$) are candidates for an increase of their corresponding product frequencies and are summarized in set $\mathcal{S}^+$.

Likewise, products associated with batches showing a shortfall below minimum batch sizes (i.e. $\underline{\delta}_i(\mathbf{p}) > 0$) are candidates for a product frequency reduction summarized in set $\mathcal{S}^-$. Moreover,

suppose a product s with $\underline{\delta}_i(\mathbf{p}) > 0$ for some batch and $f_s = 1$. In this case, a frequency reduction is clearly impossible. But a similar effect can be realized by increasing the frequencies of the complementary products $\mathcal{S}\backslash\{s\}$. Hence, in such a case subset $\mathcal{S}\backslash\{s\}$ is added to $\mathcal{S}^+$.

In order to reduce the number of frequency adaptations to be tested, increases and reductions leading to clearly infeasible frequency combinations can be discarded. Upper and lower bounds for f_s based on lower and upper batch size limits are given by $f_s \leq T \cdot \frac{\omega_s}{\bar{q}_s}$ and $f_s \geq T \cdot \frac{\omega_s}{\underline{q}_s}$ where T can be substituted by $T = \frac{\sum_{i\in\mathcal{I}} q_i}{\rho} \cdot \frac{\rho}{\sum_{s\in\mathcal{S}} \omega_s} = \frac{\sum_{i\in\mathcal{I}} q_i}{\Omega}$ with $\Omega = \sum_{s\in\mathcal{S}} \omega_s$. Furthermore, for the total batch size holds $\sum_{t\in\mathcal{S}} f_t \cdot \underline{q}_t \leq \sum_{i\in\mathcal{I}} q_i \leq \sum_{t\in\mathcal{S}} f_t \cdot \bar{q}_t$. In combination, it follows

$$f_s \leq \frac{\sum_{i\in\mathcal{I}} q_i}{\Omega} \cdot \frac{\omega_s}{\bar{q}_s} \leq \frac{(\sum_{t\in\mathcal{S}} f_t \cdot \bar{q}_t) \cdot \omega_s}{\Omega \cdot \bar{q}_s} \quad \Longleftrightarrow \quad f_s \cdot \left(\frac{\bar{q}_s \cdot \Omega}{\omega_s} - \underline{q}_s\right) \leq \sum_{t\in\mathcal{S}|t\neq s} f_t \cdot \bar{q}_t \tag{24}$$

$$f_s \geq \frac{\sum_{i\in\mathcal{I}} q_i}{\Omega} \cdot \frac{\omega_s}{\underline{q}_s} \geq \frac{\left(\sum_{t\in\mathcal{S}} f_t \cdot \underline{q}_t\right) \cdot \omega_s}{\Omega \cdot \bar{q}_s} \quad \Longleftrightarrow \quad f_s \cdot \left(\frac{\underline{q}_s \cdot \Omega}{\omega_s} - \bar{q}_s\right) \geq \sum_{t\in\mathcal{S}|t\neq s} f_t \cdot \underline{q}_t. \tag{25}$$

For any $s \in \mathcal{S}^+ \cup \mathcal{S}^-$, it is checked whether (24) or (25) is violated with f_t, $t \neq s$ kept fixed and $f_s + 1$ or $f_s - 1$ if $s \in \mathcal{S}^+$ or $s \in \mathcal{S}^-$, respectively. If (24) or (25) is violated, s can be discarded from set $\mathcal{S}^+$ or $\mathcal{S}^-$.

Once the sets $\mathcal{S}^+$ and $\mathcal{S}^-$ are determined, successively alterations of the sequence $\mathbf{p}$ are tested. For all products $s \in \mathcal{S}^+$, an additional batch of product s is added to sequence $\mathbf{p}$. Again, all reasonable insertion options are checked for feasibility. I.e. a batch of product s is added at position $i \in \{1, ..., I\}$ in sequence $\mathbf{p}$ if $p_{i-1} \neq s$ and $p_{i+1} \neq s$. The resulting sequence $\mathbf{p}'$ has $I + 1$ batches and is checked for feasibility. The described procedure is subsumed in pseudocode as $\textsc{Add}(\mathbf{p}, s)$ and returns a set of sequences with $I + 1$ batches. If $\textsc{Add}(\mathbf{p}, s)$ produces feasible sequences, the sequence with minimum total cost is returned and the procedure stops. Otherwise, the sequences are kept in set $\mathcal{P}$. Similarly, the procedure $\textsc{Del}(\mathbf{p}, s)$ evaluates all reasonable deletions of batches of product s from sequence $\mathbf{p}$. Again, $\textsc{Del}(\mathbf{p}, s)$ returns a set of sequences (now with $I - 1$ batches) which are checked for feasibility. In case, no feasible sequence is found, all evaluated sequences are stored in set $\mathcal{P}$.

If no feasible solution was found by adding ad deleting batches, the whole repair procedure is repeated with a new starting sequence $\mathbf{p}' \in \mathcal{P}$. Sequence $\mathbf{p}'$ is the sequence which minimizes the total batch size violation, i.e. $\mathbf{p}' = \arg\min_{\mathbf{p}\in\mathcal{P}} \delta(\mathbf{p}) = \sum_{i\in\mathcal{I}_\mathbf{p}} \bar{\delta}_i(\mathbf{p}) + \underline{\delta}_i(\mathbf{p})$. This criterion tries to identify a solution close to feasibility. If the procedure does not stop with a feasible solution, some stopping criterion, such as the number of iterations, is checked. Furthermore, it may happen that the same sequence appears again during the search. This can be interpreted as a sign for an infeasible problem or a tabu list can be maintained in order to avoid cycles. Algorithm 2 in the Appendix describes the outlined procedure in pseudocode.

6. Computational experiments

6.1. Case study from chemical industry

In the following a serial one-to-one pipeline connecting two chemical production sites is assumed. The provider site produces the raw and intermediate materials for the consumer site. Suppose that the consumer site produces Styrene-based chemicals, such as Polystyrene and Styrene-Butadiene rubber. Raw materials for Styrene production are Benzene and Ethylene. Among others, a steam cracker consuming Naphtha provides Ethylene and Pygas which is subsequently refined to Benzene. Pygas and Naphtha are mixtures of hydrocarbons, whereby Pygas can be interpreted as a sub-mixture primarily consisting of aromatic chemicals with main component Benzene. However, the local production of Benzene and Pygas does not suffice to fully supply downstream assets such that deficits of Pygas and Benzene have to be imported via pipeline. Besides, the cracker feed, Naphtha, has to be imported from the provider site, too.
The net demand rates provided in tons per day for Naphtha, Pygas, and Benzene under normal processing conditions can be found in Table 3 accompanied by stock holding cost rates in € per ton and day as well as lower and upper batch size limits.

chemical s	stock cost c_s^{hold} [€/day]	net demand rates ω_s [tons/day]	batch size limits lower	upper
Naphtha	0.10	3,000	3,000	7,000
Pygas	0.15	2,000	2,000	10,000
Benzene	0.20	1,000	1,000	5,000

Table 3: Net demand rate ω_s, holding cost rates c_s^{hold}, and batch limits for the first case study

Typically, it can be assumed that the market value of (basic) chemicals increases with increasing purity. Hence, stock holding cost can be assumed to increase with corresponding market value. Lower batch size limits are set to daily demand values of each chemical in order to avoid too erratic schedules. Upper batch size limits reflect operational stock capacities differing for the various products.
The pipeline's pumping capacity is assumed to be $\rho = 7,000$ tons/day. The pipeline capacity suffices to supply the consumer site with all three raw and intermediate chemicals. The pipeline is assumed to be operated at maximum capacity (as operating cost are optimized for this pump rate) or is completely inactive. The pipeline length is assumed to be 600 km. With an average transport speed of 5 km/h, a transport time of $\tau = 5$ days results.
The three considered chemicals are mixable, chemically related liquids and can be transported via the same pipeline. At the provider site these three chemicals are surplus and transported via a commonly used pipeline. Ordering these chemicals to increasing purity leads to 1. Naphtha, 2. Pygas, and 3. Benzene. E.g. an interface consisting of Pygas and Naphtha is handled as Naphtha and stored in the Naphtha tanks where the concentration of aromatics increases due to the Pygas inflow. The cracking process separates these components again. Consequently, downgraded ingredients are re-processed implying additional processing cost. I.e. an interface of Benzene and Naphtha has to be downgraded to Naphtha and the Benzene part is re-processed twice. Table 4 shows the interface quantities $q_{s,t}^{\Delta}$ and associated transition cost $c_{s,t}^{setup}$. With these data, the first

chemical	interface quantities $q^{\Delta}_{s,t}$ [tons]			transition cost $c^{setup}_{s,t}$ [€]		
s	Naphtha	Pygas	Benzene	Naphtha	Pygas	Benzene
Naphtha	0	20	50	0	250	1,500
Pygas	-20	0	30	250	0	450
Benzene	-50	-30	0	1,500	450	0

Table 4: Interface quantities $q^{\Delta}_{s,t}$ and transition cost $c^{setup}_{s,t}$ for the first case study

part of the proposed heuristic delivers product frequencies of $\mathbf{f}^* = (1.44, 2.10, 1.0)$. For the sequencing step, the frequency vector is rounded to the nearest power-of-two such that $\mathbf{f}^0 = (1, 2, 1)$. The sequencing procedure produces the sequence $\mathbf{p}^0 = (Pygas, Benzene, Pygas, Naphta)$ with total cost of 5,669.13 € per day. Unfortunately, the corresponding batch sizes violate the batch size constraints as Table 5 shows.

batch pos. i	product scheduled	lower batch size $\underline{q}_i$	batch size q_i	upper batch size $\bar{q}_i$	pumping time $\frac{q_i}{\rho}$	idle time u_i
1	Pygas	2,000	**1,900**	10,000	0.27	0.17
2	Benzene	1,000	2,440	5,000	0.35	0.17
3	Pygas	2,000	2,839	10,000	0.41	0.00
4	Naphta	3,000	**7,100**	7,000	1.01	0.00
		Total			2.04	0.34

Table 5: Batch sequence of first case study before repair routine

Applying the repair routine to find feasible sequence, first tries to rearrange the originally found sequence. In this particular case, there are $\binom{4}{2} = 6$ possible swaps of which only the swap of batches 2 and 4 is reasonable. This permutation does not yield a feasible sequence. Hence, changing product frequencies is necessary. From Table 5 it can be observed that at the size of batch 1 falls below the lower limit. Hence, the frequency of Pygas might be reduced and, hence, Pygas is added to $\mathcal{S}^-$. In contrast, the size of batch 4 exceeds the upper limit. The corresponding product Naphtha is added to set $\mathcal{S}^+$.

Next, additional batches of the products in set $\mathcal{S}^+$ are plugged into $\mathbf{p}^0$. Hence, a batch of Naphtha can be added to $\mathbf{p}^0$ between batches 1 and 2 or batches 2 and 3. Both insertions lead to a feasible solution with total cost of 5,983.52 €. Table 6 displays pumping and idle times as well as batch sizes for the feasible sequence $\mathbf{p}^1 = (Pygas, Naphta, Benzene, Pygas, Naphta)$. Figure 4 visualizes the batch size corrections q^{Δ}_i corresponding to $\mathbf{p}^1$.

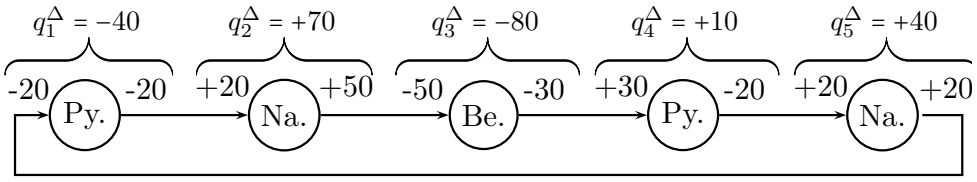

Figure 4: Illustration of interface calculation in batch pipeline systems

batch pos. i	product scheduled	lower batch size $\underline{q}_i$	batch size q_i	upper batch size $\bar{q}_i$	pumping time $\frac{q_i}{\rho}$	idle time u_i
1	Pygas	2,000	2,778	10,000	0.40	0.00
2	Naphta	3,000	4,002	7,000	0.57	0.00
3	Benzene	1,000	2,802	5,000	0.40	0.00
4	Pygas	2,000	2,697	10,000	0.39	0.00
5	Naphta	3,000	4,055	7,000	0.58	0.39
Total					2.34	0.39

Table 6: Batch sequence of first case study after repair routine

6.2. Case study from petroleum industry

In this section the case study presented in Relvas et al. (2013) is reviewed. As the presented scheduling problem is a operational, finite-horizon planning problem, the problem data are slightly altered to fit the requirements of the planning framework proposed here. There are three notable aspects which differ structurally between the both planning models. First, no cost rates for product transition or stock holding are assumed. Second, instead of transition cost, precedence constraints are used in Relvas et al. (2013), i.e. there are forbidden and allowed product transitions. Third, storage capacities and settling periods as well as typical lot sizes are given in Relvas et al. (2013).

Storage capacities and settling periods implicitly limit the batch sizes. Therefore, these can be transformed into batch size limits for each product. Hence, Storage capacities are used to calculate upper batch size limits by $\bar{q}_s = \frac{\rho \cdot l_s^{cap}}{\rho - \omega_s}$ as outlined above. Similarly, lower batch sizes are calculated as $\underline{q}_s = p_s \cdot \omega_s$ (scenario S1). As an alternative scenario (S2), the maximal typical lot sizes given in Table 1 of Relvas et al. (2013) are used.

As no transition cost, but precedence constraints are given, $c_{s,t}^{setup}$ is set to 0 if a transition $s \to t$ is possible. Otherwise, $c_{s,t}^{setup}$ is set to a large number M. Likewise, holding costs c_s^{stock} are assumed to be 0, too. Note that this implies that only a feasible solution is to be found. Hence, total costs have no reasonable interpretation and are, therefore, not reported. However, the proposed heuristic is not designed to cope with hard precedence constraints, i.e. it is not guaranteed that a precedence-feasible solution is obtained. Therefore, the repair routing needs to be adapted to check for precedence violations of a given sequence. The necessary alterations, however, are minor and, therefore, omitted here. For the sake of readability, the case study data are summarized in Table 7. The product's demand rates are calculated as averages from Table 5 of Relvas et al. (2013). All flow-related values represent volume units (VU). Demand rates are given per day. Settling periods are given in days, too. To complete the data, pumping capacity ρ is assumed to be $15,600$ VU per day and τ is calculated as 1.225 days.

Equipped with these data, for scenario 1 a feasible sequence w.r.t. batch size as well as precedence constraints could be found after the repair routine. The sequence has 11 batches with frequencies $\mathbf{f}^{sc1} = (3, 2, 1, 2, 2, 1)$. The total cycle time is about 20 days which is fairly similar to case study CS2 from Relvas et al. (2013) where 16 batches are scheduled for a time horizon of 15 days. The details of the sequence are displayed in Table 8.

The results in Table 8 show that the batches scheduled are in tendency larger than the batches

s	l_s^{cap}	p_s	d_s	$\underline{q}_s$	$\bar{q}_s^{S1}$	$\bar{q}_s^{S2}$	transition allowed P1	P2	P3	P4	P5	P6
P1	81,500	1	5,968	5,968	131,998	21,800	—	✓	✓	✓	—	—
P2	32,000	2	2,429	4,858	37,901	16,000	✓	—	—	—	—	—
P3	24,000	1	598	598	24,957	16,000	✓	—	—	✓	✓	—
P4	27,800	1	2,153	2,153	32,251	11,800	✓	—	✓	—	✓	—
P5	10,320	1	322	322	10,537	3,200	—	—	✓	✓	—	✓
P6	13,120	1	607	607	13,651	6,200	—	—	—	—	✓	—

Table 7: Case study data extracted from Relvas et al. (2013) containing storage capacities (in VU), settling periods (in days), demand (in VU), batch size limits (in VU) and allowed product transitions.

batch pos. i	own solution	lower batch size $\underline{q}_i$	batch size q_i	upper batch size $\bar{q}_i$	pumping time $\frac{q_i}{\rho}$	idle time u_i
1	P1	5,968	36,733	131,998	2.35	1.37
2	P2	4,858	37,901	37,901	2.43	0.00
3	P1	5,968	76,083	131,998	4.88	0.00
4	P3	598	11,970	24,957	0.77	0.00
5	P4	2,153	10,844	32,251	0.70	0.00
6	P5	322	1,292	10,538	0.08	1.45
7	P6	607	12,150	13,651	0.78	1.70
8	P5	322	5,154	10,538	0.33	0.00
9	P4	2,153	32,251	32,251	2.07	0.00
10	P1	5,968	6,641	131,998	0.43	0.00
11	P2	4,858	10,718	37,901	0.69	0.00
		Total			15.50	4.52

Table 8: Final batch sequence of second case study (scenario 1)

calculated by Relvas et al. (2013). As the upper batch size limits are comparatively high and there are no cost information considered, this is not surprising. E.g. for batches 2 and 9 the maximum possible batch size is assigned. For many other batches, however, batch sizes are similar to the typical values reported in Relvas et al. (2013).

Nonetheless, for scenario 2 the upper batch size limits are set to the maximal typical values. Of course, setting tighter batch size limits complicates finding a feasible solution. Therefore, the repair routine requires some more iterations for finding a sequence and precedence feasible solution for scenario 2. Here, a feasible sequence with 12 batches and frequencies $\mathbf{f}^{sc2} = (4, 2, 1, 2, 2, 1)$ is found. The total cycle time is halved to about 10.3 days compared to scenario 1 which is to be expected as batch sizes have to be reduced considerably for some products. Table 9 shows the details of the found solution.

For scenario 2, most batches are typically sized, whereby batches 1, 5, 8, 9, and 11 are smaller than the smallest typical batch size reported in Relvas et al. (2013). However, the deviations are comparatively small. Nonetheless, further narrowing the batch size limits by increasing lower limits to the smallest typical values complicates the problem instance to an extent that prohibits finding a feasible solution (at least in a reasonable amount of time). Nonetheless, quite similar results compared to the solutions reported in Relvas et al. (2013) could be found although a more restricted problem is addressed here as the 0-switch rule is applied. As in this particular

batch pos. i	product scheduled	batch size (0-switch) q_i	batch size (relaxed) q_i	typical lower batch size	typical upper batch size
1	P1	**15,190**	**9,058**	17,300	21,800
2	P2	12,451	11,434	8,000	16,000
3	P1	21,800	17,300	17,300	21,800
4	P4	10,509	11,800	3,800	11,800
5	P5	**322**	2,489	800	3,200
6	P6	6,200	6,200	3,100	6,200
7	P5	2,967	800	800	3,200
8	P3	**6,108**	**6,108**	8,000	16,000
9	P1	**10,528**	17,300	17,300	21,800
10	P2	12,360	13,377	8,000	16,000
11	P1	**13,440**	17,300	17,300	21,800
12	P4	11,482	10,191	3,800	11,800

Table 9: Final batch sequence of second case study (scenario 2), batches with unusual batch sizes are set bold

case study stock holding cost are assumed to be zero, calculating stock levels correctly does not matter. Hence, the 0-switch rule can be relaxed, i.e. equality is no longer required in (9). When relaxing the 0-switch rule and optimizing for a minimal deviation from typical batch sizes, the number of untypically-sized batches can be further minimized as shown in 4th column in Table 9.

7. Conclusion

This paper formulates a scheduling model for serial one-to-one pipelines with multiple products. Given a static parameter setting, determining an optimal sequence minimizing stock holding and setup costs can be modeled as a variant of the economic lot scheduling model. Depending on the batch separation technology applied, the ELSP has to cope with sequence-independent (physical batch separation) or sequence-dependent setup costs (interface separation). Additionally, side constraints on the minimal and maximal batch size are added to the model. As the derived ELSP, particularly the sequence-dependent version, is hard to solve to optimality, the heuristic of Dobson (1992) is adapted and enhanced to meet the specific requirements of proposed scheduling problem. In particular, a repair routine is presented in order to find a feasible solution w.r.t. the batch size constraints. Case studies from chemical and petroleum industry illustrate the applicability of the proposed heuristic.

Further research might be devoted to incorporate more technical details into the planning framework such as settling times, switch-over times or inspection periods. Furthermore, an extension to pipeline networks with multiple injection and retraction points or systems allowing reverse product flows might be of interest.

References

E. E. Bomberger. A dynamic programming approach to a lot size scheduling problem. *Management science*, 12(11):778–784, 1966.

R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

D. Cafaro and J. Cerdá. Optimal scheduling of multiproduct pipeline systems using a non-discrete MILP formulation. *Computers & chemical engineering*, 28(10):2053–2068, 2004. ISSN 0098-1354.

D. Cafaro and J. Cerdá. Dynamic scheduling of multiproduct pipelines with multiple delivery due dates. *Computers & Chemical Engineering*, 32:728–753, 2008.

D. Cafaro and J. Cerdá. Optimal scheduling of refined products pipelines with multiple sources. *Industrial & Engineering Chemistry Research*, 48(14):6675–6689, 2009.

D. Cafaro and J. Cerdá. Operational scheduling of refined products pipeline networks with simultaneous batch injections. *Computers & Chemical Engineering*, 34(10):1687–1704, 2010.

D. Chatfield. The economic lot scheduling problem: A pure genetic search approach. *Computers & Operations Research*, 34(10):2865–2881, 2007.

C. P. Company. Informational topics, 2015. http://www.colpipe.com/docs/default-source/info-topics/informational-topics.pdf?sfvrsn=6.

G. Dobson. The economic lot-scheduling problem: achieving feasibility using time-varying lot sizes. *Operations Research*, 35(5):764–771, 1987.

G. Dobson. The cyclic lot scheduling problem with sequence-dependent setups. *Operations Research*, 40(4):736–749, 1992.

C. L. Doll and D. C. Whybark. An iterative procedure for the single-machine multi-product lot scheduling problem. *Management Science*, 20(1):50–55, 1973.

A. C. Inkpen and M. H. Moffett. *The Global Oil & Gas Industry: Management, Strategy and Finance*. PennWell Books, 2011.

T. Kirschstein. *Integrated Supply Chain Planning in Chemical Industry: Potentials of Simulation in Network Planning*. Springer, 2015.

S. Magataõ, L. Magataõ, F. Neves-Jr, and L. Arruda. Novel milp decomposition approach for scheduling product distribution through a pipeline network. *Industrial & Engineering Chemistry Research*, 54(18):5077–5095, 2015.

L. Magatão, L. Arruda, and F. Neves-Jr. A mixed integer programming approach for scheduling commodities in a pipeline. *Computers & Chemical Engineering*, 28(1-2):171–185, 2004.

L. Magatão, L. Arruda, and F. Neves-Jr. Using clp and milp for scheduling commodities in a pipeline. *Computer Aided Chemical Engineering*, 20:1027–1032, 2005.

L. Magatão, L. Arruda, and F. Neves-Jr. A combined clp-milp approach for scheduling commodities inÂ aÂ pipeline. *Journal of Scheduling*, 14(1):57–87, 2011.

W. L. Maxwell and H. Singh. The effect of restricting cycle times in the economic lot scheduling problem. *AIIE Transactions*, 15(3):235–241, 1983.

S. MirHassani. An operational planning model for petroleum products logistics under uncertainty. *Applied Mathematics and Computation*, 197:744–751, 2008.

S. MirHassani and H. Fani Jahromi. Scheduling multi-product tree-structure pipelines. *Computers & chemical engineering*, 35(1):165–176, 2011.

I. Moon, E. A. Silver, and S. Choi. Hybrid genetic algorithm for the economic lot-scheduling problem. *International Journal of Production Research*, 40(4):809–824, 2002.

S. Moradi and S. MirHassani. Transportation planning for petroleum products and integrated inventory management. *Applied Mathematical Modelling*, 39(23):7630–7642, 2015.

S. Moradi and S. MirHassani. Robust scheduling for multi-product pipelines under demand uncertainty. *The International Journal of Advanced Manufacturing Technology*, pages 1–9, 2016.

H. Mostafaei, P. M. Castro, and A. Ghaffari-Hadigheh. A novel monolithic milp framework for lot-sizing and scheduling of multiproduct treelike pipeline networks. *Industrial & Engineering Chemistry Research*, 54(37):9202–9221, 2015.

H. Mostafaei, P. M. Castro, and A. Ghaffari-Hadigheh. Short-term scheduling of multiple source pipelines with simultaneous injections and deliveries. *Computers & Operations Research*, 73: 27–42, 2016.

A. Moura, C. de Souza, A. Cire, and T. Lopes. Planning and scheduling the operation of a very large oil pipeline network. In *Principles and Practice of Constraint Programming*, pages 36–51. Springer, 2008.

M. Narro Lopez and B. Kingsman. The economic lot scheduling problem: theory and practice. *International Journal of Production Economics*, 23(1-3):147–164, 1991.

R. Rejowski and J. Pinto. Scheduling of a multiproduct pipeline system. *Computers & Chemical Engineering*, 27:1229–1268, 2003.

R. Rejowski and J. Pinto. A novel continuous time representation for the scheduling of pipeline systems with pumping yield rate constraints. *Computers & Chemical Engineering*, 32(4): 1042–1066, 2008.

R. Rejowski et al. Efficient MILP formulations and valid cuts for multiproduct pipeline scheduling. *Computers & chemical engineering*, 28(8):1511–1528, 2004. ISSN 0098-1354.

S. Relvas, H. Matos, A. Barbosa-Póvoa, J. Fialho, and A. Pinheiro. Pipeline scheduling and inventory management of a multiproduct distribution oil system. *Industrial & engineering chemistry research*, 45(23):7841–7855, 2006.

S. Relvas, A. Barbosa-Póvoa, and H. Matos. Heuristic batch sequencing on a multiproduct oil distribution system. *Computers & Chemical Engineering*, 33(3):712–730, 2009.

S. Relvas, S. Boschetto Magatão, A. P. Barbosa-Póvoa, and F. Neves Jr. Integrated scheduling and inventory management of an oil products distribution system. *Omega*, 41(6):955 – 968, 2013.

S. Sethi and G. Sorger. A theory of rolling horizon decision making. *Annals of Operations Research*, 29(1):387–415, 1991.

H. van Essen, H. Croezen, and J. Nielsen. Emissions of pipeline transport compared with those of competing modes, 2003. Delft, Association of Petrochemicals Producers in Europe.

A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

B. J. Wagner and D. J. Davis. A search heuristic for the sequence-dependent economic lot scheduling problem. *European Journal of Operational Research*, 141(1):133–146, 2002.

A. Pseudocodes

Algorithm 1 Procedure for sequencing batches with given frequencies

procedure SEQUENCING($\mathbf{f}$)
  normalize product frequencies $f_s \leftarrow f_s / \min_{s \in \mathcal{S}} f_s$
  round f_s to nearest power-of-two
  $\mathcal{S}^p \leftarrow \arg\max_{s \in \mathcal{S}} f_s$ ▷ determine most frequent products
  build sub-sequence $\mathbf{p}$ by solving TSP with $s \in \mathcal{S}^p$
  $\mathcal{S}^{rest} \leftarrow \mathcal{S} / \mathcal{S}^p$ ▷ determine products to be sequenced
  repeat
    determine $\bar{f}^{rest} \leftarrow \max_{s \in \mathcal{S}^{rest}} f_s$ and $\mathcal{S}^p \leftarrow \arg\max_{s \in \mathcal{S}^{rest}} f_s$
    concatenate $\mathbf{p}$ with itself $\frac{\min_{s \in \mathcal{S}^p} f_s}{\bar{f}^{rest}}$ times ▷ $\frac{\min_{s \in \mathcal{S}^p} f_s}{\bar{f}^{rest}}$ is a multiple of two
    for $s \in \mathcal{S}^p$ **do**
      add s at its best position in $\mathbf{p}$
    end for
    $\mathcal{S}^{rest} \leftarrow \mathcal{S}^{rest} / \mathcal{S}^p$
  until $\mathcal{S}^{rest} \neq \emptyset$
  return p
end procedure

Algorithm 2 Repair procedure for finding a feasible batch sequence

procedure REPAIR($\mathbf{p}$)
  repeat
  //try swaps of current sequence $\mathbf{p}$ to find feasible solution
    set $\mathbf{f}' \leftarrow \mathbf{f}$ ▷ retain product frequencies of $\mathbf{p}$
    for $(i,j) \in \mathbf{p}$ **do**
      if $p_i \neq p_j \wedge p_{i-1} \neq p_j \wedge p_{i+1} \neq p_j \wedge p_{j-1} \neq p_i \wedge p_{j+1} \neq p_i$ **then**
        $\mathbf{p}' \leftarrow \text{SWAP}(\mathbf{p}, i, j)$ ▷ exchange batches i and j
        if $\mathbf{p}'$ is a feasible sequence **then**
          return $\mathbf{p}'$
        end if
      end if
    end for
  //change frequencies
    $\bar{\delta}_i(\mathbf{p}) \leftarrow \max\{0, q_i - \bar{q}_{p_i}\}$ and $\underline{\delta}_i(\mathbf{p}) \leftarrow \max\left\{0, \underline{q}_{p_i} - q_i\right\}$ ▷ calculate batch size violations
    $\delta(\mathbf{p}) \leftarrow \sum_{i \in \mathcal{I}} \bar{\delta}_i(\mathbf{p}) + \underline{\delta}_i(\mathbf{p})$ ▷ assign total slack
    $\mathcal{S}^+ \leftarrow \{s \mid s = p_i \wedge \bar{\delta}_i > 0\}$ ▷ subset for which frequencies are increased
    if $\exists i$ with $\underline{\delta}_i < 0 \wedge f_{p_i} = 1$ **then**
      $\mathcal{S}^+ \leftarrow \mathcal{S}^+ \cup \mathcal{S} \backslash \{s \mid s = p_i \wedge \underline{\delta}_i < 0 \wedge f_{p_i} = 1\}$ ▷ subset with counterincrease
    end if
    $\mathcal{S}^- \leftarrow \{s \mid s = p_i \wedge \underline{\delta}_i > 0 \wedge f_{p_i} > 1\}$ ▷ subset for which frequencies are reduced
    apply (25) or (24) to reduce $\mathcal{S}^-$ and $\mathcal{S}^+$
    $\mathcal{P} \leftarrow \varnothing$ ▷ initialize set of new sequences
  // try frequency increase
    for $s \in \mathcal{S}^+$ **do**
      $\mathcal{P}_s^+ \leftarrow \text{ADD}(\mathbf{p}, s)$ ▷ add batch of product s at all reasonable positions
      if any $\mathbf{p}' \in \mathcal{P}_s^+$ is feasible **then**
        return $\mathbf{p}'$ ▷ return feasible sequence with minimal cost
      else
        $\mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{P}_s^+$ ▷ retain $\mathcal{P}_s^+$
      end if
    end for
  // try frequency reductions
    for $s \in \mathcal{S}^-$ **do**
      $\mathcal{P}_s^- \leftarrow \text{DEL}(\mathbf{p}, s)$ ▷ delete batch of product s from all reasonable positions
      if any $\mathbf{p}' \in \mathcal{P}_s^-$ is feasible **then**
        return $\mathbf{p}'$ ▷ return feasible sequence with minimal cost
      else
        $\mathcal{P} \leftarrow \mathcal{P} \cup \mathcal{P}_s^-$ ▷ retain $\mathcal{P}_s^-$
      end if
    end for
    $\mathbf{p} \leftarrow \arg\min_{\mathbf{p}' \in \mathcal{P}} \delta(\mathbf{p}')$ ▷ select sequence with minimal total violation
  until stopping criterion met or cycle appears
end procedure