

OIL-DERIVATIVES PIPELINE LOGISTICS USING DISCRETE-EVENT SIMULATION

Vanina G. Cafaro
Diego C. Cafaro
Carlos A. Méndez
Jaime Cerdá

INTEC (UNL-CONICET)
Güemes 3450
Santa Fe, 3000, ARGENTINA

ABSTRACT

The management of oil-product pipelines represents a critical task in the daily operation of petroleum supply chains. Efficient computational tools are needed to perform this activity in a reliable and cost-effective manner. This work presents a novel discrete event simulation system developed on Arena® for the detailed scheduling of a multiproduct pipeline consisting of a sequence of pipes that connect a single input station to several receiving terminals. The pipeline is modeled as a non-traditional multi-server queuing system involving a number of servers at every pipe-end that perform their tasks in a synchronized manner. Based on priority rules, the model decides which server should dispatch the entity waiting for service to the associated depot. Each priority rule can lead to a different delivery schedule, which is evaluated by using several criteria. Combined with optimization tools, the proposed simulation technique permits to easily manage real-world pipelines operations with low computational effort.

1 INTRODUCTION

Pipeline management plays a key role in the petroleum business. Oil pipelines represent the most reliable and cost-efficient way to transport large amounts of liquid fuels over long distances. In contrast to other transportation modes, pipelines can operate continuously with almost no interruptions despite bad weather conditions. Furthermore, pipelines have an important edge on environmental and safety issues. They offer great environmental and safety benefits since they do not crowd our highways and rivers and produce negligible air pollution. They also have lower spill rates than competing modes of transportation, namely trucks and barges.

Refined products pipelines are able to transport a variety of oil derivatives that move end to end in successive batches. Moreover, multiproduct pipelines can operate in two different ways: segregated or fungible. Segregated products are branded or blend stock materials whose identity is maintained throughout the transportation route, and the same batch that is received for shipment in the origin is delivered at the destination. On the other hand, fungible batches consist of generic products that meet published specifications. Shippers will receive an equivalent product that match the same product specifications, but may not be the original lot that has been shipped at the specified input terminal.

1.1 Pipeline scheduling

Scheduling of multiproduct pipelines is a very difficult task with many constraints to be considered. Process operation involves two major activities: generating the input and the delivery schedules. In a first

step, the sequence of batch injections, the entering products, the batch sizes, the pump rates, and the input terminal where products will be injected are initially defined. Finding the optimal product sequence and lot sizes is a combinatorial problem aimed at minimizing the interface costs due to product mixing and/or pipeline cleaning. Since separation devices are rarely used, some sequences are definitely forbidden because of product contamination. In a second stage, the delivery schedule specifies the product batches leaving the pipeline and the amounts diverted to the assigned destinations during every pumping run. This stage provides the times at which pumps should be turned on/off to perform the delivery plan. Its main goal is to reduce the number of pipeline stoppages and pump switchings so as to get savings on the energy cost consumed for restarting flow in idle segments, and on pump maintenance costs. A Discrete-Event Simulator of multiproduct pipeline operations is a particularly useful computer-aided tool for quickly generating more efficient, realistic and robust schedules. It permits to easily find alternative detailed schedules by using different operational criterion.

Several approaches were proposed to study pipeline scheduling problems, including rigorous optimization models, knowledge-based techniques (Sasikumar et al. 1997), discrete-event simulation (Mori et al. 2007, García-Sánchez, Arreche, and Ortega-Mier 2008), and decomposition frameworks (Hane and Ratliff 1995, Neves et al. 2007, Boschetto et al. 2008). Rigorous optimization methods generally consist of a single MILP (Mixed Integer Linear Programming) or MINLP (Mixed Integer Nonlinear Programming) mathematical formulations and are usually grouped into two classes: discrete and continuous, depending on the way that volume and time domains are handled. Discrete formulations divide both the pipeline volume into a number of single-product packs, and the planning horizon into several time intervals (Magatão, Arruda, and Neves 2004, Zyngier and Kelly 2009, Rejowski and Pinto 2008, Herrán, de la Cruz, and de Andrés 2010). Most formulations generally use uniform time and volume division. However, a recent paper of Rejowski and Pinto (2008) assumes that each pipeline segment is composed by packs with equal or different pre-specified volumes to account for reductions in the pipeline diameter, and the horizon length comprises time intervals of adjustable duration to allow changes in the pump injection rate. Neves et al. (2007) presented a decomposition approach for the planning of pipeline operations over a monthly horizon. The decomposition relied on a heuristic pre-processing block that accounts for demand requirements, production planning, and typical lot sizes to determine a candidate set of product sequences. In addition, the heuristic block provides time windows for pump and delivery operations at every terminal. Afterwards, the pre-processed information is used by a continuous-time MILP formulation to determine the exact start/finish times of batch input and reception. Since seasonal costs of the electric energy are considered, the model includes binary variables just to avoid pumping operations during high-energy cost periods. Boschetto et al. (2008) reformulated the hybrid approach of Neves et al. (2007) using a different decomposition strategy now involving three blocks: (i) a resource-allocation block determining candidate sequences of batch injections, (ii) a pre-analysis block specifying the precise volumes to be either pumped from source nodes or received in destination nodes, and providing the earliest start/finish times for stripping operations in every destination node, and (iii) a continuous-time MILP model determining the exact timing of pump and delivery operations at each node.

Most of the computational burden in multiproduct pipeline scheduling comes from three difficult tasks: pump sequencing, batch sizing, and batch allocation to receiving terminals. By heuristically choosing them, the remaining operational decisions can be taken in a short CPU time. However, the final pipeline schedule is greatly influenced by those heuristic-based decisions previously taken (Boschetto et al. 2008). In contrast, the available MILP-continuous optimization tools for pipeline scheduling, like the one proposed by Cafaro and Cerdá (2004, 2008), do not require any kind of decomposition nor discretization scheme, and are able to find the optimal input schedule from a single refinery by minimizing the sum of pumping, interface and inventory costs. Moreover, the same authors developed a continuous formulation for managing pipeline networks with multiple inlet points (Cafaro and Cerdá 2009). Nonetheless, they just provide the set of “aggregate” batch stripping operations to be done during every pumping run without specifying the detailed sequence of individual “cuts” performed by the pipeline operator. Generally, there are many ways to distribute the inputted batches among the assigned destinations. Hence, gene-

rating an efficient delivery schedule to accomplish the optimal pipeline operation is a difficult task that has not been solved yet.

1.2 Simulating pipeline operation

In the last decade, few contributions on discrete event simulation of refined products pipelines have been published. Mori et al. (2007) recently developed a simulation model for the scheduling of injection and stripping operations in a real-world pipeline network. The network consists of a series of single pipes that connect multiple refineries, harbors and distribution centers, and transport many oil derivatives. García-Sánchez, Arreche, and Ortega-Mier (2008) presented a hybrid methodology combining tabu search and a discrete-event simulation model for addressing a real-world multiproduct pipeline scheduling problem. In this work, tabu search technique is used to improve unsatisfactory schedules that are easily tested by the simulation model. Product batches are divided into equal sized discrete units whose destination is pre-defined at the time they are injected into the pipeline network.

Instead, our work introduces a discrete event simulation model for a trunk pipeline transporting refined products from a single origin to multiple distribution terminals on segregated or fungible mode. In other words, some products may be interchangeable. The trunk line is made up of a sequence of pipes, each one connecting either an input to an output terminal or just a pair of distribution terminals between themselves. From the discrete simulation viewpoint, the pipeline can be regarded as a coordinated non-traditional multi-server queuing system. The servers perform their tasks in a synchronized manner, with each one having its own queue of fixed-sized batch elements (entities). There is a server at the end of each pipe and its queue is composed by the sequence of batch elements contained in that pipe. Since every pipe should be permanently full of liquid and has a constant volume, the length of any server queue will remain fixed throughout the whole time horizon.

The injection schedule is given to the simulator by solving a rigorous optimization model. Consequently, the simulation model is mainly used to evaluate the efficiency of the schedule provided and also to generate alternative detailed output schedules by considering multiple scenarios and heuristic rules.

2. GENERATING A DETAILED PIPELINE DELIVERY SCHEDULE

2.1 Illustrating example

Figure 1 represents a typical multiproduct pipeline transport system. It consists of a single refinery where oil products are injected, and five distribution terminals at different sites along the pipeline. The first line in Figure 1 depicts the location of every batch inside the pipeline (linefill) at the start of the time horizon. Initially, there are four batches $B4(P2)$ - $B3(P1)$ - $B2(P3)$ - $B1(P4)$, with 400-400-350-350 volumetric units of the products given between brackets, respectively. The following line in the same figure illustrates the pipeline contents after completing the first injection. It consists of 400 units of product $P4$, that are pumped from time 0.00 h to 8.00 h (represented by a right arrow) into the “new” batch $B5(P4)_{new}$. It can also be observed a series of up arrows indicating the associated products deliveries to every terminal. Due to liquid incompressibility, the volume of product injected in the pipeline origin equals the sum of product deliveries to receiving depots, i.e. $400 = 100 + 100 + 100 + 50 + 50$. The optimization model proposed by Cafaro and Cerdá (2004, 2008) not only generates an efficient input schedule but also provides the set of stripping volumes to be transferred during every batch injection. However, as it will be shown later, there are many ways to accomplish the input schedule proposed.

2.2 Hierarchical solution approach

The scheduling problem is hierarchically solved in two stages with the first one generating the input schedule through the optimization module. At the second stage, a detailed delivery schedule based on the information provided by the optimization approach is to be developed. To do that, this work introduces an

efficient discrete event simulation system developed using Arena® (Kelton, Sadowski, and Sturrock 2007) for both the validation of the pipeline schedule provided by the optimization module and the generation of the detailed output schedule. The proposed model permits to visualize the pipeline operations by means of a friendly animation interface showing the dynamics of the pipeline system over time.

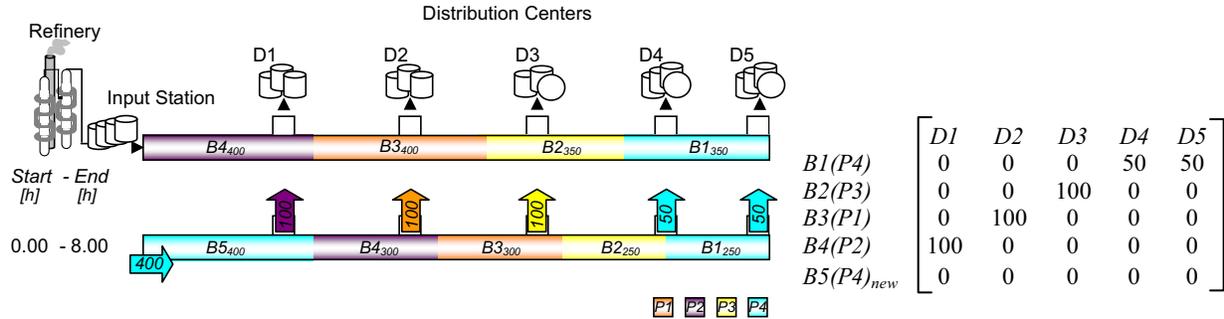


Figure 1. A typical input operation in a multiproduct pipeline system

The simulator uses as input data the set of batches to be stripped and the related number of entities to be delivered to the distribution terminals, while performing a pumping run. It is the so-called terminal-batch assignment matrix $Q_k^o(i,j)$ whose element $q_k^o(i,j)$ represents the product demand at terminal j covered by batch i during pumping run k . For the injection run illustrated in Figure 1, the terminal-batch assignment matrix Q_k is given at the right side.

However, the detailed order of execution of stripping operations is not given. Besides, some of them could be done in two or more non-consecutive steps. Then, some further work should be made before fully developing the pipeline delivery schedule. Such a task will be performed by the proposed simulation model.

Since a pumping run is divided into a sequence of events, and batches flowing through the line are made up of entities, the possible destinations of an entity at an event can be easily derived from the matrix $Q_k^o(i,j)$. When a new input event occurs, therefore, each pipe server knows if the first entity on its queue is eligible for being transferred to the associated terminal. Otherwise, the entity should move to the next pipe. If two or more servers can dispatch the leading entry to their output terminals in order to meet unsatisfied demands, the simulation model should decide, based on priority rules, which one is chosen. To simulate the pipeline schedule provided by the optimization package, the specified stripping volumes for every run k should be delivered and the terminal demands $q_k^o(i,j)$ must be satisfied. At every time event, the simulation model should update the unsatisfied demands $q_k(i,j)$ in every terminal, initially equal to $q_k^o(i,j)$. If $q_k(i,j)$ drops to zero, such a terminal j is no longer eligible to receive an entity from batch i .

When all unsatisfied demands $q_k(i,j)$ are null, the output schedule for run k has been generated. Previous contributions (Mori et al. 2007, García-Sánchez, Arreche, and Ortega-Mier 2008) assumed that the destination for each entity was already given by the optimization package. In our approach, some capabilities have been provided to the proposed simulation model for selecting the route to be followed by every entity based on three key elements: (i) the assignment matrix $Q_k^o(i,j)$ for every run k , (ii) the batch to which each entity belongs, and (iii) a set of priority rules selecting both the leaving entity and the receiving terminal, if several cut operations are eligible for execution. Different pipeline output schedules can indeed be generated by changing the priority rules. The simulation model can also consider operational details like loading and unloading of individual tanks at input and distribution terminals, instead of handling them in an aggregate manner. As a result, the simulation model can track the evolution of the inventory in every individual tank over time. Moreover, pipeline stoppages due to high-cost peak periods can easily be handled.

3. PROBLEM STATEMENT

Given: (a) a multiproduct pipeline connecting an oil refinery to several distribution terminals; (b) the number and type of products to be transported through the pipeline; (c) the set of product batches to be introduced in the pipeline origin (input schedule); (d) the associated set of stripping volumes to be transferred during every batch injection (terminal-batch assignment matrix $Q_k^o(i,j)$); (e) the scheduled production runs to be loaded into the tanks of the input station; (f) the initial pipeline conditions (sequence of batches inside the line at $t = 0$ and their sizes); (g) the initial inventory level of every product in terminal tanks; (h) the hourly product demand profile at the distribution centers; (i) the constant product pumping rate; and (j) the horizon length.

The detailed delivery schedule is next generated by explicitly defining:

1. The sequence of batch portions to be pumped into the pipeline;
2. The size of every portion and the starting/end times of the related injections;
3. The amount and type of product delivered to a storage tank from a batch arriving to an output terminal, during every injection;
4. The time at which a batch portion has been completely loaded in the terminal tank;
5. The product inventory management at the delivery terminals by considering discharged product lots and client demands on a hourly basis.

3.1 Major assumptions

In order to develop a pragmatic representation of the real-world problem, next assumptions are considered in the discrete-event simulation model:

- A unidirectional pipeline connecting a single refinery to multiple distribution terminals is considered. However, the model can be extended to manage network topologies.
- The pipeline is always full of liquid products and operates either on segregated or fungible mode. On the latter mode, a single batch can have many destinations.
- Batches of products are injected into the pipe one after the other, with no physical barrier between them.
- The interface or contamination loss between a particular pair of refined products is a known constant.
- Due to liquid incompressibility, every time an element of a batch is injected, one and only one entity already in the line is simultaneously transferred from the pipeline to a single receiving terminal.
- Every batch portion is pumped at a fixed flow rate.
- Distribution centers are tank farms with dedicated storage units of known capacity for each product.
- At most one terminal tank is connected to the pipeline at any time event, and the setup time for switching from one tank to another is negligible.
- Refinery production schedules have been previously developed. Scheduled start/end times and rates of incoming product flows to storage tanks at the input station are problem data.
- Daily client demands are deterministic data given on an hourly basis.

3.2 Problem objective function

The sequence in which product deliveries to distribution terminals are accomplished has great impact on the system operational costs. In particular, pipeline stoppages are quite expensive (Hane and Ratliff 1995) and should be avoided. A pipe stoppage occurs whenever a delivery at some terminal is interrupted and a different stripping operation starts at an upstream (nearest-to-refinery) point. This causes the interruption of the flow in the pipeline segment connecting the activated and the deactivated output nodes, and the consequent shutdown of several pump stations. The main cost of a stoppage is associated with the loss of

energy of the fluid momentum, since the stopped flow will be surely put in motion again for resupplying downstream destinations. In addition to the energy cost, the maintenance cost increases with the number of stoppages, given that the time between pump repairs strongly depends on the number of shutdowns. To measure the quality of the resulting output schedule, the so-called *accumulated idle volume* is defined. This variable is computed by adding the product volumes in idle pipes throughout the complete horizon. The accumulated idle volume together with the total number of cut operations required to meet the specified terminal demands are the two performance measures used to compare alternative pipeline output schedules.

4. PROPOSED SIMULATION MODEL STRUCTURE

A refined products pipeline network is primarily composed of an input station, a set of receiving terminals and pipes. Tank storage facilities, such as the input station and the receiving terminals, are connected through pipes. Once oil products arrive to the input station, coming from neighboring refineries, they are temporarily stored in tanks until the execution of injection runs. An injection run implies the transfer of material from the tanks of the input station to the pipeline system. The type of product injected, the batch size and the expected pumping rate should also be specified. As a result, each injection run creates a new batch that will be pumped into the line from the input station. The sequence of injection runs determines the input schedule. After a product batch has been introduced, the material moves forward due to the injection of new batches at the input station. While doing so, another product batch already in transit into the line is simultaneously transferred from the pipeline to a single receiving terminal, due to liquid incompressibility.

4.1 Main model structure

The simulation of pipeline operations is based on a list of time events, each one representing the injection time of a single element into the system. The event list is generated by the simulation model scheduler block accounting for the pipeline input schedule. In our case, the optimization approach of Cafaro and Cerdá (2004, 2008) permits to generate the input schedule for the following month, indicating the sequence of batches to be injected and the batch attributes (product, volume, pump rate, and start pumping time). Since product batches are to be handled in a discretized manner, their volumes are expressed as a number of small, equal-size batch elements called entities. Entity attributes are inherited from the batch to which it belongs. Similarly, the linefill of a pipe consists of a sequence of entities defining the queue of the pipe server. Each event represents the pumping of a single entity into the line. Therefore, the state of the pipeline system just changes when an event is accomplished.

Each time an elementary pumping operation is run at the input station (i.e. at the inlet of the first pipe) a new entity enters the first-server queue, and another entity at the exit of one of the pipes should be dispatched to the corresponding receiving terminal. The set of servers should jointly decide which one will dispatch the first entity on its queue to the associated distribution terminal (dispatching server). Moreover, it will be determined which servers should transfer the entity to be “serviced” to the next pipe (servers located upstream of the dispatching server) and which ones will remain idle because there are no new arrivals (servers located downstream of the dispatching server).

In other words, a pipe server can take three different actions whenever an event is accomplished: (a) it remains idle because there is no product arrival to its queue; (b) it transfers the first entity waiting for service to the next pipe, or (c) it delivers the first entity on the queue to its receiving terminal. Actions (b) and (c) are triggered by a new entity arrival to the server queue. As a result, just a single distribution terminal will be active on every pumping event. The service time of an entity is defined in terms of two pieces of information: the entity volume (a user choice) and the pumping rate for the current injection. After the servers perform their jobs, the simulation clock is advanced to the next event time. Delays can arise if the selected distribution terminal does not have enough storage capacity to receive the departure entity. Such pipeline “blockages” lead to the so-called disrupted pipeline operation.

4.2 Simulation blocks

The model structure involves three blocks, each one representing a main component of the pipeline system: the input station, the receiving terminals, and the pipes. Figure 2 shows the model blocks together with key simulation elements, such as entities (batch elements) and resources.

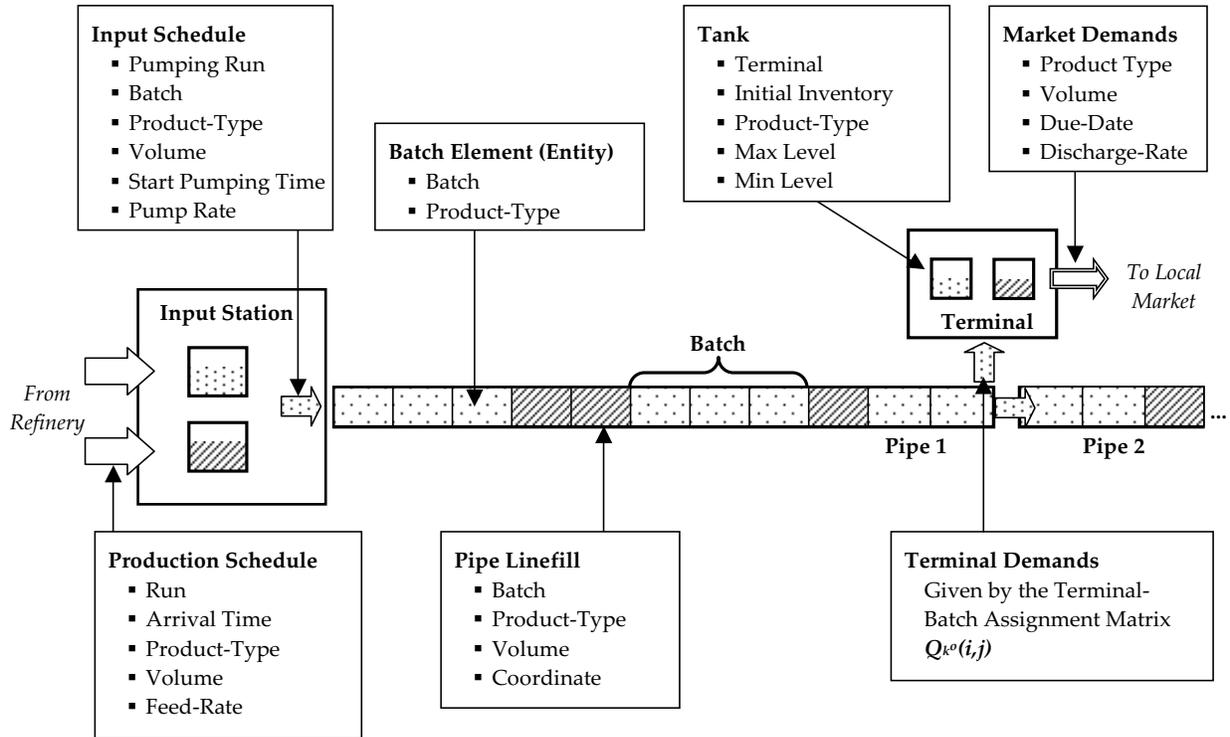


Figure 2. Major components of the discrete-event simulation model

4.2.1 Input station

The *input station* is located at the origin of the pipeline system. Product batches are discharged from input tanks and injected into the line, while some production runs from the neighboring refineries are being loaded. The input station is composed of storage tanks and pumps, and the simulation system forces to stop the pipeline activity when the available stock is running out.

Operations at the input station are modeled through different components. The first one is the production schedule. New products arrivals (production runs) are sent to storage tanks at pre-defined time events. The inventory level in every storage tank is modeled as a continuous variable, allowing to accurately control the evolution of product stocks at the input station.

Since product batches are to be handled in a discretized manner, their volumes are expressed as a number of small, equal-size batch elements called entities. Each entity represents a specific volume of a certain product. If the entity size decreases, a larger number of batch elements will be required, and a more accurate model will be achieved. Hence, a major decision is to choose a proper entity volume.

The second operational module of the input station is the input schedule. Such program indicates the sequence of batches to be injected into the line and the batch attributes, i.e., product type, volume, mean pump rate, and start pumping time. Entity attributes are inherited from the batch to which it belongs.

Finally, a key decision to be made is the pump rate at which batch elements are to be injected. Design and operational constraints should be taken into account. Pressure limits have to be respected, and turbulent flow should be maintained in order to minimize product interface volumes. A main term of the trans-

portation cost function is associated to the energy consumed for moving product batches into the line. The pumping cost is strongly affected by the pump rate of the batches. This is the reason why the proposed simulation model is able to choose different pump rates for different batch parts or entities, a critical matter to accurately represent the operation of a real pipeline system. Moreover, oil pipeline operators avoid running pump stations at daily peak periods because a much higher energy price must be paid for the electrical power consumption. The pipeline simulation system can easily account for high-pumping cost intervals.

4.2.2 Pipes

The pipeline system is divided into different segments (pipes) that connect the nodes of the distribution network. Each pipe is modeled as a fixed-size FIFO queue with a single server at the pipe extreme that permits the movement of material entities from one position to another. By combining different pipes, with particular sizes and service rates, the model is able to simulate the operation of almost every pipeline system structure.

Every time an entity enters a pipeline segment at the inlet point, it pushes the entity positioned at the other extreme out of the queue. In other words, at every pipe end, a server is located that dispatches the first entity on the queue whenever a new entity enters the pipe at the inlet section. Since the system transports multiple products, the model keeps track of every pipe linefill by updating the server queues at every time event. The actions that a pipe server can take on the entity waiting for service are: (i) no action; (ii) move to the next pipe; (iii) load in a terminal tank.

At the beginning of the scheduling horizon it is necessary to create the initial condition of the pipeline contents, which is basically carried out by the initialization process. The volume of each pipe is determined by the number of entities it holds. This number remains constant all over the scheduling horizon, as the volume of each pipe is a fixed quantity given by the cross section and the segment length.

4.2.3 Terminals

Terminals are tank-farms from which products are sent to consumer markets. In the proposed simulation model, arrivals of product entities from the line and deliveries to regional markets are simultaneously handled.

The tanks have a maximum level which cannot be exceeded and, in general, a minimum level of material is also required. As it was previously mentioned, continuous variables are used to model the inventory level in storage tanks.

The pipeline output schedule is automatically generated by the simulation model according to the requirements of each terminal. Terminal requirements are given by the terminal-batch assignment matrix $(Q_k^o(i,j))$ resulting from the optimization module.

On the other hand, product batches are sent to local markets from terminal tanks mainly by truck. Market demands are continuously updated according to delivery due-dates and tank discharge rates. Thus, it is possible to define a daily or hourly pattern demand. Also, empirical/probabilistic distributions can be used to represent stochastic demands.

4.3 Priority rules

When a product batch entity reaches the position of a terminal that demands it, the entity may be transferred from the pipeline to an available terminal tank. At every pumping event, only one terminal can receive a single product unit from the pipeline. Hence, every material entity in the pipeline located between the input station and the selected terminal moves on, while the rest of the pipeline segments remain idle. In order to decide which of the eligible servers should dispatch the first entity on its queue to the associated terminal, the simulation model can easily apply alternative heuristic rules assigning different priori-

ties to terminals, so as to choose the one that should receive the demanded products first. Three priority rules were applied to solve the case study introduced in the next section:

(a) The *Nearest-First* (NF) rule prioritizes the product delivery to the eligible terminal closest to the origin. If two alternative terminals are able to receive a product unit, the one positioned closer to the beginning of the line will be selected. As a result, no required product will overpass a demanding terminal since upstream locations are always prioritized. In this way, we make sure that the *NF* rule will always generate a feasible output schedule.

(b) The *Farthest-First* (FF) rule prioritizes product deliveries to the farthest eligible terminal from the origin. In this case, it is necessary to verify if the delivery of an entity to the prioritized terminal does not prevent from satisfying product demands at some upstream depot. Since a unidirectional pipeline is considered, an entity that overpasses a terminal location can no longer be transferred to it. If such an entity is absolutely necessary to meet a depot demand, the resulting output schedule would be infeasible. When a terminal T_j is “restrictive”, no other eligible terminal located farther from the origin can be the selected destination, and only terminals T_1 to T_j can receive a product entity.

(c) The *Nearest to current terminal* (NC) rule prioritizes the product delivery to the eligible terminal closest to the one being currently served. If two alternative terminals are able to receive a product unit, the one positioned closer to the terminal that is being served will be selected. Obviously, the current terminal has the highest priority. Prioritizing the nearest-to-current terminal, an important reduction in the total volume of idle pipeline segments and a reduction in the number of stripping operations (turning on/off pump stations) can be simultaneously obtained.

In sum, by using different priority rules, the logic structure of the model generates alternative delivery schedules considering the current state of the pipeline (product availability and batch location) and the depots requirements. The delivery schedule is given by the sequence of transfers of material entities from the pipes to receiving terminals over time.

5. CASE STUDY: A REAL-WORLD PROBLEM OF THE OIL INDUSTRY

The proposed simulation-based model has been applied to the solution of a real-world problem first introduced by Rejowski and Pinto (2003) and later solved by Cafaro and Cerdá (2004, 2008). This challenging scheduling problem addresses the daily operation of the OSBRA pipeline owned by the Petrobras company, which transports almost 20% of the total Brazilian oil derivatives. OSBRA is the most important Brazilian pipeline, which connects the REPLAN refinery located in Sao Paulo with five distribution centers situated in the cities of Ribeirao Preto, Uberaba, Uberlandia, Goiania and Brasilia. The pipeline system transports four refined products (gasoline, diesel, LPG and jet-fuel) to five distribution terminals along 955 km. The model makes use of the monthly input schedule generated through the optimization module proposed by Cafaro and Cerdá (2008). The pipeline system is divided into relatively small entities whose volume is fixed to 100 m³. Thus, a total of 1635 entities are required to fill the whole line.

5.1 The proposed animated interface

Figure 4 depicts the simulation-based interface developed using the Arena® simulation package. The animated interface allows the visualization of the pipeline operations, showing the dynamic evolution of the pipeline network state over time. The software Arena® permits the development of graphic representations to assess the model operation in real time. The main components of the pipeline system are also depicted in the same Figure 4: the trunk pipeline, the input station and the receiving terminals. The current “status” of all the pipeline segments is given by the product entities moving along the different queues.

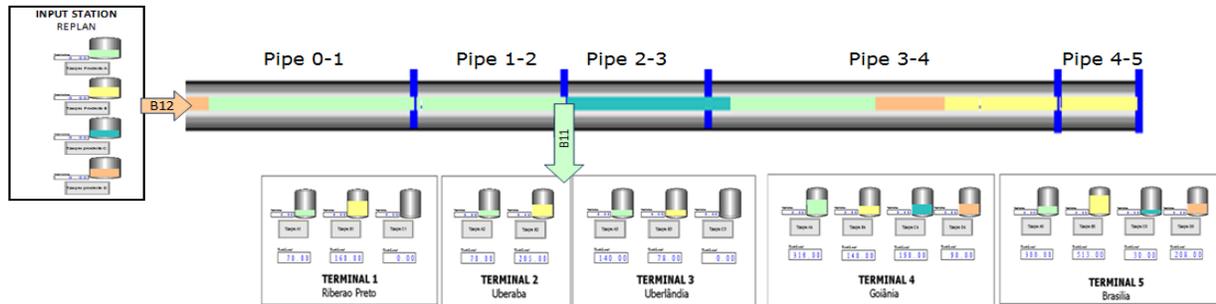


Figure 4. View of the model animation interface

Each entity represents a fixed volume of a certain product, located in a specific place of the pipeline network at a given time event. Through this animation interface it is easy to follow the evolution of the pipeline content during the scheduling horizon. In addition, the inventory level in storage tanks at both the input station and the receiving terminals is also traced by the model animation interface. As seen in Figure 4, two arrows, one at the input station and the other at the active receiving terminal show the new batch being injected ($B12$) and the in-transit batch being stripped ($B11$), respectively. Moreover, global model variables, like the objective function (accumulative idle volume) and unsatisfied terminal requirements ($q_k(i,j)$) can be easily plotted in the animation interface, enhancing the comprehension of the model dynamics.

Alternative delivery schedules for a monthly horizon can be easily generated and tested in less than one minute of CPU time by using the proposed simulation-based model with different priority rules.

5.2.1 NF Priority Rule – First week delivery schedule

Figure 5a represents the delivery schedule generated by using the *NF* rule for the first week of the monthly horizon. From $t = 5$ h to $t = 14.95$ h, a portion of batch $B4$ is transferred to terminal $T2$, while the initial portion of $B6$ is injected at the input station. At $t = 14.95$ h, the planned delivery is completed, and $B4$ is now stripped to $T3$. Pipe 2-3 is activated, and the other pipes 0-1 and 1-2 remain active during the second “cut” operation. At time $t = 21.59$ h, pipe 3-4 is also activated and $T4$ begins to receive gasoline from batch $B2$. Almost 9 hours later ($t = 30.44$ h), pipe 4-5 is set in motion and $T5$ begins to receive diesel fuel from batch $B1$. Once the transfer of $B1$ is completed ($t = 43.71$ h) the first pipe stoppage in the program occurs. Terminal $T4$ begins to receive LPG from batch $B3$ and pipe 4-5 is deactivated. The liquid content in that pipeline segment is stopped and the accumulated idle volume is increased by $13,500 \text{ m}^3$.

The overall accumulated idle volume is the result of summing all the stopped pipe volumes over the scheduling horizon. Using the *NF* rule, 20 cut operations are required during the first week, and the overall accumulated idle volume amounts to $141,000 \text{ m}^3$.

5.2.2 FF Priority Rule – First week delivery schedule

Figure 5b depicts the delivery schedule given by the *FF* rule for the first week. Using the *FF* rule, 20 cut operations are also needed. However the accumulated idle volume reaches up to $286,000 \text{ m}^3$, considerably higher than the $141,000 \text{ m}^3$ obtained with the *NF* rule. This difference is reduced over the following weeks of the monthly horizon.

5.2.3 NC Priority Rule – First week delivery schedule

In turn, Figure 5c shows the delivery schedule achieved with the *NC* rule for the first weekly period. As seen, only 16 cut operations are needed and the accumulated idle volume is also reduced to $100,500 \text{ m}^3$. This implies savings of 28.7 % and 64.9 % with regards to the results obtained with the *NF* and *FF* rules,

respectively. The substantial reduction in the pipeline idle volume is even more evident throughout the monthly horizon.

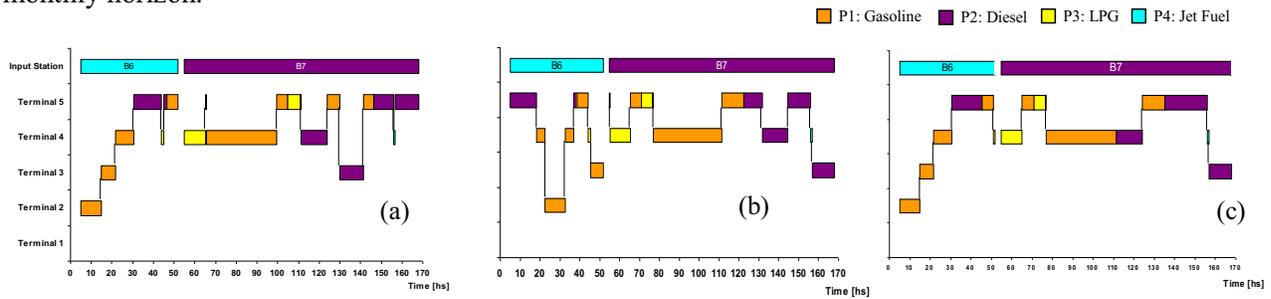


Figure 5. Delivery schedule using the (a) *NF*-rule, (b) *FF*-rule, and (c) *NC*-rule for the first week of a monthly horizon

5.2.3 Monthly delivery schedule

The complete monthly delivery schedules obtained with the *NF*, *FF* and the *NC* rules are illustrated in Figures 6a, 6b and 6c, respectively. The first line in these figures shows the pumping runs introducing batches *B6* to *B14* at the input station. The other lines depict the receiving tasks at every terminal while executing the corresponding batch injections. Both delivery schedules are then assessed according to the number of operations required to fulfill the product requirements, and the accumulated volume of idle product throughout the planning horizon. By using the *NF* priority rule for the whole monthly horizon, the accumulated volume of idle product is 1,404,500 m³, and 65 cut operations are performed. When the *FF* rule is applied, the total volume of idle product rises to 1,472,500 m³, but a slightly lower number of cut operations are required: 63. Finally, when the *NC* rule is used, the accumulated volume of idle product descends to 835,000 m³ and the delivery schedule is completed in only 55 operations.

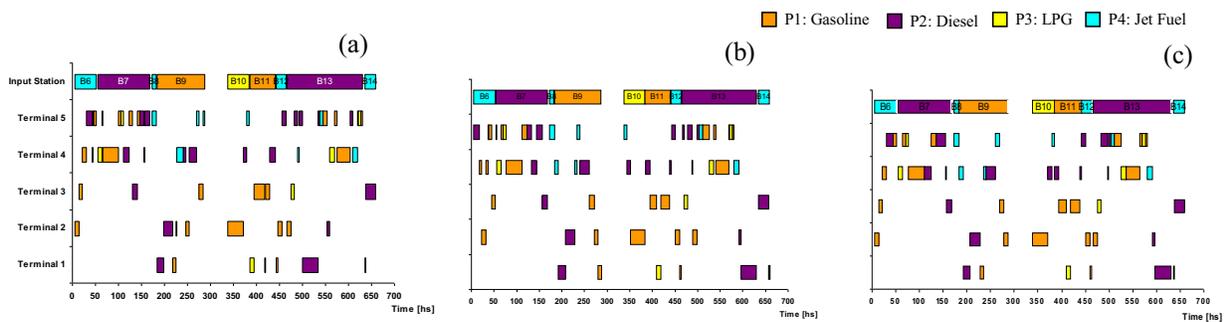


Figure 6. Delivery Schedule using the (a) *NF*-rule, (b) *FF*-rule, and (c) *NC*-rule for the monthly horizon

6. CONCLUSIONS

This work addresses the efficient management of one of the most critical components in the petroleum supply chain: refined products pipeline networks. An advanced discrete event simulation tool has been developed. The novel approach is very useful for validating operational pipeline schedules provided by rigorous optimization techniques. At the same time, it allows generating and testing alternative monthly product delivery schedules in less than one minute of CPU time, particularly for single-source multiple-destinations pipeline networks. Results show that different priority arrays lead to significantly different delivery schedules, strongly affecting the cost-efficiency of the solution achieved. For instance, prioritizing nearest-to-refinery terminals may lead to a reduction in the total volume of segments stopped, but the number of stripping operations (turning on/off pump stations) will surely increase. On the other hand,

prioritizing the farthest terminals may cause an important growth in the number and volume of pipeline stoppages. It is quite clear that these very simple rules will rarely lead to the "optimal" delivery schedule.

However, the development of a new rule that prioritizes the product delivery to the eligible terminal closest to the one being currently served returns an improved solution. This rule generates an alternative schedule that significantly reduces the total accumulated idle volume (about a 40 % with respect to the other rules), and there is a notable decrease in the number of cut operations.

Furthermore, the proposed logic structure permits to generate other detailed delivery schedules embedding alternative heuristic methods based on different terminal priority arrays. In addition, it allows the visualization of the dynamic evolution of the system over time, using a friendly animated interface. The proposed approach can be easily extended to permit the use of emerging simulation-based optimization tools in order to improve the performance of the resulting schedules. Future work will be focused on developing efficient priority rules combined with heuristic search and rigorous formulations to find out cost-efficient and robust solutions for the detailed scheduling of multiproduct pipeline networks with different configurations.

ACKNOWLEDGEMENTS

Financial support received from FONCYT-ANPCyT under Grant PICT 2006-01837, from CONICET under Grant PIP-2221, from Universidad Nacional del Litoral, Argentina, under Grant PI-66-337 and from AECID under Grant PCI D/024726/09 are fully appreciated.

REFERENCES

- Boschetto, S.N., L.C. Felizari, L. Yamamoto, L. Magatão, S.L. Stebel, F. Neves-Jr, L.V.R Arruda, R. Lüders, P.C. Ribas, and L.F.J. Bernardo. 2008. An integrated framework for operational scheduling of a real-world pipeline network. *Computer Aided Chemical Engineering* 25:259-264.
- Cafaro, D.C. , and J. Cerdá. 2004. Optimal scheduling of multiproduct pipeline system using a non-discrete MILP formulation. *Computers and Chemical Engineering* 28:2053-2068.
- Cafaro, D.C., and J. Cerdá. 2008. Dynamic scheduling of multiproduct pipelines with multiple delivery due dates. *Computers and Chemical Engineering* 32:728-753.
- Cafaro, D.C., and J. Cerdá. 2009. Optimal scheduling of refined products pipelines with multiple sources. *Industrial & Engineering Chemistry Research* 48:6675-6689.
- García-Sánchez, A., L.M. Arreche, and M. Ortega-Mier. 2008. Combining simulation and tabu search for oil-derivatives pipeline scheduling. *Studies in Computational Intelligence* 128:301-325.
- Hane, C.A. and H.D. Ratliff. 1995. Sequencing inputs to multi-commodity pipelines. *Annals of Operations Research* 57:73-101.
- Herrán, A., J.M. de la Cruz, B. de Andrés. 2010. A mathematical model for planning transportation of multiple petroleum products in a multi-pipeline system. *Comp. & Chem. Eng.* 34:401-413.
- Kelton, W.D., R.P. Sadowski, and D.T. Sturrock. 2007. *Simulation with ARENA*. Fourth Edition. New York: McGraw-Hill.
- Magatão, L., L.V.R. Arruda, and F. Neves-Jr. 2004. Mixed integer programming approach for scheduling commodities in a pipeline. *Computers & Chemical Engineering* 28:171-185.
- Mori, F.M., R. Lüders, L.V.R. Arruda, L. Yamamoto, M.V. Bonacin, H.L. Polli, M.C. Aires, and L.F.J. Bernardo. 2007. Simulating the operational scheduling of a real-world pipeline network. *Computer Aided Chemical Engineering* 24:691-696.
- Neves-Jr, F., L. Magatão, S.L. Stebel, S.N. Boschetto, L.C. Felizari, D.I. Czaikowski, R. Rocha, and P.C. Ribas. 2007. An efficient approach to the operational scheduling of a real-world pipeline network. *Computer Aided Chemical Engineering* 24:697-702.
- Rejowski, R., and J.M. Pinto. 2003. Scheduling of a multiproduct pipeline system. *Computers and Chemical Engineering* 81:433-466.

- Rejowski, R., and J.M. Pinto. 2008. A novel continuous time representation for the scheduling of pipeline systems with pumping yield rate constraints. *Computers and Chemical Engineering* 32:1042-1066.
- Sasikumar, M., P.R. Prakash, S.M. Patil, and S. Ramani. 1997. Pipes: A heuristic search model for pipeline schedule generation. *Knowledge-Based Systems* 10:169-175.
- Zyngier, D., and J.D. Kelly. 2009. Multi-product inventory logistics modeling in the process industries. In *Optimization and logistics challenges in the enterprise. Springer optimization and its applications*, ed. W. Chaovalitwongse, K. C. Furman, and P. M. Pardalos, 30:61-95. New York: Springer.

AUTHOR BIOGRAPHIES

VANINA G. CAFARO is a PhD. Student in the Center for Advanced Process Systems Engineering (CAPSE) at the INTEC research institute located in Santa Fe, Argentina. She obtained her B.S. degree in Industrial Engineering from Universidad Nacional del Litoral (UNL) in Argentina. Her research interests include production planning and scheduling of pipelines networks, optimization and discrete-event simulation. Her email address is [`<vcafaro@fiq.unl.edu.ar>`](mailto:vcafaro@fiq.unl.edu.ar).

DIEGO C. CAFARO is a Professor of Industrial Engineering at Universidad Nacional del Litoral (UNL), and Posdoctoral Fellow at the Argentine National Scientific and Technical Research Council (CONICET). He received his Ph. D. in Chemical Technology from UNL in 2008, and his B.S. in Industrial Engineering from Universidad Nacional del Litoral in 2003. His research contributions have been in the areas of oil pipeline planning and scheduling. His teaching interests include courses on Modeling and Optimization in Process Systems Engineering. His email address is [`<dcafarro@fiq.unl.edu.ar>`](mailto:dcafarro@fiq.unl.edu.ar).

CARLOS A. MÉNDEZ is a Professor of Industrial Engineering at Universidad Nacional del Litoral (UNL) in Argentina and Associate Researcher at the Argentine National Scientific and Technical Research Council (CONICET) in the area of Process Systems Engineering. He obtained his B.S. degree in Information Systems Engineering from Universidad Tecnológica Nacional (UTN) in Argentina and his PhD in Engineering from UNL. He has published over 100 refereed journal articles, book chapters, and conference papers. His research and teaching interests include production planning and scheduling, optimization, logistics and discrete-event simulation. His email address is [`<cmendez@intec.unl.edu.ar>`](mailto:cmendez@intec.unl.edu.ar).

JAIME CERDÁ is a Professor of Chemical and Industrial Engineering at Universidad Nacional del Litoral (UNL), and Superior Researcher at the Argentine National Scientific and Technical Research Council (CONICET). He received his Ph. D. in Chemical Engineering from Carnegie-Mellon University in 1980, his M.S. in Chemical Engineering from Carnegie-Mellon University in 1979, and his B.S. in Chemical Engineering from Universidad Nacional del Litoral at Argentina in 1968. He is a former Industrial Engineering Department Head of the School of Chemical Engineering at UNL. His research contributions have been in the areas of planning and scheduling of batch and continuous processes, energy integration, process synthesis, supply chain operational management, oil pipeline planning and scheduling, dynamic vehicle routing and pickup & delivery problems. His email address is [`<jcercerda@intec.unl.edu.ar>`](mailto:jcercerda@intec.unl.edu.ar).