

A Continuous Nonlinear Approach To Mixed-Integer Problems Based On Complementarity Constraints

Fabio Dias Fagundez, João Lauro Dorneles Facó, Adilson Elias Xavier

Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil
fabio.fagundez@ufrj.br; jldfacó@ufrj.br; adilson@cos.ufrj.br

1. Abstract

Certain mixed-integer linear programming (MILP) models can be reformulated as continuous nonlinear (nonconvex) programming (NLP) models with the use of complementarity equations. This equivalence can be employed to solve many problems, which were originally formulated as MILP's, but later on reformulated as nonlinear programs. It is important to notice that any NLP feasible point can be directly mapped to an integral MILP feasible point, in such a manner that any NLP local optimum will define an upper bound to the original MILP minimization problem (or a lower bound, if in a maximization problem). Moreover, it may be faster to converge to an NLP local optimum with efficient NLP methods than to find an integral MILP solution, even if sub-optimal. Therefore, in addition to simply solving the NLP, we propose the following approach to fasten the convergence to the (global) optimum of an MILP problem: (i) solve the NLP - perhaps with different initial points, (ii) feed the MILP with the (best) NLP solution; (iii) solve the MILP, bounded by the NLP solution. Notice that when solving the MILP within a branch-and-bound procedure, it might be interesting to find new upper bounds during the visitation of a new branched node - for instance, if the integrality gap is still big. For each one of these nodes, exists an equivalent NLP formulation, with less constraints than the original one, as more binary variables are set. Examples cases from the scheduling fields are discussed to illustrate the proposed approach.

2. Keywords: scheduling, nonlinear programming, mixed-integer programming

3. Introduction

Mixed-Integer Linear Programming (MILP) is a widely employed technique for solving engineering and economics problems. Due to the combination of Linear Programming (LP) efficient methods, suitable cut generators and clever enumeration procedures, larger MILP instances are being solved day after day. Unfortunately, no matter how efficient is one's LP method or how tight is one's cut, the discrete optimization problem is fundamentally an enumerative problem. This kind of problems suffers from the curse of dimensionality, featuring exponential growth of possible solutions from a linear growth of binary variables. It has been previously shown that discrete problems can be replaced by continuous complementarity problems, which do not suffer from exponential growth, although being highly nonconvex [1]. Complementarity problems can be tackled by nonlinear programming (NLP) methods [2], although subject to converging to local optima as they are nonconvex. A problem when modeled as an MILP will present a global optimum which can be found after possibly unreasonable computing time, whereas if modeled as an NLP it will converge to a local optimum after an acceptable computing time. When an MILP is being solved within a branch-and-bound framework, the larger the gap between the current LP solution and the best integral MILP solution found so far, the harder is the problem. The NLP model can act as a shortcut to find integral solutions at each branch-and-bound node. An NLP feasible point is equivalent to an MILP feasible point. Therefore, an NLP local solution is equivalent to an integral MILP feasible point, defining an upper bound (if solving a minimization problem) on the correspondent MILP, which can improve the pruning of the branch-and-bound procedure. In fact, the NLP solution is a valid solution for the mixed-integer problem, and may be kept as *the* solution in a real-world situation or be used as an incumbent. In this paper we apply this solving strategy to crude oil scheduling problems.

This paper is divided as follows. Section 4 defines crude oil scheduling as an example of a mixed-integer problem in order to illustrate the proposed approach. Section 5 presents the solving strategy and preliminary computational results. Section 6 features our concluding remarks. It is worth to notice that the equations presented in section 4 can be employed in general process scheduling problems, not being limited to crude oil scheduling.

4. Crude Oil Scheduling

Scheduling problems in the chemical process industry can be modeled as mixed-integer optimization problems, where constraints on discrete variables stand for assignment and sequencing decisions, and continuous equations model mass, volume, energy or component balances. Floudas and Lin [3] recently presented a survey on process scheduling, where they emphasize the importance of MILP in this field. The guarantee of global optimality is considered as the highlight of this approach. However, due to their NP-completeness, such models suffer from the curse of dimensionality, and may reach unacceptable computational times to find a solution for a real-world problem. Therefore, the research community has been constantly working on formulations to reduce models' dimensions, particularly within nonuniform time-discretization frameworks (see [4] for a thorough discussion on this subject).

The schedule is a dynamic system, where one action at a given time instant impacts the future states of the system. The actions one perform are the control variables, while the measurable system features are the state variables. State equations explicit how a state is calculated from previous controls and states. The scheduling of crude oil is a complex task, featuring nonlinear (due to crude blending) and combinatorial (due to assignment and sequencing) aspects. It can be stated as how to define: (i) ship allocation on the port jetties; (ii) transfer operations between ships, tanks, process units, and pipelines; (iii) sequence of pipeline parcels (end products and crude oil), in such a manner that an objective cost function is minimized and operational constraints are respected.

The logistic system can be divided in three main subsystems (Figure 1): port, distribution center, and refinery, all of them connected by pipelines [5]. It is also possible to consider a single system, when the port jetties are directly connected to refinery crude tanks [6][7]. In addition to these three systems, one may possibly consider a fourth system: the tanker fleet, whose schedule updates the estimated times of arrival (ETA) for each ship. According to Shah [8], a reasonable approach is to solve the systems hierarchically. We follow this approach in this paper, considering two systems: the port (tankers, jetties, tanks, and pipelines) and the refinery crude area

(pipelines, tanks, and distillation crude unit). However, it is important to mention that the equations presented herein could be employed in other arrangements as well. Portside tanks serve as a buffer to keep the pipelines in continuous operation, even when tankers are late. In general, a (refinery or portside) tank stores a certain class of crude (e.g. heavy oil tanks cannot store light oil). Ideally, a good schedule will use a small number of tanks, but it is important to notice that inventory costs are secondary when compared to the cost of not meeting the refinery production plan or delaying the ships. The refinery's demand for crude oil (as well as derivatives production) must be met by the port scheduling. Jetties can be restrictive on what tankers and cargoes to handle, according to their dimensions (draught and length) and pumping capacity. A ship must berth, unload, and leave the port within a time window defined by contract, otherwise the oil company will pay heavy demurrage fees. For instance, Brazilian demurrage costs amounted to US\$ 1.5 billion in 2006 [9]. Therefore, the port schedule's main objective is to minimize demurrage costs, while keeping the refinery plan. A jetty is available for berthing only after the previous ship had enough time to leave the port. In the refinery side, the crude distillation unit operates continuously, around an operational feed flow. Blending is not allowed in the lines, i.e., each transfer operation has only one source equipment and one destination equipment at a given time. Running tanks are not allowed either, i.e., a tank cannot receive and send crude simultaneously. In fact, a tank can make a delivery to another equipment (e.g. pipeline or crude distillation unit) only if the necessary "idle time" has been observed (e.g. to separate brine from crude oil or to assure a lab analysis).

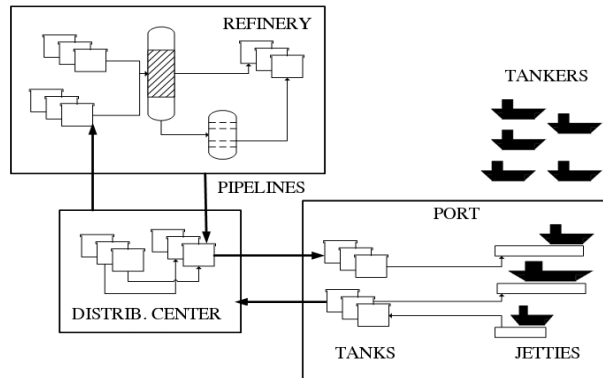


Figure 1. Crude oil logistic system

Logistic systems can be interpreted as graphs, defined during the problem's formulation, featuring equipments as nodes, connected by *flow arcs*. A system graph is built by checking which equipments are connected by pump lines, and which are compatible in terms of crude oil and physical dimensions. Figure 2 illustrates a port infrastructure with 3 jetties, 5 tanks (3 for end products, the other 2 to send crude oil), 2 pipelines connecting the port to a refinery (one to receive end products, the other to send crude oil), and 3 tankers that must be scheduled. In this example, tanker N_3 can berth on jetties P_3 and P_2 , but cannot berth on jetty P_1 . Moreover, N_3 's cargo is a crude oil that can be pumped to tank T_5 . Therefore, there is a flow arc (represented by the lower traced line) between tanker N_3 and tank T_5 , through jetty P_3 . The fundamental scheduling activity is the transfer operation, which is made up by a pair of equipments (source-destination) connected by an arc, and a flow from the source to the destination. The control vector $u(t_i)$ is the vector whose each entry $u_j(t_i)$ stands for a nonnegative flow on arc j at time t_i . The optimization problem is to define a feasible sequence of $u(t_i)$, for all instants t_i , which minimizes the objective function J . All control variables $u_j(t_i)$ are bounded. Operational constraints, such as "one equipment N cannot be the destination of two transfer operations at the same time t_i , in order to avoid inline blending" can be modeled by different manners.

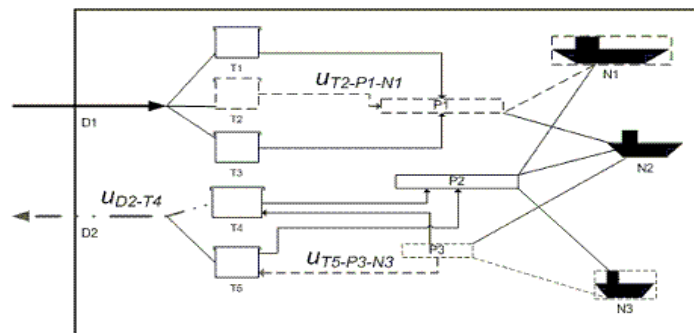


Figure 2. Graph example

Table 1 shows two modeling possibilities, with A_N as the set of indexes for all arcs whose destination is N : (a) the MILP formulation, as commonly found in the literature [7][8], and (b) the NLP formulation proposed here. As N can participate in at most one transfer operation at time t_i , either all flows in the A_N arcs are zero at t_i (no transfer happens with destination N at time t_i), or only one flow is greater than zero at t_i (N is the destination of only one transfer operation at time t_i). Both formulations enforce this behavior.

Table 1. Mixed-Integer and Complementarity Models

(a) Mixed-Integer	(b) Complementarity
$\sum_j b_j(t_i) \leq 1$	$\sum_{k \in A_N} \sum_{j > k \in A_N} u_k(t_i) u_j(t_i) = 0$
$0 \leq u_j(t_i) \leq b_j(t_i) u_j^{\max}(t_i)$	$0 \leq u_j(t_i) \leq u_j^{\max}(t_i)$
$u_j(t_i) \in \mathbb{R}^n, b_j(t_i) \in \{0, 1\}, j \in A_N$	$u_j(t_i) \in \mathbb{R}^n, j \in A_N$

Model (a) requires an additional control vector $b(t_i)$ of binary variables, whose each entry $b_j(t_i)$ is associated to the $u_j(t_i)$ entry. If $b_j(t_i)$ is set to 1, then a positive flow is allowed on arc j ; otherwise (if set to zero), no flow is allowed on arc j . This is assured by the manipulation of the bounds on $u_j(t_i)$: if $b_j(t_i) = 1$, the bounds are preserved; otherwise, they are set to zero. The summation constraint guarantees that at most one $b_j(t_i)$ can be evaluated as 1 (one) at $t_i, j \in A_N$. All others binary variables associated to A_N must be evaluated as zero. Model (b) relies on the control vector $u(t_i)$ only. There is no need for additional binary variables. The summation of the products of all A_N flows two by two is equal to zero if and only if all flows are equal to zero or only one flow is greater than zero, making N as the destination of at most one transfer operation, as required. The main disadvantage of this formulation is that it defines a nonconvex nonlinear program. In the next section, “idle time” and “berthing” constraints are formulated in a similar fashion.

Port and refinery crude area systems can be modeled with basically the same equations. Therefore, a single NLP model can be used for both systems. It features material flows as control variables (Eq. 2), and volumes and qualities as state variables (Eq. 3), calculated by means of state equations (Eq. 4-6). Moreover, bilinear equations model scheduling decisions: unique definition of source and destination in a transfer operation (Eq. 7), idle time to segregate impurities in portside or refinery tanks (Eq. 8), berthing time / setup time (Eq. 9), and constant flow constraints for pipelines or crude distillation units (Eq. 10). Both control and state variables are bounded by upper and lower limits. The objective function (Eq. 1) is a summation of different costs, that can be prioritized with the use of weights. The following costs are considered: demurrage (Eq. 11), unattained demand (Eq. 12), and inventory (Eq. 13).

$$\min J = \sum_{\text{cost}} w_{\text{cost}} C_{\text{cost}} \quad (1)$$

$$0 \leq u(t_i) \leq u^{\max}(t_i) \quad (2)$$

$$x^{\min} \leq x(t_i) = [v(t_i) \quad p(t_i)]^T \leq x^{\max} \quad (3)$$

$$v(t_i) = v(t_{i-1}) + (t_i - t_{i-1}) \cdot U \cdot u(t_{i-1}) \quad (4)$$

$$p(t_i) = F(x(t_{i-1}), u(t_{i-1}), t_i - t_{i-1}) \quad (5)$$

$$p_n^{\text{density}}(t_i) = (u_{\text{inlet}, n}(t_{i-1}) p_{\text{inlet}, n}^{\text{density}}(t_{i-1})(t_i - t_{i-1}) + v_n(t_{i-1}) p_n^{\text{density}}(t_{i-1})) / (u_{\text{inlet}, n}(t_{i-1})(t_i - t_{i-1}) + v_n(t_{i-1})) \quad (6)$$

$$0 = r_n(t_i) = \sum_{k \in A_n} \sum_{j > k \in A_n} u_k(t_{i-1}) \cdot u_j(t_{i-1}) \quad (7)$$

$$0 = z_n(t_i) = \sum_{t'=t_i-T_{\text{idle}}}^{t_i-1} \sum_{k \in A_{\text{out}, n}} \sum_{j \in A_{\text{in}, n}} u_k(t_{i-1}) \cdot u_j(t') \quad (8)$$

$$0 = s_n(t_i) = \sum_{t'=t_i-T_{\text{berth}}}^{t_i-1} \sum_{j \in A_{n,p}} \sum_{k \in A_{n,!p}} u_k(t_{i-1}) \cdot u_j(t') \quad (9)$$

$$0 = q_n(t_i) = u_{0,n} - \sum_{j \in A_n} u_j(t_{i-1}) \quad (10)$$

$$C_{\text{demurrage}} = \sum_{j \in \text{Ships}} \sum_{t_i} c_j^{\text{demur}} \text{cargo_gap}_j(t_i) \quad (11)$$

$$C_{\text{demand}} = \sum_{j \in \text{Pipelines}} \sum_{p \text{ in Products}} c_j^{\text{demand}} (\text{demand}_{j,p} - \sum_{t_i} v_{j,p}(t_i)) \quad (12)$$

$$C_{\text{inventory}} = \sum_{j \in \text{Storages}} \sum_{t_i} c_j^{\text{inventory}} v_j(t_i) \quad (13)$$

Equation 4 features U as an incidence square matrix with entries belonging to the $\{0, 1, -1\}$ set. Eq. 5 is a general blending functional, while Eq. 6 is a particular blending function for density. Eq. 7 enforces that only one flow can be used by equipment n at time t_i , therefore, a transfer operation has only one source and only one destination at time t_i . Eq. 8 enforces the idle time, while Eq. 9 enforces the necessary berthing time for ships. Eq. 10 force a constant flow $u_{0,n}$ for a given equipment n – this constraint can be easily changed to force a variable flow, if needed. Eq. 11 deals with demurrage cost: we do not employ the classic demurrage formulation, but one that is also proportional to the remaining volume to be transferred that is delayed (*cargo_gap*). In all cost equations c_j^{cost} is a different arbitrary unitary cost.

An MILP model is developed, very similar to the NLP model, with the replacement of nonlinear equations 2, 7, 8, and 9 by mixed-integer equations 2a, 7a, 8a, and 9a. This model is based on the literature [7][8]:

$$0 \leq u(t_i) \leq B(t_i) \cdot u^{\max}(t_i), \quad b_j(t_i) \in \{0,1\}, B(t_i) = \text{Diag}(b_j(t_i)) \quad (2a)$$

$$r_n(t_i) = \sum_{j \in A_n} b_j(t_{i-1}) \leq 1 \quad (7a)$$

$$z_n(t_i) = \sum_{j \in A_{out,n}} b_j(t_{i-1}) + \sum_{t'=t_i-1}^{t_i-1} \sum_{j \in A_{in,n}} b_j(t') \leq 1 \quad (8a)$$

$$s_n(t_i) = \sum_{j \in A_{n,p}} b_j(t_{i-1}) + \sum_{t'=t_i-1}^{t_i-1} \sum_{j \in A_{n,p}} b_j(t') \leq 1 \quad (9a)$$

Equation 4a features a diagonal matrix $B(t_i)$, whose main diagonal is made up by the binary variables b_j – added to the model in the MILP formulation. Equations 7a, 8a, and 9a model represent the following constraints: equipment n can belong to only one transfer operation at time t_i , idle time, and berthing time/ setup time constraints.

5. Numerical Experiments

Trivial points are points where the control-vector u is zero for all time intervals. These points are very easy to be constructed, but they are not feasible in the original problem formulation. At trivial points, the demurrage and demand costs are maximal. The norms of the additional states are $\|z\| = \|r\| = \|s\| = 0$ and $\|q\| \gg 0$. Moreover, equations (7) to (10) are very restrictive, insofar that most NLP methods when initialized in a trivial point will stuck around this initial point. Our strategy is to relax Eq. (7) to (10) and add them to the objective function, obtaining the merit function J' , where e is the unary-vector:

$$J' = J + M \sum_t e^T r(t) + e^T s(t) + e^T z(t) + e^T q(t) \quad (14)$$

Five test instances were coded in AMPL [10]. The MILP and NLP instances consider the same costs and did not consider blending equations, in order to allow a comparison between their solutions (Table 2). The nonlinear models are relaxed with the linear penalty (Eq. 14). Standard commercial solvers were employed in this evaluation: CPLEX (v. 10.1.0) [11], SNOPT (v. 6.1) [12], and MINOS (v. 5.5) [13]. The number of variables shown in Table 3 is the one given by AMPL after its pre-solve procedure. All cases were solved with similar CPU times around 1 second, in a workstation with the following configuration: Intel Core Duo T2250 1.73GHz, RAM 1 GB, Linux OpenSUSE 10.1.

Table 2 – Solutions of the test cases

Case	MILP (CPLEX)	NLP (SNOPT)	NLP (MINOS)
1(A)	J = 1460 (Global) 14 iterations	J = 1460 (Global) 51 iterations	J = 1460 (Global) 17 iterations
1 (B)	J = 1600 (Global) 13 iterations	J = 1600 (Global) 13 iterations	J = 1625 (Local) 5 iterations
2	J = 0.33 (Global) 63 iterations	J = 0.33 (Global) 12 iterations	J = 0.33 (Global) 191 iterations
3 (A)	J = 0 (Global) 324 iterations / 8 B&B nodes	J = 0 (Global) 812 iterations	J = 0 (Global) 411 iterations
3(B)	J = 0 (Global) 397 iterations / 25 B&B nodes	J = 18.27 (Local) 544 iterations	J = 0 (Global) 472 iterations

Table 3 – Dimensions of the test cases

Case	MILP			NLP			
	Binary var.	Continuous var.	Constraints	Linear var.	Nonlinear var.	Lin. constr.	Nonlin. constr.
1 (A)	12	25	31	12	19	19	6
1 (B)	12	25	36	12	19	24	6
2	34	82	111	34	77	64	23
3 (A)	93	158	265	93	76	92	11
3 (B)	93	158	275	57	78	86	11

Test 1: Two crude tanks and one pipeline connected to a refinery, whose crude demand has to be fulfilled. There are 2 configurations: (A) allows the pipeline to be idle in certain periods, (B) keeps the pipeline with a constant flow, during the entire schedule. The MINOS run converged to a local minimum in (B) configuration. The NLP objective function is (Eq. 14), while the MILP

is (Eq. 1). Please notice that the (B) configuration can also represent a refinery problem, with two crude tanks and one crude distillation unit.

Test 2: Two crude tanks, one jetty and two tankers, whose cargoes have to be unloaded. The NLP objective function is (Eq. 14), while the MILP is (Eq. 1). Please notice that this configuration can also represent a refinery problem, with two crude tanks and one pipeline with two planned crude parcels.

Test 3: Three crude tanks, one jetty, three tankers, whose cargoes have to be unloaded, and one pipeline connected to a refinery, whose crude demand has to be fulfilled. There were 2 configurations: (A) allows the pipeline to be idle in certain periods, (B) keeps the pipeline with a constant flow, during the entire schedule. The SNOPT run converged to a local minimum in (B) configuration, incurring in demurrage costs. The NLP objective function is (Eq. 14), while the MILP is (Eq. 1). Please notice that this configuration can also represent a refinery problem, with three crude tanks, one pipeline with three planned crude parcels and one crude distillation unit.

Table 4 shows the MILP iterations when the NLP solutions were employed as initial incumbents to the MILP problem. A solution defined by the NLP formulation is transformed to a MILP point by simply adding the binary variables and replacing the complementarity constraints by the mixed-integer ones. For each positive flow, the corresponding binary variable is set to 1 (one), while for each null flow, the corresponding binary variable is set to 0 (zero). The number of branch-and-bound iterations and visited nodes is significantly reduced, even for these preliminary test cases.

Table 4 – MILP starting at different initial points.

Case	Iterations		
	$(x,u)^0 = (x^0, 0)$	$(x,u)^0 = (x,u)^{\text{SNOPT}}$	$(x,u)^0 = (x,u)^{\text{MINOS}}$
1(A)	14	13	13
1 (B)	13	4	4
2	63	47	40
3 (A)	324 (8 nodes)	215	215
3(B)	397 (25 nodes)	265 (6 nodes)	215

7. Conclusions

In this paper we discussed a nonlinear approach to model the scheduling of continuous processes, based on flow rates as control variables. We compared this approach with a typical MILP one and discussed a hybrid scheme, which combines both. If one wants to solve the MILP until global optimality, it is possible to use the NLP as an auxiliary problem. Preliminary numerical results showed a significative reduction of MILP iteration with an NLP-generated initialization, encouraging us to continue this research, with larger instances in the near future.

8. References

1. P. M Pardalos, O. A. Prokopyev, S. Busygin. Continuous Approaches for Solving Discrete Optimization Problems. In Handbook on Modeling for Discrete Optimization, G. Appa, L. Pitsoulis, H. P. Williams (ed.), Springer, 2006
2. S. Leyffer. Complementarity Constraints as Nonlinear Equations: Theory and Numerical Experience. In S. Dempe and V. Kalashnikov (ed.), Optimization and Multivalued Mappings, Springer, 2006
3. C. A. Floudas and X. Lin. Mixed integer linear programming in process scheduling: modeling, algorithms, and applications. *Annals of Operations Research*, 2005, 139, 131–162.
4. C. A. Floudas and X. Lin. Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review. *Computers and Chemical Engineering*, 2004, 28, 2109-2129
5. R. Más, J.M. Pinto. A mixed-integer optimization strategy for oil supply in distribution complexes, *Optimization and Engineering*, 2003, 4 (1), 23-64.
6. P. C. P. Reddy, I.A. Karimi, R. Srinivasan. A new continuous-time formulation for scheduling crude oil operations, *Chemical Engineering Science*, 2004, 59, 1325-1341
7. M. V. Magalhães, N. Shah. Crude Oil Scheduling, *Foundations of Computer-Aided Process Operations Proceedings*, 2003, 323–326.
8. N. Shah. *Mathematical Programming Techniques for Crude Oil Scheduling*, *Computers and Chemical Engineering*, 1996, 20 suppl., S1227-S1232.
9. W. Collyer. Sobreestadia de navios: a regra 'once on demurrage, always on demurrage', *Jus Navigandi*, 2006, 1166. Web: <http://jus2.uol.com.br/doutrina/texto.asp?id=8889> [Last access on February 2008]
10. R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming*, Duxbury Press, 2003
11. ILOG, Inc. *ILOG AMPL/CPLEX System Version 8.0 User's Guide*, 2002
12. P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization, *SIAM Journal on Optimization*, 2002, 12 (4), 979–1006
13. B. A. Murtagh and M. A. Saunders. A projected Lagrangian algorithm and its implementation for sparse non-linear constraints, *Mathematical Programming Studies*, 1982, 16, 84–117.