Computers and Chemical Engineering 84 (2016) 1–11

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

MILP reformulations for the design of biotechnological multi-product batch plants using continuous equipment sizes and discrete host selection

G. Sandoval[a], D. Espinoza[b], N. Figueroa[c], J.A. Asenjo[a,*]

[a] Center of Biotechnology and Bioengineering, CeBiB, Departamento de Ingeniería Química y Biotecnología, Universidad de Chile, Santiago, Chile
[b] Departamento de Ingeniería Industrial, FCFM, Universidad de Chile, Santiago, Chile
[c] Instituto de Economía, Pontificia Universidad Católica de Chile, Santiago, Chile

ARTICLE INFO

Article history:
Received 20 August 2014
Received in revised form 24 July 2015
Accepted 1 August 2015
Available online 19 August 2015

Keywords:
Multi-product batch plant
MINLP
MILP
Production path

ABSTRACT

In this article we present a new approach, relying on mixed-integer linear programming (MILP) formulations, for the design of multi-product batch plants with continuous sizes for processing units and host selection. The main advantage of the proposed approach is its scalability, that allows us to solve, within *reasonable* precision requirements, realistic instances. Furthermore, we show that many other alternatives are either numerically unstable (for the problem sizes that we are interested in), unable to solve large instances, or much slower than the proposed method. We present extensive computational experiments, which show that we are able to solve almost all tested instances, and, in average, we are ten times faster than alternative approaches. As we use a high level implementation language (AMPL) we should get further time improvements if lower level implementations are used (C, C^{++}).

Reproducibility of our results can be tested using our models and data available on-line at BPLIB.[1]

1. Introduction

Conventional multi-product batch process literature using an optimization-based approach model the design and synthesis of such plants with Mixed-Integer Non-Linear Programming (MINLP) formulations (Floudas, 1995). The usual objective is to minimize the investment cost subject to the fulfillment of the production targets of a given set of products. Major drawbacks are given by the combinatorial nature of mixed-integer programming and possible nonconvexities due to non-linearities. In computational optimization numerical issues of these formulations given by rounding errors, numerical instabilities and approximation errors are well-documented (Goldberg, 1991; Koch, 2004; Margot, 2009; Vielma, 2013).

Since Robinson and Loonkar (1972) different procedures have been proposed to tackle these problems (Reklaitis, 1990; Rippin, 1993; Barbosa-Póvoa, 2007; Verderame et al., 2010; Nikolopoulou and Ierapetritou, 2012) but a method that is more efficient for a particular example is hardly predictable (Ponsich et al., 2007) and nowadays the development of effective solution approaches and algorithms remains very necessary (Grossmann and Guillén-Gosálbez, 2010).

The logarithmic change of variables proposed by Kocis and Grossmann (1988) linearizes most of the functions and leads to a convex MINLP problem, approach used by Ravemark and Rippin (1998) and Montagna et al. (2000) among others. Another approach chosen by Pinto et al. (2001) and Ponsich et al. (2007) among others is the use of specially designed solvers which can usually find good feasible solutions by the use of heuristic procedures (Grossmann et al., 2000). In practice the best *off the shelf* solvers for this kind of problems are the open source codes BONMIN and SCIP and the commercial solvers BARON and DICOPT that stand out in Mittelmann's benchmarks for optimization software (Mittelmann, 2013). Nevertheless none of them guarantee convergence to a global optimum, converging in some instances to local optima or not converging altogether. For the particular case of BARON and DICOPT performance failures are reported for non-convex models (Ponsich et al., 2007; Rebennack et al., 2011; Li et al., 2012); nevertheless even in cases where theoretically the algorithms work, we found that in practice, they do not converge to the global optimum. We have run precise experiments that demonstrate these failures in convex MINLP formulations (see Section 2).

It is a fact that there is a huge gap between Mixed-Integer Linear Programming (MILP or MIP) and MINLP solvers technology (Nowak,

* Corresponding author. Tel.: +56 229784723; fax: +56 226991084.
E-mail addresses: gsandova@ing.uchile.cl (G. Sandoval), daespino@dii.uchile.cl (D. Espinoza), nicolasf@uc.cl (N. Figueroa), juasenjo@ing.uchile.cl (J.A. Asenjo).

[1] Available at: http://www.dii.uchile.cl/ daespino/

http://dx.doi.org/10.1016/j.compchemeng.2015.08.001
0098-1354/

Nomenclature

Indices

h	host
i	product
j	stage
m	number of duplicated units
UP,LO	upper and lower bounds

Sets

$\mathcal{E}^1$	set of batch stages, $\subset \mathcal{E}$
$\mathcal{E}^2$	set of semi-continuous stages, $\subset \mathcal{E}$
$\mathcal{E}^3$	set of chromatographic stages, $\subset \mathcal{E}$
$\mathcal{E}$	set of all stages
$\mathcal{H}$	set of hosts
$\mathcal{I}$	set of products
$\mathcal{M}$	set of available units operating in parallel in-phase or out-of-phase
$\mathcal{R}$	set of routes: stages j needed to process the product i synthesized by host h
$\mathcal{U}$	set of available hosts h for product i synthesis

Parameters

δ	time horizon
c_j^n, γ_j^n	cost coefficients related to Y_j^n with $n \in \{1, 2, 3\}$
d_i	production target for product i
S_{ij}^n, S_{ij}^n	size factor for product i in stage j related to Y_j^n. $s_{ij}^n = \ln(S_{ij}^n) n \in \{1, 2, 3\}$
T_{ij}^0, T_{ij}^0	time factor for product i in the batch or chromatographic stage j. $t_{ij}^0 = \ln(T_{ij}^0)$
T_{ij}^1, t_{ij}^1	time factor for product i in the semi-continuous or chromatographic stage j. $t_{ij}^1 = \ln(T_{ij}^1)$

Variables

v	slack variable
X_j^n, X_j^n	number of units operating in parallel in-phase (X_j^1) and out-of-phase (X_j^2) in stage j. $x_j^n = \ln(X_j^n)$
Y_j^1, y_j^1	volumetric capacity for batch units and retentate or feed tank for semi-continuous or chromatographic stages. $y_j^1 = \ln(Y_j^1)$
Y_j^2, Y_j^2	volumetric capacity for permeate of product tanks for semi-continuous or chromatographic stages. $y_j^2 = \ln(Y_j^2)$
Y_j^3, Y_j^3	capacity of semi-continuous items. $y_j^3 = \ln(Y_j^3)$
Y_i^4, Y_i^4	batch size for final product i. $y_i^4 = \ln(Y_i^4)$
Y_i^5, Y_i^5	cycle time for product i. $y_i^5 = \ln(Y_i^5)$
z_{ih}^1	binary variable: 1 if protein i is synthesized by host h; 0 otherwise
z_j^2	binary variable: 1 if stage j is used to process at least one of the products
z_{jm}^3	binary variable: 1 if m units are operating in parallel in-phase in stage j; 0 otherwise
z_{jm}^4	binary variable: 1 if m units are operating in parallel out-of-phase in stage j; 0 otherwise

2005). Nowadays mixed-integer linear techniques are fast, robust and able to provide solutions to problems with up to millions of variables (Geißler et al., 2012). Taking advantage of this Voudouris and Grossmann (1992) used reformulation schemes to develop MILP models for the preliminary design of multi-product batch plants, introducing binary variables for the selection of discrete available equipment sizes. From this point, to the design decisions other were included as synthesis, production planning and scheduling (Voudouris and Grossmann, 1993); design and planning in a multiperiod scenario (Moreno and Montagna, 2007); design of multi-product batch plants considering duplication of units in series (Moreno et al., 2009) and the design and planning of multi-product batch plants using mixed-product campaigns (Corsano et al., 2009). Most recently these MILP formulations have been used to account for the design and scheduling of this type of plants (Fumero et al., 2011, 2012a,b) and for the design under uncertainty considering different types of decisions (Durand et al., 2012, 2014; Moreno and Montagna, 2012; Moreno-Benito et al., 2014).

A key feature in these design problems is the use of *Big-M* constraints to account for selection decisions despite being problematic (Bosch and Trick, 2005). Some authors that have included this type of constraints in their formulations are Gupta and Karimi (2003), Corsano et al. (2009), Moreno et al. (2009), Moreno and Montagna (2012). Obviously, these authors have found that the value of the *Big-M* parameters has a tremendous impact on the solution time; see for example Moreno et al. (2007). In addition it has been proven experimentally that other methods, as the convex hull formulation presented by Montagna et al. (2004) are better to account for selection decisions.

In this paper we develop a robust methodology to solve the design problem of a biotechnological multi-product batch plant in situations where equipment can be manufactured according to customer needs, as fermentors or tanks in general. To do that, we develop a MILP formulation which does not rely on the use of *Big-M* constraints and does not use a discrete range of equipment sizes. To do that we use four basic techniques (see Fig. 1): First, an extension of the non-linear (but convex) formulation proposed by Kocis and Grossmann (1988) is applied. Secondly, to deal with non-linear convex inequalities *a priori* we constructed linear outer (or inner) approximations of them which allow us to compute (*a posteriori*) true feasible solutions and lower (or upper) bounds.

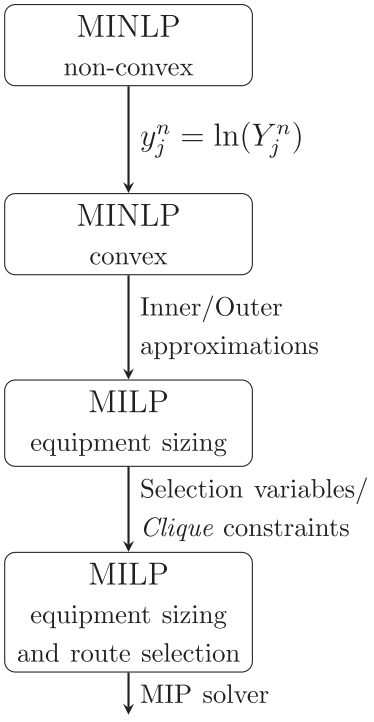

Fig. 1. Basic techniques used to model synthesis and design decisions considering continuous equipment sizes and discrete host selection.

Thirdly, to deal with integer variables, we used advanced reformulation techniques coming from the mixed-integer-programming literature (*clique* constraints). Finally, once the initial problem is transformed into a standard mixed-integer programming problem, it is possible to take advantage of mature commercial MIP solvers.

This approach, at least in our experiments, is more stable numerically, scalable, and faster to solve than current alternatives and can deal with the more general problem of jointly selecting equipment sizes and alternative production paths for multiple products. Using our approach, it is possible to quickly and accurately compute solutions at any desired precision level. In our extensive computational experiments (see Fig. 11) we found that current non-linear solvers only solved 43% of the instances generated for this study, while our approach was able to solve over 95% of the studied instances in a running time that, on average, was more than ten times faster than MINLP solvers in equivalent and standard MINLP formulations. To make these comparisons we introduce the performance profiles; a methodology borrowed from the optimization literature.

The rest of this paper is organized as follows. In Section 2 typical drawbacks found by a commonly used MINLP solver and the standard MINLP formulation is presented. In Section 3 classic and novel formulations for the design problem are described. Relevant information about the methodology used to benchmark different formulations and to avoid numerical instabilities is given in Section 4 and computational results are presented and discussed in Section 5. Finally, the conclusions are presented in Section 6.

2. Current limitations

Our main objective is finding a robust and scalable methodology for the design of biotechnological multi-product batch plant considering equipment sizing (design decisions) and selecting the downstream processing stages (synthesis decisions). Given the complexities that to date have been added to the original design problem we decided to go back to the problem studied by Iribarren et al. (2004) where only design and synthesis decisions are modeled. In their paper they designed a biotechnological batch plant for the production of four recombinant proteins, i, where each can be synthesized by two different hosts, h, having four microorganisms in total. In addition to that three of the fifteen processing stages, j, may be performed by two different unit operations, d. In their formulation they used constant size (S_{ijdh}) and time (T_{ijdh}) factors to model each stage; considered duplication of units in parallel in-phase, G_{jd}, and out-of-phase, M_{jd}, in order to diminish either the equipment sizes V_j or cycle times TL_i, respectively, and used *Big-M* constraints to account for the selection of hosts and equipment.

As a correctness test, we took the example presented in Iribarren et al. (2004) and splitted into 16 different instances that only allow equipment selection. Then we tested two different yet equivalent formulations based on their model but removing host selection[2].

In the first (C1) the selection of hosts was eliminated by limiting the set of available hosts, H_i, to just one per protein, and in the second (C2), by setting the values of the selection binary variables to 1 for the selected hosts and 0 for those non-selected. If the solution is being found by the solvers, we should observe two things:

(a) both model formulations (C1) and (C2) give the same solution, and
(b) the minimum of the separated instances is equivalent to the global minimum of the problem with host selection.

[2] To further isolate the results obtained from problems with non-standard local-settings, the GAMS modelling language was used and the experiments were run in the NEOS server (Gropp and Moré, 1997; Czyzyk et al., 1998; Dolan, 2001) available in http://www.neos-server.org.

Table 1
Comparison of the number of constraints and variables of some selected instances solved using models that account for selection with *Big-M* constraints – models (C1) and (C2) – and a classic formulation for design decisions only, (P1). All instances were solved using the DICOPT solver.

	Variables		Constraints		Status of solution
	Discrete	Continuous	Linear	Non-linear	
(C1)	480	185	303	10	Incorrect
(C2)	512	323	682	18	Incorrect
(P1)	360	71	179	9	Correct

Contrary to what we expected differences in the objective function value for both (C1) and (C2) formulations went from 1% to 78% in the 16 instances studied (data not shown), and even more striking, the solver finds a local minimum which is worse than those found for most instances without host selection.

These numerical instabilities seem to be aggravated with size since it is known that DICOPT works fine for small instances. Situation in accordance to the results obtained by Ponsich et al. (2007). In order to show these differences in sizes we built Table 1 to compare the number of constraints and variables involved in the smaller instances of the cases (C1) and (C2), that were incorrectly solved according to the aforementioned results, with the size of a smaller instance that was correctly solved by a classic formulation (P1) that solves an equipment sizing problem similar to that presented by Iribarren et al. (2004), but with no selection of hosts or equipments, and using DICOPT solver. This last formulation is presented en Section 3.1.1.

3. Problem formulation

Two major contributions are presented in this section. First, *clique* constraints are introduced to formulate the discrete part of the model allowing the selection of the production path without the use of *Big-M* constraints, in models (P2) and (P4). Second, a new approach, in Section 3.2, to handle non-linearities using standard reformulation techniques from the optimization field that permits the use of linear solvers leading to more reliable results and faster computing time. The relation among the four different models studied is shown in Fig. 2.

3.1. MINLP formulation

3.1.1. The equipment-sizing problem (P1)

In this section we present the most basic formulation for the design of biotechnological multi-product batch plants as only equipment sizing and duplication of units in parallel are considered.

The plant consists of a sequence of batch, semi-continuous and chromatographic stages used to manufacture different products i;

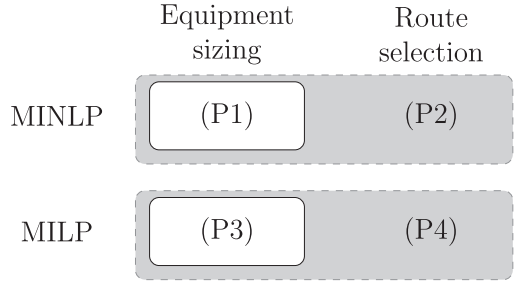

Fig. 2. Formulations compared in this article. Model (P1) is the most basic formulation that only includes design decisions. Model (P2) includes the selection of the downstream processes without the use of *Big-M* constraints. Models (P3) and (P4) are the transformed models of (P1) and (P2), respectively, using our proposed inner and outer approximations.

where semi-continuous as well as chromatographic stages are composed by the semi-continuous items plus feed and product tanks. At each stage j there are X_j^2 groups of units operating in parallel out-of-phase and each group is conformed by X_j^1 units operating in-phase. For semi-continuous or chromatographic stages feed and product tanks can only be duplicated out-of-phase. Single production campaigns are considered and batches are transferred from one stage to the next without delay (zero wait policy).

The objective is to minimize the investment costs of main equipments of the plant (see Eq. (1)) given fixed production targets, d_i, over a time horizon δ.

$$\min \text{cost} = \sum_{j\in\mathcal{E}^1} X_j^1 X_j^2 \left(c_j^1 {Y_j^1}^{\gamma_j^1}\right) + \sum_{j\in\mathcal{E}^2\cup\mathcal{E}^3} \left[X_j^2 \left(c_j^1 {Y_j^1}^{\gamma_j^1}\right) + X_j^2 \left(c_j^2 {Y_j^2}^{\gamma_j^2}\right) + X_j^1 X_j^2 \left(c_j^3 {Y_j^3}^{\gamma_j^3}\right)\right] + v\rho\delta \quad (1)$$

Variables $Y_j^{\cdot}$ represent the different equipment sizes. Parameters $c_j^{\cdot}$ and $\gamma_j^{\cdot}$ are cost coefficients distinctive for each kind of equipment and v is a slack variable included to assure feasibility (Montagna et al., 2004).

Making the change of variables introduced by Kocis and Grossmann (1988) we get the new objective function (2).

$$\min \text{cost} = \sum_{j\in\mathcal{E}^1} c_j^1 \exp\left(x_j^1 + x_j^2 + y_j^1\gamma_j^1\right) + \sum_{j\in\mathcal{E}^2\cup\mathcal{E}^3} \left[c_j^1 \exp\left(x_j^2 + y_j^1\gamma_j^1\right) + c_j^2 \exp\left(x_j^2 + y_j^2\gamma_j^2\right) + c_j^3 \exp\left(x_j^1 + x_j^2 + y_j^3\gamma_j^3\right)\right] + v\rho\delta \quad (2)$$

At each stage and for each product the size of the units must allow the processing of the incoming batch which can be splitted among X_j^1 units to not surpass the upper bound capacity of the equipment. In batch stages this constraint can be written as Eq. (3a); convexified in Eq. (3b).

$$Y_j^1 \geq \frac{S_{ij}^1 Y_i^4}{X_j^1} \quad \forall i \in \mathcal{I}, j \in \mathcal{E}^1 \quad (3a)$$

$$y_j^1 + x_j^1 \geq s_{ij}^1 + y_i^4 \quad \forall i \in \mathcal{I}, j \in \mathcal{E}^1 \quad (3b)$$

As in semi-continuous or chromatographic stages duplication is allowed just for semi-continuous items, feed and product tanks are sized using constraints (4) and (5).

$$y_j^1 \geq s_{ij}^1 + y_i^4 \quad \forall i \in \mathcal{I}, j \in \mathcal{E}^2 \cup \mathcal{E}^3 \quad (4)$$

$$y_j^2 \geq s_{ij}^2 + y_i^4 \quad \forall i \in \mathcal{I}, j \in \mathcal{E}^2 \cup \mathcal{E}^3 \quad (5)$$

Chromatographic columns have to process the incoming batch and both duplication in-phase and out-of-phase are allowed. Duplication in-phase is modeled in size constraint (6) since this permits smaller units and duplication out-of-phase is reflected in time constraints.

$$y_j^3 + x_j^1 \geq s_{ij}^3 + y_i^4 \quad \forall i \in \mathcal{I}, j \in \mathcal{E}^3 \quad (6)$$

The cycle time for each product i, Y_i^5, is defined as the time elapsed between the production of two consecutive batches and is given by the larger operating time, T_{ij}, among the stages in the process. This time can be decreased if a duplication of units out-of-phase is used:

$$Y_i^5 \geq \frac{T_{ij}}{X_j^2} \quad \forall i \in \mathcal{I}, j \in \mathcal{E} \quad (7)$$

As batch stages operate for a fixed time, T_{ij}^0, cycle time constraint in its convex form is given by Eq. (8):

$$y_i^5 + x_j^2 \geq t_{ij}^0 \quad \forall i \in \mathcal{I}, j \in \mathcal{E}^1 \quad (8)$$

Semi-continuous stages, on the other hand, operate during a time that depends on the final batch size, Y_i^4. For those stages the cycle time is constrained as in Eq. (9).

$$Y_i^5 \geq \frac{T_{ij}^1 \frac{Y_i^4}{X_j^1 Y_j^3}}{X_j^2} \quad \forall i \in \mathcal{I}, j \in \mathcal{E}^2 \quad (9a)$$

$$y_i^5 + x_j^2 \geq t_{ij}^1 + y_i^4 - x_j^1 - y_j^3 \quad \forall i \in \mathcal{I}, j \in \mathcal{E}^2 \quad (9b)$$

Lastly, chromatographic stages are modeled considering both fixed and variable operation times leading to the highly non-linear constraint (10).

$$Y_i^5 \geq \frac{T_{ij}^0 + T_{ij}^1 \frac{Y_i^4}{X_j^1 Y_j^3}}{X_j^2} \quad \forall i \in \mathcal{I}, j \in \mathcal{E}^3 \quad (10a)$$

$$y_i^5 + x_j^2 \geq \ln\left[\exp\left(t_{ij}^0\right) + \exp\left(t_{ij}^1 + y_i^4 - x_j^1 - y_j^3\right)\right] \quad \forall i \in \mathcal{I}, j \in \mathcal{E}^3 \quad (10b)$$

Production targets for all products, d_i, must be satisfied within the time horizon δ.

$$\sum_{i\in\mathcal{I}} \frac{d_i Y_i^5}{Y_i^4} \leq \delta + v\delta \quad (11a)$$

$$\sum_{i\in\mathcal{I}} \frac{d_i}{\delta} \exp\left(y_i^5 - y_i^4\right) \leq 1 + v \quad (11b)$$

Finally, variables for duplication in-phase X_j^1 are restricted to integer values using constraints (12) and (13), where z_{jm}^3 are binary variables and $\mathcal{M}$ a set of available units to operate in parallel in-phase. The same is valid for variables for duplication out-of-phase, X_j^2.

$$x_j^1 = \sum_{m\in\mathcal{M}} z_{jm}^3 \ln(m) \quad \forall j \in \mathcal{E} \quad (12)$$

$$\sum_{m\in\mathcal{M}} z_{jm}^3 = 1 \quad \forall j \in \mathcal{E} \quad (13)$$

Appropriate upper and lower bounds are also considered for all of the variables.

3.1.2. The design problem with selection of routes and equipment sizing (P2)

More recent models (like Iribarren et al. (2004)) take into account the joint selection of the production processes including selection of hosts and equipment. Their formulation uses classical *Big-M* constraints. Since we know that these constraints are problematic (Bosch and Trick, 2005) in this work we propose a different way to formulate the integer part of the problem replacing the *Big-M* by *clique* constraints. With this formulation all constraints are ignored except the upper bound on the variables (Dietrich et al., 1993). Model (P2) includes the sizing of the equipment, the duplication of units in parallel working in-phase and out-of-phase and accounts for the selection of the global process selecting routes which are defined as the series of unit operations used to purify

a protein given a certain host that synthesizes it. In this way once the pair product-host, (i, h), is selected the set of stages conforming the process is fixed.

The objective function becomes:

$$\begin{aligned}\min \text{ cost} = &\sum_{j\in\mathcal{E}^1} z_j^2 c_j^1 \exp\left(x_j^1 + x_j^2 + y_j^1\gamma_j^1\right)\\ &+ \sum_{j\in\mathcal{E}^2\cup\mathcal{E}^3} z_j^2\left[c_j^1\exp\left(x_j^2 + y_j^1\gamma_j^1\right) + c_j^2\exp\left(x_j^2 + y_j^2\gamma_j^2\right)\right.\\ &\left.+c_j^3\exp\left(x_j^1 + x_j^2 + y_j^3\gamma_j^3\right)\right] + \nu\rho\delta \end{aligned} \tag{14}$$

Since some stages can be unused and just one route per protein can be selected we introduced two binary variables: z_{ih}^1 and z_j^2. z_{ih}^1 is equal to 1 when for product i synthesis host h is selected and 0 otherwise and z_j^2 is 1 when stage j is used to process at least one of the products and 0 otherwise. Constraint (15) enforces to chose just one host h to produce the protein i and constraint (16) permits stage j to be used just in case at least one product needs it to be processed.

$$\sum_{(i,h)\in\mathcal{U}} z_{ih}^1 = 1 \tag{15}$$

$$z_j^2 \geq z_{ih}^1 \quad \forall(i,h,j)\in\mathcal{R}|(i,h)\in\mathcal{U} \tag{16}$$

For chromatographic stages constraints take the form of Eqs. (17)–(20) that are trivially satisfied if host h is not selected to produced protein i ($z_{ih}^1 = 0$). When the host h is selected for protein i ($z_{ih}^1 = 1$) and the stage j has to be performed to process product i then $z_j^2 = 1$ and constraints are the same as in previous formulation (Section 3.1.1).

$$y_j^1 z_j^2 \geq s_{ihj}^1 z_{ih}^1 + y_{ih}^4 z_{ih}^1 \qquad \forall(i,h,j)\in\mathcal{R}, j\in\mathcal{E}^3 \tag{17}$$

$$y_j^2 z_j^2 \geq s_{ihj}^2 z_{ih}^1 + y_{ih}^4 z_{ih}^1 \qquad \forall(i,h,j)\in\mathcal{R}, j\in\mathcal{E}^3 \tag{18}$$

$$y_j^3 z_j^2 + x_j^1 z_j^2 \geq s_{ihj}^3 z_{ih}^1 + y_{ih}^4 z_{ih}^1 \qquad \forall(i,h,j)\in\mathcal{R}, j\in\mathcal{E}^3 \tag{19}$$

$$y_{ih}^5 z_{ih}^1 + x_j^2 z_j^2 \geq \ln\left[\exp\left(t_{ihj}^0\right) + \exp\left(t_{ihj}^1 + y_{ih}^4 - x_j^1 - y_j^3\right)\right] z_{ih}^1 \quad \forall(i,h,j)\in\mathcal{R}, j\in\mathcal{E}^3 \tag{20}$$

If stage j is not necessary for the process ($z_j^2 = 0$); then equipment sizes are set to 0 with constraints as (21) and no unit is considered to conform that stage (constraint (22)):

$$y_j^{1,LO} z_j^2 \leq y_j^1 \leq y_j^{1,UP} z_j^2 \quad \forall j\in\mathcal{E} \tag{21}$$

$$\sum_{m\in\mathcal{M}} z_{jm}^3 = z_j^2 \quad \forall j\in\mathcal{E} \tag{22}$$

Finally, in the planning horizon constraint (23) only the terms associated to the selected host per protein are considered.

$$\sum_{(i,h)\in\mathcal{U}} \frac{d_{ih}}{\delta} z_{ih}^1 \exp\left(y_{ih}^5 - y_{ih}^4\right) \leq 1 + \nu \tag{23}$$

3.2. Mixed-integer linear formulations

To obtain more accurate solutions and, specially in larger instances, in a reasonable running time we present a MILP reformulation, which can be solved using any commercial MILP solver. These models are basically equal to their MINLP counterpart but replacing the non-linear objective and time constraints with sets of linear functions which give arbitrarily good lower or upper approximations of their respective original functions. The actual optimal solution is in between both approximations and the precision level is given by the number of cutting points selected to generate the set of linear functions to replace each non-linear function and the actual selection of the approximation points used for example equispaced or non-equispaced. In this way the accuracy of the solution can be as high as desired at the cost of longer computing time.

3.2.1. Inner and outer approximations

Given a convex function of one variable $g(x)\leq 0$ and a set of points $\{x_k\}_{k=1,\ldots,n}$ in the domain of g then, is easy to see that:

$$\left\{x|g(\hat{x}_k) + \nabla g(\hat{x}_k)(x-\hat{x}_k)\leq 0 \quad k=1,\ldots,n\right\} \supseteq \left\{x|g(x)\leq 0\right\} \tag{24}$$

and

$$\left\{x|g(\hat{x}_k) + \frac{g\left(\hat{x}_{k+1}\right) - g\left(\hat{x}_k\right)}{\hat{x}_{k+1} - \hat{x}_k}\left(x-\hat{x}_k\right)\leq 0 \quad k=1, ldots, n\right\} \subseteq \{x|g(x)\leq 0\}, \tag{25}$$

which allows for straightforward lower and upper approximations of g. Using this fact, it is easy to find inner and outer approximations of the problems (P1) and (P2).

In fact, for each non-linear constraints of the form $g_j(x)\leq 0$, and considering an arbitrary set of cutting points in the domain $\{x_k\}_{k=1,\ldots,n}$ the consideration of the set of constraints

$$g_j(\hat{x}_k) + \nabla g_j(\hat{x}_k)\left(x-\hat{x}_k\right)\leq 0 \quad k=1,\ldots,n \tag{26}$$

which leads to a larger feasible set, as can be seen in Fig. 3a. On the other hand, we consider the set of constraints

$$g_j(\hat{x}_k) + \frac{g_j\left(\hat{x}_{k+1}\right) - g_j\left(\hat{x}_k\right)}{\hat{x}_{k+1} - \hat{x}_k}\left(x-\hat{x}_k\right)\leq 0 \quad k=1,\ldots,n-1 \tag{27}$$

which leads to a smaller feasible set, as can be seen in Fig. 3b.

In the same way, the minimization of the cost objective function, $f(x)$, can be replaced by

$$\begin{aligned}&\min \quad \nu\\ &s.t. \qquad \nu \geq f(\hat{x}_k) + \nabla f(\hat{x}_k)(x-\hat{x}_k) \quad k=1,\ldots,n\end{aligned} \tag{28}$$

which leads, together with an outer approximation of constraints, to a lower bound of the true cost. The objective function can also be replaced by

$$\begin{aligned}&\min \quad \nu\\ &s.t. \qquad \nu \geq f(\hat{x}_k) + \frac{f\left(\hat{x}_{k+1}\right) - f\left(\hat{x}_k\right)}{\hat{x}_{k+1} - \hat{x}_k}(x-\hat{x}_k)\\ &\qquad\quad k=1,\ldots,n\end{aligned} \tag{29}$$

which leads, together with an inner approximation of constraints, to an upper bound of the true cost.

In what follows, $\nabla f(\hat{x}_k)$ or $\frac{f(\hat{x}_{k+1}) - f(\hat{x}_k)}{\hat{x}_{k+1} - \hat{x}_k} = \alpha_k$ in the lower or upper approximation, respectively, $\hat{x}_k = b_k$ and $f(\hat{x}_k) = \beta_k$.

3.2.2. Reformulation for the equipment-sizing problem (P3)

The assumptions for this model are the same as those for the MINLP proposed in Section 3.1.2.

Objective function. Cost functions of Eq. (2) are individually linearized using the approximations given in Section 3.2.1 which leads to Eq. (30):

$$\min \text{ cost} = \sum_{j\in\mathcal{E}^1} v_j^1 + \sum_{j\in\mathcal{E}^2\cup\mathcal{E}^3}\left[v_j^1 + v_j^2 + v_j^3\right] \tag{30}$$

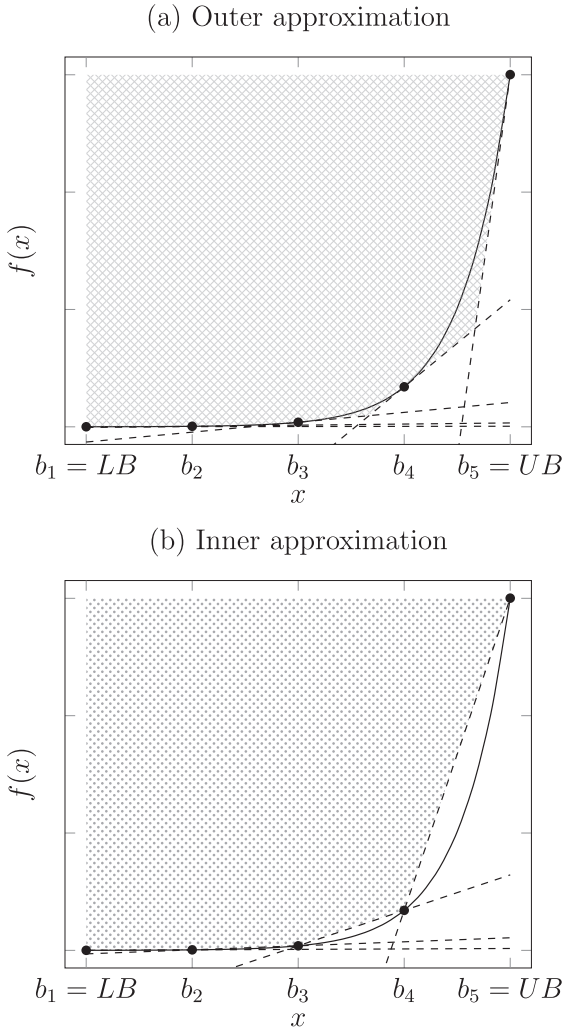

Fig. 3. Feasible region (patterned area) of (a) outer and (b) inner approximations (dashed lines) of an exponential function (solid line). Points b_i are the cutting points and LB and UB are the lower and upper bounds of x.

Constraints. Batch and semi-continuous stages and binary variables for duplication of units constraints in this MILP model are the same as those in the MINLP model shown in Section 3.1.1.

Chromatographic stages. Size constraints for feed and product tanks and column size constraint are the same as those in the MINLP model shown in Section 3.1.1.

Time constraint (31) is obtained from the linearization of Eq. (10):

$$y_i^5 + x_j^2 \geq \alpha_k^{6ij}\left(y_i^4 - x_j^1 - y_j^3 - b_k^{6ij}\right) + \beta_k^{6ij} \quad \forall i \in \mathcal{I}, j \in \mathcal{E}^3, k \in \mathcal{K}^6 \tag{31}$$

Planning horizon. From linearization of Eq. (11) constraint (32) is obtained:

$$\sum_{i \in \mathcal{I}} v_i^7 \leq 1 \tag{32}$$

Auxiliary variables. Cost functions in the objective function are linearized as shown in Eq. (33) and planning horizon constraint is linearized as shown in Eq. (34):

$$v_j^1 \geq \alpha_k^{1j}\left(x_j^1 + x_j^2 + \gamma_j^1 y_j^1 - b_k^{1j}\right) + \beta_k^{1j} \quad \forall j \in \mathcal{E}^1, k \in \mathcal{K}^1 \tag{33}$$

$$v_i^7 \geq \frac{d_i}{\delta}\alpha_k^{7i}\left(y_i^5 - y_i^4 - b_k^{7i}\right) + \frac{d_i}{\delta}\beta_k^{7i} \quad \forall i \in \mathcal{I}, k \in \mathcal{K}^7 \tag{34}$$

3.2.3. Reformulation for the design problem considering route selection and equipment sizing (P4)

Similar to the model presented in Section 3.2.2 this model was built based on its MINLP counterpart and most of constraints remain the same.

The objective function is the same as that from model (P3) and for stages design only differences are encountered for time constraints of chromatographic stages. In this way, applying the inner or outer approximations to Eq. (20) constraint (35) is obtained:

$$y_{ih}^5 + x_j^2 \geq \alpha_k^{6ihj}\left(y_{ih}^4 - x_j^1 - y_j^3\right) + \left(\beta_k^{6ihj} - \alpha_k^{6ihj} b_k^{6ihj}\right) z_{ih}^1$$
$$\forall (i,h,j) \in \mathcal{R}, j \in \mathcal{E}^3, k \in \mathcal{K}^6 \tag{35}$$

This set of equations together with constraints of the type of (36) for variables $y_j^1, Y_j^2, y_{ih}^4, y_{ih}^5, X_j^1$ and X_j^2 will model the same situation as in (P2).

$$y_j^{3,LO} z_j^2 \leq y_j^3 \leq y_j^{3,UP} z_j^2 \quad \forall j \in \mathcal{E}^3 \tag{36}$$

If $z_j^2 = 0$ constraint (35) is trivially satisfied. On the other hand if $z_{ih}^1 = 0$ constraint (35) becomes:

$$x_j^2 \geq \alpha_k^{6ihj}\left(-x_j^1 - y_j^3\right) \quad \forall (i,h,j) \in \mathcal{R}, j \in \mathcal{E}^3, k \in \mathcal{K}^6 \tag{37}$$

Since α_k^{6ihj} is a positive parameter constraint (37) will be always satisfied only if $|x_j^1| > |y_j^3|$ or if both variables are bigger than or equal to 0. To assure this data preprocessing is necessary. As X_j^1 is always bigger than 0 we normalized variables Y_j^3 by their lower bound. More details in Section 4.3.3.

For the case of planning horizon constraint little difference is found between (32) and (38). The last one takes into account host selection:

$$\sum_{(i,h) \in \mathcal{U}} v_{ih}^7 \leq 1 \tag{38}$$

Finally, constraints for auxiliary variables v_j^1, v_j^2, v_j^3 and v_{ih}^7 are different from those for problem (P3) to account for route selection:

$$v_j^1 \geq \alpha_k^{1j}\left(x_j^1 + x_j^2 + \gamma_j^1 y_j^1\right) + \left(\beta_k^{1j} - \alpha_k^{1j} b_k^{1j}\right) z_j^2 \quad \forall j \in \mathcal{E}^1, k \in \mathcal{K}^1 \tag{39}$$

$$v_j^1 \leq z_j^2 v_j^{1,UP} \quad \forall j \in \mathcal{E}^1 \tag{40}$$

$$v_{ih}^7 \geq \frac{d_i}{\delta}\alpha_k^{7ih}\left(y_{ih}^5 - y_{ih}^4\right) + \frac{d_i}{\delta}\left(\beta_k^{7ih} - \alpha_k^{7ih} b_k^{7ih}\right) z_{ih}^1 \quad \forall (i,h) \in \mathcal{U}, k \in \mathcal{K}^7 \tag{41}$$

$$v_{ih}^7 \leq z_{ih}^1 v_{ih}^{7,UP} \quad \forall (i,h) \in \mathcal{U} \tag{42}$$

If $z_j^2 = 0$ constraints (39)–(42) are trivially satisfied and if $z_{ih}^1 = 0$ then constraints (41) and (42) are trivially satisfied. Finally, if $z_{ih}^1 = 1$ and $z_j^2 = 1$ then constraints (39)–(42) are the same as those in the MILP formulation without route selection.

4. Methods

4.1. Solvers and modelling language

For MINLP problems the open source BONMIN 1.5 and SCIP 3.0.1 solvers were studied. In our computational tests SCIP uses SoPlex

1.7.1 as the LP solver and BONMIN (with its default algorithm, B-Hyb) uses Cbc 2.7.1 as the MIP solver and Ipopt 3.10.0 with MUMPS as linear solver. For the case of BONMIN we tested 3 over 5 available algorithms: B-Hyb the default algorithm, B-Ecp a specific parameter setting of B-Hyb that can be faster in some cases (Bonami and Lee, 2013) and B-OA using CPLEX as the MILP solver that according to Mittelmann (2013) can be faster for convex instances. In preliminary studies solvers as KNITRO and COUENNE were also tested to solve our MINLP formulations, but their performance in our simplest instances were poorer than that for the selected solvers.

For MILP problems the commercial CPLEX solver in its version 12.4.0.0 was used as it is one of the top performer from the literature (Mittelmann, 2013).

All models were coded using the AMPL modelling language.

4.2. Execution environment

Each instance was executed using a single thread on a Intel(R) Xeon(R) CPU E5620@2.40GHz with a running time limit of 48 hours, an optimality relative gap of 0.1% for models (P3) and (P4) and 2% for (P1) and (P2), and a maximum memory usage of 6Gb of RAM.

The difference in the prescribed optimality gap for MILP and MINLP solvers is given by the fact that while MINLP problems are solved to find the actual minimum cost function within a defined optimality gap, and therefore an *a priori* optimality gap, MILP models find true upper and lower bounds for the actual cost function leading to an *a posteriori* optimality gap that is computed afterwards. As will be shown in Section 5.2, this difference ensure that our results are comparable.

4.3. Methodology

4.3.1. Instances

To compare different approaches two set of instances, with randomly generated data between given *reasonable* upper and lower bounds, were built: "sizing instances", to compare simpler models (P1) and (P3), and "routing instances" to compare more complex models (P2) and (P4). We considered a variety of different number of proteins to be produced (4–6), number of stages to conform the process (11–65), number of routes to synthesize the product (20–65) and different cost coefficients values (1–110% of nominal values).

4.3.2. Benchmarking

In order to compare the model-solver pairs studied in this work we introduce a new tool for process engineers that was introduced in the optimization field by Dolan and Moré (2002) to compare different optimization software: the performance profile.

As stated by Dolan and Moré (2002) the performance profile for a solver is the "cumulative distribution for a performance metric", for example computing time. In this way things like how many instances a solver is able to solve given some stop criteria like those shown in Section 4.2, or how fast it solves different instances of the same type of problem can be seen graphically.

As an example of how to read these plots, in Fig. 4a it can be seen that when using 17 cutting points 40% of the instances were solved to an *a posteriori* optimality gap up to 2% while when using 33 cutting points leads to an optimality gap under 0.5% for the same amount of instances.

4.3.3. Data pre-processing

It is known that zero-one problems of large-scale are hard combinatorial optimization problems (Crowder et al., 1983; Koch, 2004; Applegate et al., 2007) reason why in order to obtain reliable solutions preprocessing data is necessary. The use of tight bounds and

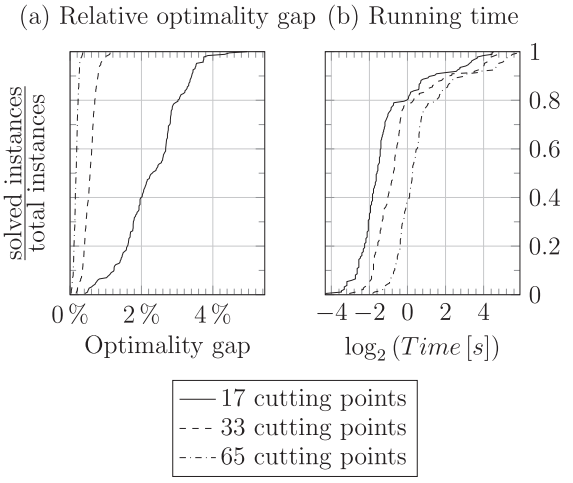

Fig. 4. Comparison of performance profiles of (a) relative optimality gap obtained *a posteriori* and (b) the logarithm of the running time of "sizing instances" solved with linear model (P3) using 17, 33 and 65 cutting points for lower and upper approximations with an optimality relative gap of 0.1%.

the normalization of the variables are necessary to decrease numerical errors.

Although not all of our instances are big enough to need data preprocessing all were subjected to the same treatment:

- All variable bounds and parameters associated to variables Y_j and Y_i were normalized by their respective lower bounds.
- Size and time factors were normalized and dimensionless considering the respective associated units. For example, as size factor for tanks have units of batch size divided by a volume this parameters are dimensionless by multiplying by the lower bound of the final batch size and dividing by the respective tank lower bound.
- Lower bounds for the cycle time were tightened using time constraints and upper bounds for final batch product were tightened using size constraints.

5. Results and discussion

In this section we show the robustness of our proposed MILP transformations and its superiority over classic MINLP formulations with *Big-M* constraints using performance profiles, a methodology borrowed from the optimization literature. Our approach is not only able to find correct solutions in realistic situations unlike MINLP formulations but also in a small fraction of the time required by those approaches. Major implications of these features are the exactness of the solutions that make this information reliable for decision-making; and as time reduction is significant numerous alternatives can be tested with the same formulation or with complexified models that may address the combination of different types of decisions.

This presentation is organized as follows: first, we describe the instances generated for comparison then discuss the selection of the cutting points for the proposed approach and finally, we compare MINLP and MILP formulations in terms of their performance solving the sets of instances using time as the metric.

5.1. Size of instances

To compare the most basic and easy to solve problems (P1) and (P3) a set of 186 instances ("sizing instances") were generated varying the number of proteins to be produced (2–6), the number of stages that conform each process (11–35) and the cost coefficient values (1–110% of nominal values). Sizes of these instances in terms

Table 2
Sizes of sample instances solved using non-linear model (P1).

	Variables		Constraints	
	Discrete	Continuous	Linear	Non-linear
Small	264	57	139	9
Medium1	840	157	407	5

Table 3
Sizes of sample instances solved using linear model (P3) with 33 cutting points for linear inner approximation.

	Variables		Constraints	
	Discrete	Continuous	Linear	Non-linear
Small	264	87	1357	–
Medium1	840	237	3096	–

of number of variables and constraints are shown in Tables 2 and 3, where "Small" corresponds to an example of one of the smaller instances solved with different models and "Medium1", to an example of the bigger instances solved for these two models. As it can be seen in both tables new auxiliary variables and the sets of linear functions generated to replace non-linear restrictions makes the problem from 7 to 12 times bigger in terms of linear constraints when 33 cutting points are used for linearization with an increase in about 50% of continuous variables. However, as we will see later, this increase in variables and constraints leads to smaller execution times and more accurate results.

To test models (P2) and (P4), as they were posed to solve more complex scenarios, a set of 249 new and bigger instances ("routing instances") were generated varying the number of proteins to be produced (4–6), the number of stages conforming the global process (18–65) and the number of routes available to produce the proteins (20–40). Sizes of these instances in terms of number of variables and constraints are shown in Tables 4 and 5, where "Medium2" corresponds to an example of one of the smaller "routing instances" solved with models (P2) and (P4) and "Large", to an example of the bigger instances solved in this work. As it can be seen in both tables, in comparison with Tables 2 and 3, the number of discrete variables increases by 5% with the addition of selection variables, z, and the number of linear constraints increases by 15% for (P4) and is around double for (P2). As we will see later, this addition permits the resolution of more complex scenarios, while at the same time not affecting execution time or optimality gap in comparison with the more basic formulation.

Table 4
Sizes of sample instances solved using non-linear model (P2).

	Variables		Constraints	
	Discrete	Continuous	Linear	Non-linear
Small	279	57	251	9
Medium1	881	157	719	5
Medium2	488	152	1262	22
Large	1794	613	15766	462

Table 5
Sizes of sample instances solved using linear model (P4) with 33 cutting points for linear inner approximation.

	Variables		Constraints	
	Discrete	Continuous	Linear	Non-linear
Small	279	87	1539	–
Medium1	881	237	3602	–
Medium2	488	229	4829	–
Large	1794	926	46489	–

5.2. Selection of cutting points

Contrary to MINLP problems (P1) and (P2) that are solved to an *a priori* optimality gap, models (P3) and (P4) give true upper and lower bounds for the actual cost function of each instance and therefore an optimality gap that is computed *a posteriori*. Fig. 4a shows the performance profiles of the gaps obtained *a posteriori* for the "sizing instances" solved with 17, 33 and 65 cutting points that generate 16, 32 and 64 linear functions for the inner approximations, respectively. Here we can see that our linear model is able to solve all instances with a maximum gap of 5% in less than 16 seconds when using 17 cutting points, and a gap of less than 0.5% in less than 64 seconds when using 65 cutting points. The running time profiles can be seen in Fig. 4b. If 65 cutting points had been selected, the optimality gap for non-linear solvers would have been around 0.5% as that is the worst gap obtained *a posteriori* with CPLEX (Fig. 4a). Given this, in our final experiments we use a set of 33 cutting points, since this option gives the largest improvement in gap versus the increase in execution time, and a slightly bigger optimality gap criteria of 2% was selected for non-linear solvers.

Once the number of points is selected, the specific values of these points must be chosen. The most obvious choice is equispaced points which, for the (relevant) exponential function e^x generates small errors for low values of x and large errors for high values. Another alternative is to use the expression (43), where N is the total number of cutting points including $-\infty$ as x_1 and $\bar{x}$ as the upper bound of x. This is a good approximation in order to minimize the *maximum value* of the error (see Fig. 5).

$$x_k = 2\ln\left(\frac{k-1}{N-1}\right) + \bar{x} \qquad \forall k \in 2...N \tag{43}$$

We can see in Fig. 5 that the choice of equispaced points leads to better approximations for low values of x, but much worse for high values. For our numerical experiments, equispaced points work better: while execution time remains the same for both approaches, *a posteriori* gaps were slightly worse for non-equispaced points (Fig. 6).

This leaves open important questions about the optimal point selection to improve the precision of upper and lower approximations. Our preliminary simulations seem to indicate that giving substantial attention to smaller values of x could significantly improve the results, but this is left for further research.

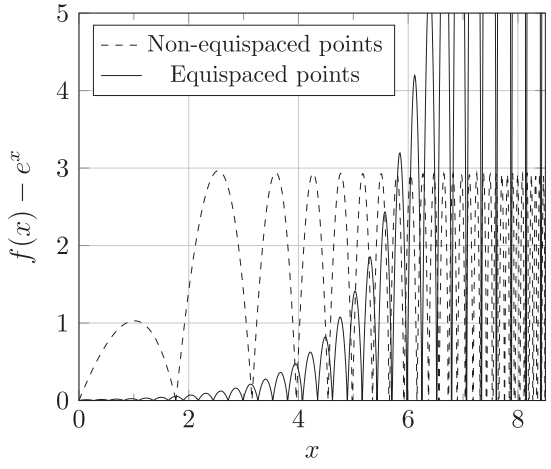

Fig. 5. Comparison of absolute errors using 2 different sets of 33 cutting points where $f(x)$ are the linear functions used to approximate the exponential function between 2 cutting points.

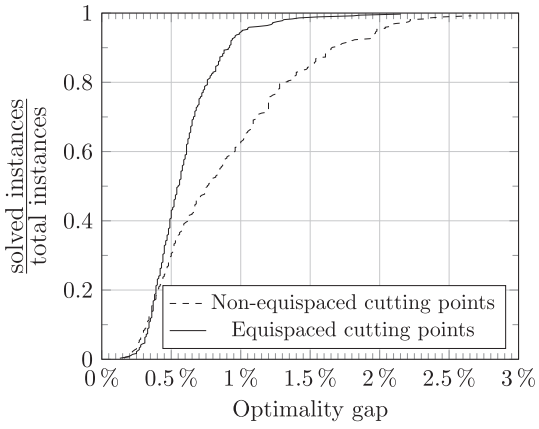

Fig. 6. Comparison of performance profiles of *a posteriori* gaps obtained using 33 cutting points to solve model (P4) where $f(x)$ are the linear functions used to approximate the exponential function between 2 cutting points. Time limit was set in 12 h.

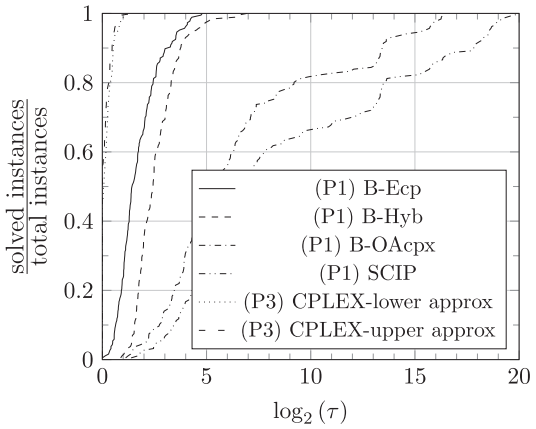

Fig. 8. Comparison of performance profiles of the logarithm of the ratio of the computing time of the pair model-solver versus the best time of the pairs model-solvers for "sizing instances" solved with models (P1) and (P3) with an optimality relative gap of 0.1% for the linear solver and 2% for non-linear solvers.

5.3. Equipment sizing: comparison of problems (P1) and (P3)

We tested three different combinations of solvers-models: the linear model (P3) was solved using CPLEX as solver while the non-linear model, (P1), was solved using SCIP and 3 of the 5 algorithms that are available for using BONMIN which were chosen based on BONMIN users' manual and Mittelmann's benchmarking (Mittelmann, 2013) information. All of the instances were solved using the stopping criteria and conditions mentioned in Section 4.2.

Fig. 7 shows the performance profile of running time using BONMIN-Hyb, BONMIN-Ecp, BONMIN-OAcpx, SCIP and our CPLEX-based approach. From this, we can see that problem (P1) was solved faster using any BONMIN algorithm than using SCIP solver. Moreover, SCIP only worked well in about the 75% of the instances, while BONMIN is able to solve the 85% of the instances using the OA algorithm and the 100% of the studied instances using either the Ecp or the Hyb algorithm. On the other hand, model (P3) solved using CPLEX is able to solve all of the instances studied, as well as BONMIN, but taking much less time than the problem (P1). As B-Ecp and B-Hyb seems to be equally good for those instances that take longer to be solved we decided to use as the performance metric the ratio of the computing time of the model-solver versus the best time of all of the model-solvers, denoted by τ. Those performance profiles are plotted in Fig. 8 where we can see even more clear than the CPLEX-based approach is always better than all of the other options and that Ecp algorithm is always better than Hyb for this asked optimality gap.

5.4. Routes selection: comparison of problems (P2) and (P4)

As a test of correctness, models (P2) and (P4) were solved using the generated "sizing instances" where just one route was available to produce each product. Contrasting these results to those obtained by (P1) and (P3) it can be seen in Fig. 9 that both formulations, for equipment sizing and considering routes, are consistent solving the same amount of instances in virtually the same amount of time. Performance profiles of the relative difference between the value of the objective function obtained with simpler – (P1) and (P2) – and more complex formulations – (P3) and (P4) – are presented in Fig. 10 where it can be seen that the differences between MILP formulations, as well as for MINLP formulations solved using BONMIN, are at most the optimality gap asked for each solver. The case of SCIP is different because in almost the 10% of the instances the solver gives results with a difference in the objective function between models (P1) and (P2) greater than the optimality gap which shows that this solver is not reliable to solve this type of problems.

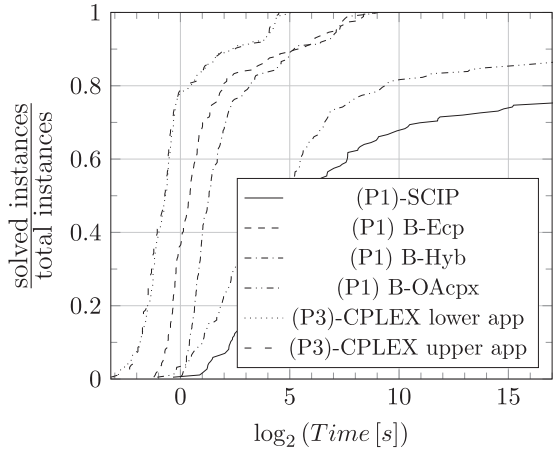

Fig. 7. Comparison of performance profiles of the logarithm of running time of "sizing instances" solved using models (P1) and (P3) with an optimality relative gap of 0.1% for the linear solver and 2% for non-linear solvers.

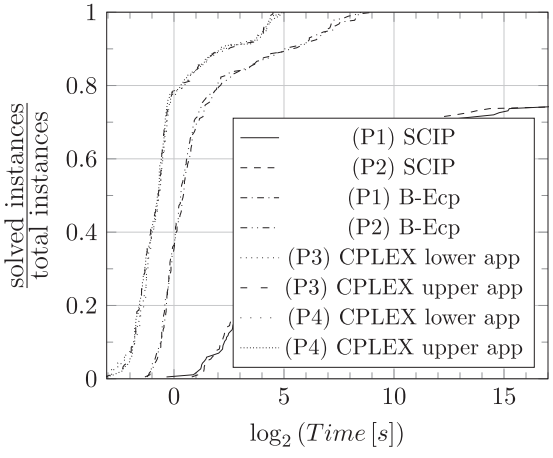

Fig. 9. Comparison of performance profiles of the logarithm of running time of "sizing instances" solved using models (P1), (P2), (P3) and (P4) with an optimality relative gap of 0.1% for the linear solver and 2% for non-linear solvers.

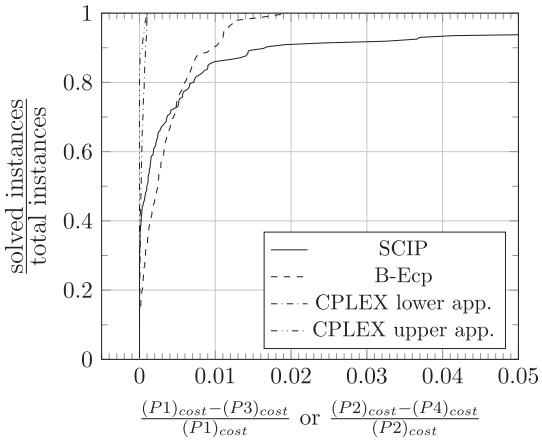

Fig. 10. Comparison of performance profiles of relative difference between simple and more complex formulation for "sizing instances". Models (P1) and (P2) were solved to an optimality gap of 2% and (P3) and (P4), to an optimality gap of 0.1%.

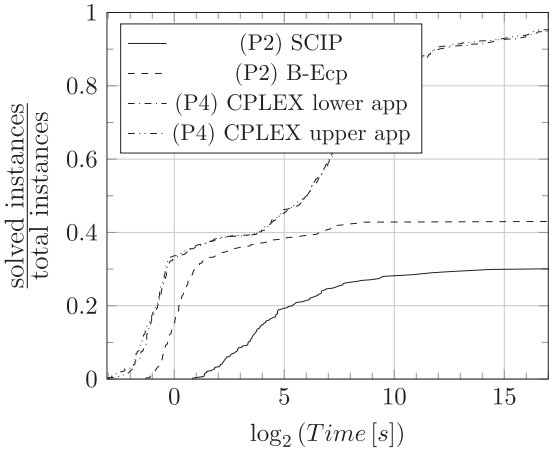

Fig. 11. Comparison of performance profiles of the logarithm of running time of "routing" and "sizing instances" solved using models (P2) and (P4) with an optimality relative gap of 0.1% for the linear solver and 2% for non-linear solvers.

As a final step in this work we compare in Fig. 11 the performance profiles of total running time obtained after solving all instances generated for this work (435 in total). Here we can see that for "routing instances" (P4) is much more robust than (P2), that was not able to solve any of those cases. In average (geometric average), the instances take about 40 s to be solved using model (P4), which is about the 3% of the time required by (P2)-BONMIN and less than the 1% of the time required by (P2)-SCIP. Nevertheless, for some punctual instances that represent the 4.6% of the instances studied, the time and/or memory usage were not enough to get the desired optimality gap. From this it can be stated that for the solution of more realistic instances or even to solve real problems considering continuous equipment sizes, the formulation proposed in this work is much more reliable and faster than the usual and widely studied standard MINLP formulation.

6. Conclusions

In this work we present a scalable approach to solve, within reasonable running times and quality assurance requirements, the problem of designing a biotechnological multi-product batch plant that support continuous equipment sizes and discrete host and/or process selection, up to sizes of real instances and that can be applicable to any kind of multi-product batch plant.

The proposed method was proved to be more *numerically stable* than other alternative approaches for the same problem giving true optimal solutions, and in general, faster than other tested approaches. Our method takes advantage of two facts: the continuous relaxation of the feasible region is convex and bounded (which allow us to build, up front, inner or outer approximations of the feasible space, and thus report true upper/lower bounds for each instance); and the fact that mixed-integer linear solvers are much more stable numerically and scalable in size than MINLP algorithms. Also, this approach relies on "off the shelve" optimization and modelling software, which makes it more amiable to practitioners.

To assert our claims, we borrow algorithm comparison tools from the optimization community, which are an interesting form to test the quality of competing algorithms to tackle the same class of problems.

Although, in a real scenario, semi-continuous units such as centrifuges and microfilters, among others, are available only in discrete sizes, unlike tanks that can be built according to customer needs, we feel that the proposed approach is robust enough to consider such issues, however, this was left as a next step in our research. Additionally, to increase the precision of our results for real cases, it is possible to explore a two step approach where after using the proposed method to obtain upper and lower approximations for the objective function we can refine different upper and lower variable bounds making them tighter and perform a re-optimization.

Finally, if true speed is the goal; we know that low-level implementations of dynamic inner/outer approximation can provide further time reductions, however, we feel that this is beyond the scope of this work.

Acknowledgements

This work was supported by a CONICYT scholarship for doctoral studies, FONDECYT Grant 1110024, Núcleo Milenio Información y Coordinación en Redes P10-024-F, and CONICYT for funding of Basal Centre, CeBiB, FB0001.

References

Applegate DL, Cook W, Dash S, Espinoza DG. Exact solutions to linear programming problems. Oper Res Lett 2007];35:693–9.
Barbosa-Póvoa AP. A critical review on the design and retrofit of batch plants. Comput Chem Eng 2007];31:833–55.
Bonami P, Lee J. BONMIN users's manual; 2013].
Bosch R, Trick M. Integer programming. In: Burke EK, Kendall G, editors. Search methodologies: introductory tutorials in optimization and decision support techniques. Springer; 2005]. p. 67–92.
Corsano G, Aguirre PA, Montagna JM. Multiperiod design and planning of multiproduct batch plants with mixedproduct campaigns. AIChE J 2009];55:2356–69.
Crowder H, Johnson EL, Padberg M. Solving large-scale zero-one linear programming problems. Oper Res 1983];31:803–34.
Czyzyk J, Mesnier MP, More JJ. The NEOS server. IEEE Comput Sci Eng 1998];5:68–75.
Dietrich BL, Escudero LF, Chance F. Efficient reformulation for 0-1 programs-methods and computational results. Discret Appl Math 1993];42:147–75.
Dolan ED, Moré JJ. Benchmarking optimization software with performance profiles. Math Program 2002];91:201–13.
Dolan ED. NEOS Server 4.0 administrative guide, Technical Report, Technical Memorandum ANL/MCS-TM-250. Mathematics and Computer Science Division, Argonne National Laboratory; 2001].
Durand GA, Mele FD, Bandoni JA. Determination of storage tanks location for optimal short-term scheduling in multipurpose/multiproduct batch-continuous plants under uncertainties. Ann Oper Res 2012];199:225–47.
Durand GA, Moreno MS, Mele FD, Montagna JM, Bandoni A. Comparing the performances of two techniques for the optimization under parametric uncertainty of the simultaneous design and planning of a multiproduct batch plant. Iberoam J Ind Eng 2014];5:42–54.
Floudas CA. Nonlinear and mixed-integer optimization: fundamentals and applications. Oxford University Press; 1995].
Fumero Y, Corsano G, Montagna JM. Detailed design of multiproduct batch plants considering production scheduling. Ind Eng Chem Res 2011];50:6146–60.

Fumero Y, Montagna JM, Corsano G. Simultaneous design and scheduling of a semi-continuous/batch plant for ethanol and derivatives production. Comput Chem Eng 2012a];36:342–57.
Fumero Y, Corsano G, Montagna JM. Planning and scheduling of multistage multiproduct batch plants operating under production campaigns. Ann Oper Res 2012b];199:249–68.
Geißler B, Martin A, Morsi A, Schewe L. Using piecewise linear functions for solving MINLPs. In: Mixed Integer Nonlinear Programming. Springer; 2012]. p. 287–314.
Goldberg D. What every computer scientist should know about floating-point arithmetic. ACM Comput Surv (CSUR) 1991];23:5–48.
Gropp W, Moré J. Optimization environments and the NEOS server. Approx Theory Opt 1997]:167–82.
Grossmann IE, Guillén-Gosálbez G. Scope for the application of mathematical programming techniques in the synthesis and planning of sustainable processes. Comput Chem Eng 2010];34:1365–76.
Grossmann IE, Caballero JA, Yeomans H. Advances in mathematical programming for the synthesis of process systems. Latin Am Appl Res 2000];30:263–84.
Gupta S, Karimi IA. An improved MILP formulation for scheduling multiproduct, multistage batch plants. Ind Eng Chem Res 2003];42:2365–80.
Iribarren OA, Montagna JM, Vecchietti AR, Andrews B, Asenjo JA, Pinto JM. Optimal process synthesis for the production of multiple recombinant proteins. Biotechnol Prog 2004];20:1032–43.
Koch T. The final NETLIB-LP results. Oper Res Lett 2004];32:138–42.
Kocis GR, Grossmann IE. Global Optimization of nonconvex mixed-integer nonlinear programming (MINLP) problems in process synthesis. Ind Eng Chem Res 1988];27:1407–21.
Li X, Chen Y, Barton PI. Nonconvex generalized Benders decomposition with piecewise convex relaxations for global optimization of integrated process design and operation problems. Ind Eng Chem Res 2012];51:7287–99.
Margot F. Testing cut generators for mixed-integer linear programming. Math Program Comput 2009];1:69–95.
Mittelmann H. Decision Tree for Optimization Software; 2013].
Montagna JM, Vecchietti AR, Iribarren OA, Pinto JM, Asenjo JA. Optimal design of protein production plants with time and size factor process models. Biotechnol Prog 2000];16:228–37.
Montagna JM, Iribarren OA, Vecchietti AR. Synthesis of biotechnological processes using generalized disjunctive programming. Ind Eng Chem Res 2004];43:4220–32.
Moreno MS, Montagna JM. New alternatives in the design and planning of multiproduct batch plants in a multiperiod scenario. Ind Eng Chem Res 2007];46:5645–58.
Moreno MS, Montagna JM. Multiperiod production planning and design of batch plants under uncertainty. Comput Chem Eng 2012];40:181–90.
Moreno MS, Montagna JM, Iribarren OA. Multiperiod optimization for the design and planning of multiproduct batch plants. Comput Chem Eng 2007];31:1159–73.
Moreno MS, Iribarren OA, Montagna JM. Optimal design of multiproduct batch plants considering duplication of units in series. Chem Eng Res Des 2009];87:1497–508.
Moreno MS, Iribarren OA, Montagna JM. Design of multiproduct batch plants with units in series including process performance models. Ind Eng Chem Res 2009];48:2634–45.
Moreno-Benito M, Espuña A, Puigjaner L. Flexible batch process and plant design using mixed-logic dynamic optimization: single-product plants. Ind Eng Chem Res 2014];53:17182–99.
Nikolopoulou A, Ierapetritou MG. Optimal design of sustainable chemical processes and supply chains: a review. Comput Chem Eng 2012];44:94–103.
Nowak I. Overview of global optimization methods. In: Relaxation and Decomposition Methods for Mixed Integer Nonlinear Programming; 2005]. p. 121–8.
Pinto JM, Montagna JM, Vecchietti AR, Iribarren OA, Asenjo JA. Process performance models in the optimization of multiproduct protein production plants. Biotechnol Bioeng 2001];74:451–65.
Ponsich A, Azzaro-Pantel C, Domenech S, Pibouleau L. Mixed-integer nonlinear programming optimization strategies for batch plant design problems. Ind Eng Chem Res 2007];46:854–63.
Ravemark DE, Rippin DWT. Optimal design of a multi-product batch plant. Comput Chem Eng 1998];22:177–83.
Rebennack S, Kallrath J, Pardalos PM. Optimal storage design for a multi-product plant: a non-convex MINLP formulation. Comput Chem Eng 2011];35:255–71.
Reklaitis GV. Progress and issues in computer-aided batch process design. Found Comput Aided Process Des 1990]:241–75.
Rippin D. Batch process systems engineering: A retrospective and prospective review. Comput Chem Eng 1993];17:S1–13.
Robinson JD, Loonkar YR. Minimizing capital investment for multi-product batch-plants. Process Technol 1972];17:861.
Verderame PM, Elia JA, Li J, Floudas CA. Planning and scheduling under uncertainty: a review across multiple sectors. Ind Eng Chem Res 2010];49:3993–4017.
Vielma JP. Mixed integer linear programming formulation techniques; 2013].
Voudouris VT, Grossmann IE. Mixed-integer linear programming reformulations for batch process design with discrete equipment sizes. Ind Eng Chem Res 1992];31:1315–25.
Voudouris VT, Grossmann IE. Optimal synthesis of multiproduct batch plants with cyclic scheduling and inventory considerations. Ind Eng Chem Res 1993];32:1962–80.