I&EC research
Industrial & Engineering Chemistry Research
ARTICLE
pubs.acs.org/IECR

A Spatial Branch-and-Bound Framework for the Global Optimization of Kinetic Models of Metabolic Networks

C. Pozo,[†] G. Guillén-Gosálbez,[*,†] A. Sorribas,[‡] and L. Jiménez[†]

[†]Departament d'Enginyeria Química (EQ), Escola Tècnica Superior d'Enginyeria Química (ETSEQ), Universitat Rovira i Virgili (URV), Campus Sescelades, Avinguda Països Catalans, 26, 43007 Tarragona, Spain

[‡]Departament de Ciències Mèdiques Bàsiques, Institut de Recerca Biomèdica de Lleida (IRBLLEIDA), Universitat de Lleida, Montserrat Roig 2, 25008 Lleida, Spain

ABSTRACT: The identification of the enzymatic profile that achieves a maximal production rate of a given metabolite is an important problem in the biotechnological industry, especially if there is a limit on the number of enzymatic modulations allowed. The intrinsic nonlinear behavior of metabolic processes enforces the use of kinetic models, such as the generalized mass action (GMA) models, giving rise to nonconvex MINLP formulations with multiple local solutions. In this paper, we introduce a customized spatial branch-and-bound strategy devised to solve efficiently these particular problems to global optimality. A tight MILP-based relaxation of the original nonconvex MINLP is constructed by means of supporting hyperplanes and piecewise linear underestimators. The overall solution procedure is expedited through the use of bound tightening techniques and a special type of cutting plane. The capabilities of the proposed strategy are tested through its application to the maximization of the citric acid production in *Aspergillus niger*. We also provide a numerical comparison of our algorithm with the commercial package BARON and an outer approximation-based method earlier proposed by the authors.

1. INTRODUCTION

Cellular and molecular biology has experienced a dramatic paradigm switch driven by the introduction of new technological and computational tools. This change has led to a wide acceptance of networks and their emergent properties as a central subject for understanding the evolution of cell metabolism. The emergence of systems biology as a discipline based on high throughput experimental techniques, bioinformatics methods, and mathematical modeling is the result of these advances. One of the consequences of this activity is a renewed interest in biotechnological applications that ranges from industrial products based on modified organisms to the possibility of designing new organisms.[1−4]

Cellular metabolism is a complex system that involves a huge number of components interacting in a dynamic way through nonlinear processes. This makes biological systems much more challenging than human designed factories and industrial products. In most problems, appropriate simplifications are required to grasp part of this complexity and to obtain practical results both in understanding the evolution of emergent properties and in predicting systems responses to experimental manipulation.[5−7]

Advances in molecular biology techniques have made it possible to modulate the expression of genes in a given organism in order to obtain strains with enhanced phenotypes.[8,9] Being able to improve the yield through modified strains is a crucial aspect for successful biotechnological applications. However, the intrinsic complexity of metabolic networks makes an intuitive inference of the most promising genetic changes a highly difficult (if not impossible) task. Henceforth, systematic optimization tools are required for improving metabolic engineering so that biotechnological applications can be made useful and affordable.

Optimization is not at all a new concept in biology.[10−13] It is clear that mathematical programming approaches offer a promising framework for analyzing mathematical models of biological systems in a systematic way, shedding light on the strategies that must be followed in order to improve their properties.[9,12,14−16] In particular, one of the areas in which systematic tools based on mathematical programming hold good promise is the analysis and manipulation of metabolic networks through gene expression modification.[17−20] From the point of view of industrial applications, the use of optimization methods in systems biology applications has gained wider interest.[9] Besides their application in increasing the yield of specific products, these techniques have also been used to explain the current adaptive responses of organisms and to predict the properties of new designs.[9,21,22]

While existing optimization techniques may be of some help, the complexity of cellular metabolism requires the development of global optimization methods that could be applied to these kinds of nonlinear problems. With these techniques, one expects that actual biological processes could be further improved by identifying quantitative operation principles that would help in deciding which genes should be modified and which is the optimal profile for obtaining a given goal. The fact that biological experiments are expensive and time-consuming[9] coupled with the usefulness of computational techniques when modeling metabolic networks[23] contributes to increasing the attractiveness of developing appropriate optimization approaches to address these problems.

Special Issue: Puigjaner Issue

Received: June 28, 2010
Accepted: October 19, 2010
Revised: October 11, 2010
Published: December 01, 2010

Flux balance analysis attempts this prediction by the optimization of stoichiometric models.[24] This approach leads to mixed-integer linear problems (MILP) that can be effectively solved by standard branch-and-bound techniques. This has been the main key of their success in different applications.[25−30] Unfortunately, this technique fails to capture the regulatory relationships that commonly exist between processes in metabolic networks.[31] These limitations can be overcome by resorting to kinetic metabolic models that account for the relationship between the concentration of metabolites and the fluxes in the network. Specifically, nonlinear kinetic expressions are preferred, as linear estimations have been found to be only valid for a narrow range around the approximation point.[8]

Among the available formalisms, models based on the power-law formalism in the variant form known as generalized mass action (GMA from here on) exhibit some particular advantages that make their application rather convenient.[32−37] For instance, as will be explained in detail later in the paper, they can adequately capture the nonlinear behavior of the metabolic regulations while exhibiting some linear properties when expressed in the logarithmic space. Furthermore, they are able to describe any particular metabolic network[37] what grants the generality of the framework presented herein. On the other hand, this approach gives rise to nonconvex models and, hence, to multimodality (i.e., existence of multiple solutions).[9] It should be emphasized that guaranteeing global optimality is of paramount importance in this type of problem, as a local optimal solution may lead to a completely different physical interpretation and objective function value than that associated with the global optimum, thus hampering the entire biological analysis.[38]

Global optimization addresses the computation and characterization of global optima (i.e., minima and maxima) of nonconvex functions constrained in a specified domain.[39] It has been the object of intense research during the past 15 years, but it is expected to continue as a major challenge in nonlinear optimization in the upcoming years.[40]

Global optimization approaches can be classified into stochastic or deterministic ones. Stochastic methods are nondeterministic approaches (i.e., they cannot guarantee global optimality) that make use of meta-heuristics in order to guide the search for "good" solutions from a series of pseudorandom generated points. These methods are often based on physical and biological analogies. On the other hand, deterministic methods are rigorous and, thus, can guarantee global optimality within a desired optimality gap. These methods rely on the calculation of a series of valid upper and lower bounds for the global optimum of the problem that approach each other during the execution of the algorithm until the optimality gap is reduced below a predefined tolerance. Among the different methods that may be included in this group, the most commonly used are the outer-approximation (OA)[41] and the spatial branch-and-bound (B&B) methods.[42−46]

In OA, the original problem is decomposed into two different subproblems at two different hierarchical levels: a master lower bounding problem and a slave upper bounding problem. The former is a relaxation of the original problem (i.e., it overestimates the feasible region of the original problem) that provides lower bounds on its global optimum. The latter entails the solution of the original problem in a reduced search space. In each iteration, the solution of the master problem is used as a starting point to solve locally the slave problem in a reduced search space (i.e., bounds are provided to some variables according to the solution of the master problem). If the optimality gap is found to be within a given tolerance, the algorithm terminates. Otherwise, the relaxation of the master problem is improved (i.e., is tightened) at the expense of introducing more variables.

On the other hand, in the spatial branch-and-bound (sBB here on; do not confuse with the MINLP solver sBB that implements a nonlinear branch and bound) method, the original problem is allocated in the root node of an exploration tree. Lower and upper bounds for the problem are compared, and if the desired tolerance is not met, the problem is split into two smaller subproblems (descendants) by partitioning the feasible space of a continuous variable (branching variable). Then, the two new problems are solved, if required, by recursive partitioning. If a node is proved not to contain the global optimum, then the associated branch in the sBB tree can be pruned. At the end, the global optimal solution is to be found in one of the subproblems derived during the process. This method is based on the idea of "*divide and conquer*" as each of the subproblems is smaller, and thus easier to solve, than the original one.

Multiple methods have been devised so far as variations from the original sBB. These methods include branch-and-reduce,[47,48] αBB,[49−54] symbolic reformulation,[38,55,56] reduced-space branch-and-bound,[57] branch-and-contract,[58] and the branch-and-cut framework proposed by Barton.[59] Some interval arithmetic global optimization methods[60−62] are sBB-like methods.[63] It has been observed that the performance of global optimization methods is highly dependent on the type of nonlinearities.[8] Henceforth, by exploiting the special mathematical structure of the problem under investigation,[64,65] it is possible to devise tighter relaxations that lead to faster algorithms.[66]

The application of global optimization methods to the analysis of metabolic networks that are described though nonlinear models (e.g., GMA formalism) has been scarce. Polisetty et al.[67] were the first ones to address this problem. In their work, they present a B&B procedure to identify the enzymes to be modified for efficiency in yield and cost. Later, Pozo et al.[68] proposed an outer-approximation algorithm that improved the method by Polisetty in terms of quality of the solutions provided (i.e., significantly smaller optimality gaps) and CPU time. The authors also presented a rigorous theoretical analysis on the construction of tight piecewise approximations and supporting hyperplanes. This method was also used to study the evolution of the cellular metabolism.[21,22]

In this work, we present a novel sBB method for the global optimization of metabolic networks that are modeled via the GMA formalism. Our computational procedure exploits the specific structure of the GMA models in order to construct tight MILP-based relaxations of the original nonconvex formulation. These linear relaxations are tightened through the use of a special type of cutting planes that are derived from some equations of the model. The sBB method is further expedited by tailored-made branching rules and bound contraction procedures based on interval analysis. The capabilities of this customized sBB are tested through a case study that addresses the optimization of citric acid production by *Aspegillus niger*. The results produced by our algorithm are compared with those generated by an outer approximation-based method introduced by the authors in previous works and also with the commercial global optimization package BARON.[21,22,68]

The paper is organized as follows. The problem is presented in section 2, and its mathematical formulation is proposed in section 3. The customized sBB is described in detail in section

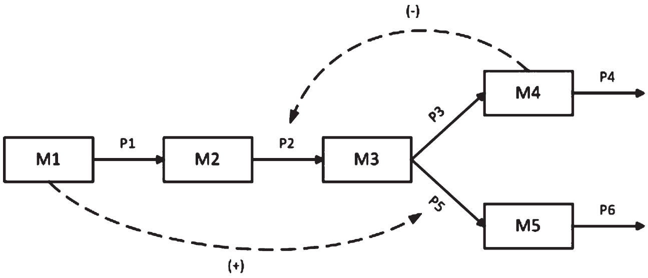

Figure 1. Example of a generic metabolic network, where processes are represented by arrows and metabolites by boxes.

4, whereas section 5 contains some numerical results. Finally, in section 6, we discuss some particular issues about the performance of the proposed methodology and its implementation.

2. PROBLEM STATEMENT

A metabolic network (Figure 1) is composed of a set of reactions and transportation processes (represented by arrows in the figure), generally ruled by enzymes, which transform organic substrates into metabolic intermediates and energy compounds (i.e., metabolites, in general). Some of these metabolites (represented by boxes in the figure) can also inhibit or facilitate some processes in the metabolic network. For instance, M4 inhibits P2 and M1 facilitates P5 in the figure.

The problem under study is the determination of the levels of the enzymes activities that maximize the synthesis rate of a particular metabolite in a metabolic pathway. The GMA representation is used to model the metabolic network behavior assuming steady state conditions. It is considered that all model parameters are deterministic in nature (i.e., perfectly known in advance without any variability). These parameters include the stoichiometric coefficients of the chemical reactions and the transportation processes, as well as the rate constants and kinetic orders of the power-law formalism describing these processes.

Under these conditions, we aim to customize a sBB global optimization method that may improve our previous results for this class of models. Due to the canonical representation provided by the GMA modeling strategy, this goal is of paramount importance for practical biotechnological applications.

3. MATHEMATICAL FORMULATION

The optimization problem is mathematically formulated as a MINLP, in which continuous variables denote metabolite concentrations and velocities, and binary variables model the changes in the enzyme levels. We first present the GMA formalism and then introduce the overall MINLP formulation.

3.1. GMA Representation. The concentration X of every single metabolite i present in the metabolic network can be determined at a particular time t from the p flows of the network:

$$\frac{dX_i}{dt} = \sum_{r=1}^{p} \mu_{ir} v_r \quad i = 1, ..., n \tag{1}$$

In eq 1, the stoichiometric coefficient, μ_{ir}, accounts for the number of molecules of metabolite i that are involved in process r. Hence, it is an integer parameter that is positive if process r contributes to the synthesis of metabolite i, negative if it depletes the concentration of i, and zero if process r does not directly influence the concentration of metabolite i. The velocity at which process r occurs, which is denoted by v_r, is described by a kinetic equation. In GMA models, the so-called power-law formalism[69−71] is the kinetic equation of choice (eq 2).

$$v_r = \gamma_r \prod_{j=1}^{n+m} X_j^{f_{rj}} \quad r = 1, ..., p \tag{2}$$

Here, γ_r is the basal state activity of the enzyme governing process r, whereas f_{rj} is the kinetic order of metabolite j in process r. This representation accounts for the m external (i.e., independent) metabolites, whose concentration is constant throughout the process (X_j = constant, $j = n + 1, ..., m$). By introducing eq 2 into eq 1 and removing the time dependence (we are interested in solving the steady state for which $dX_i/dt = 0$ applies), a complete GMA model as in eq 3 is obtained.

$$\sum_{r=1}^{p} (\mu_{ir} \gamma_r \prod_{j=1}^{n+m} X_j^{f_{rj}}) = 0 \quad i = 1, ..., n \tag{3}$$

3.2. MINLP Formulation. Since genetic manipulations will take place on an unmodified strain (i.e., at its basal state), it is convenient to express the optimal enzyme activities as a fold-change K_r over their basal state levels γ_r. According to this, we can rewrite eq 2 as follows:

$$v_r = K_r \gamma_r \prod_{j=1}^{n+m} X_j^{f_{rj}} \quad r = 1, ..., p \tag{4}$$

Here, K_r is a positive continuous variable that will take the value of 1 at the basal state (i.e., when the enzyme levels are not manipulated). Furthermore, $K_r > 1$ indicates overexpression of enzyme r, and $K_r < 1$ denotes its inhibition. This variable is allowed to change between given bounds, K_r^{LB} and K_r^{UB} as stated in eq 5.

$$K_r^{\mathrm{LB}} \le K_r \le K_r^{\mathrm{UB}} \quad r = 1, ..., p \tag{5}$$

The number of enzymes that can be modified at a time is constrained to be lower than an upper limit. The motivation for this is that a large number of genetic manipulations might be impractical. This is modeled through a disjunction that determines whether a specific enzyme is modified or not:

$$\begin{bmatrix} Y_{r1} \\ K_r^{\mathrm{LB}} \le K_r \le 1 - \delta \end{bmatrix} \vee \begin{bmatrix} Y_{r2} \\ 1 - \delta \le K_r \le 1 + \delta \end{bmatrix} \vee \begin{bmatrix} Y_{r3} \\ 1 + \delta \le K_r \le K_r^{\mathrm{UB}} \end{bmatrix} Y_{r1}, Y_{r2}, Y_{r3} \in \{\mathrm{True}, \mathrm{False}\} \quad r = 1, ..., p \tag{6}$$

Here, δ is a sufficiently small parameter (i.e., numerical results shown in this work were obtained using a value of 5×10^{-7}), and Y_r is a Boolean variable that is true if the associated term of the disjunction is satisfied and false otherwise. The disjunction in eq 6 can be reformulated into linear inequalities by applying either the Big-M or convex hull reformulations.[72,73] The latter, known to provide a relaxation at least as tight as the former,[72] gives rise to eqs 7−11.

$$K_r = K_{r1} + K_{r2} + K_{r3} \quad r = 1, ..., p \tag{7}$$

$$K_r^{\mathrm{LB}} y_{r1} \le K_{r1} \le (1 - \delta) y_{r1} \quad r = 1, ..., p \tag{8}$$

$$(1-\delta)y_{r2} \leq K_{r2} \leq (1+\delta)y_{r2} \quad r = 1, \dots, p \tag{9}$$

$$(1+\delta)y_{r3} \leq K_{r3} \leq K_r^{\mathrm{UB}} y_{r3} \quad r = 1, \dots, p \tag{10}$$

$$y_{r1} + y_{r2} + y_{r3} = 1 \quad r = 1, \dots, p \tag{11}$$

These equations enforce the definition of the binary variables y_{r1}, y_{r2}, and y_{r3}, which take the value of one if the corresponding term of the disjunction holds true and zero otherwise. These binary variables are then used to define an upper bound ME on the total number of enzymes that can be modified as follows:

$$\sum_{r=1}^{p} y_{r1} + \sum_{r=1}^{p} y_{r3} \leq \mathrm{ME} \tag{12}$$

Typically, metabolite concentrations will be allowed to change within given bounds (X^{LB} and X^{UB}, respectively):

$$X_i^{\mathrm{LB}} \leq X_i \leq X_i^{\mathrm{UB}} \quad i = 1, \dots, n \tag{13}$$

Generally, the objective of these problems is to maximize the synthesis rate of the desired product (note that any other objective function could be evaluated if required). For the sake of simplicity, we pose the problem as a minimization one by reversing the sign of the objective function:

$$\min - \sum_{r=1}^{p} \mu_{ir} v_r \tag{14}$$

Recall that only the velocities involved in the production of the desired metabolite must be considered in eq 14. The resulting MINLP that embeds the GMA equations can be expressed in compact form as follows:

$$(\mathrm{OMINLP}) \ \min -\sum_{r=1}^{p} \mu_{ir} v_r$$

$$\text{s.t. eqs 1, 4, and 7--13}$$

Model OMINLP [note that the authors have uploaded a similar model to ref 74] seeks the appropriate changes in the enzyme activities (continuous variables) that maximize the synthesis rate of the desired product. The enzyme activities calculated by the model can be implemented in the real system by tuning the expressions of the corresponding genes. Note that when the number of simultaneous modifications is not limited (recall that in our case it is), we can drop the binary variables, which gives rise to a nonconvex NLP problem.

Constraints in OMINLP define a nonconvex search space where multiple local optima may exist. Hence, in order to solve OMINLP to global optimality, we must resort to global optimization techniques.

4. SOLUTION STRATEGY

In this section, we present our customized sBB method for solving problem OMINLP to global optimality. This method makes use of a MILP-based linear relaxation of the nonlinear equations present in the MINLP formulation. We first describe in detail the way in which this relaxation is constructed before presenting the particularities of the sBB algorithm.

4.1. Relaxed Subproblem. In order to build a linear relaxation of OMINLP, we introduce two new auxiliary variables, k_r and x_i, that are defined by an exponential transformation as follows:

$$K_r = \exp k_r \quad r = 1, \dots, p \tag{15}$$

$$X_i = \exp x_i \quad i = 1, \dots, n \tag{16}$$

These variables replace the original ones, K_r and X_i, appearing in eq 4, thus giving rise to eq 17.

$$v_r = (\exp k_r)\gamma_r \prod_{j=1}^{n+m} (\exp x_j)^{f_{rj}} \quad r = 1, \dots, p \tag{17}$$

Let $p(i)$ denote the number of velocity terms explicitly expressed in the mass balance of metabolite i, that is, for which $\mu_{ir} \neq 0$. Velocities v_{r} appearing only in instances of eq 1 with $p(i) = 2$ are next transferred to linear constraints by introducing eq 17 into eq 1 and taking logarithms as follows:

$$0 = \mu_{ir} v_r + \mu_{ir'} v_{r'}$$

$$\mu_{ir}(\exp k_r)\gamma_r \prod_{j=1}^{n+m} (\exp x_j)^{f_{rj}} = -\mu_{ir'}(\exp k_{r'})\gamma_{r'} \prod_{j=1}^{n+m} (\exp x_j)^{f_{r'j}}$$

$$\ln(\mu_{ir}) + k_r + \ln(\gamma_r) + \sum_{j=1}^{n+m} f_{rj} x_j = \ln(-\mu_{ir'}) + k_{r'}$$

$$+\ln(\gamma_{r'}) + \sum_{j=1}^{n+m} f_{r'j} x_j \quad \forall\ i | p(i) = 2 \tag{18}$$

Recall that when the concentration of a metabolite is only determined by two processes, the stoichiometric coefficient of one of them must be negative, and hence, no domain violation for the logarithmic function can occur in eq 18.

On the other hand, when v_r appears in at least one instance of eq 1 with more than two terms (i.e., $p(i) \geq 3$), we make the following changes. We reformulate eq 4 by taking logarithms in both sides of the constraint:

$$\ln(v_r) = k_r + \ln(\gamma_r) + \sum_{j=1}^{n+m} f_{rj} x_j \quad \forall\ r \in r_{\mathrm{lin}} \tag{19}$$

In this equation, r_{lin} denotes the set of velocities r that are linearized by this process. In mathematical terms, $r \in r_{\mathrm{lin}} \subset \{r\} \Leftrightarrow \exists i | \mu_{ir} \neq 0 \wedge p(i) \geq 3$.

The right-hand side of eq 19 is now linear, but the logarithm in the left-hand side gives rise to a nonconvex search space. To linearize this nonconvex term, we reformulate the equation into two inequalities (eqs 20 and 21) and replace their left-hand sides with linear estimators.[75]

$$\ln(v_r) \geq k_r + \ln(\gamma_r) + \sum_{j=1}^{n+m} f_{rj} x_j \quad \forall\ r \in r_{\mathrm{lin}} \tag{20}$$

$$\ln(v_r) \leq k_r + \ln(\gamma_r) + \sum_{j=1}^{n+m} f_{rj} x_j \quad \forall\ r \in r_{\mathrm{lin}} \tag{21}$$

The left-hand side of equation eq 20 can be overestimated by L supporting hyperplanes, which are first-order Taylor expansions of the natural logarithm defined at L linearization points v_r^l within the domain $[v_r^{\mathrm{LB}}, v_r^{\mathrm{UB}}]$.

$$\ln v_r \leq \ln v_r^l + \frac{1}{v_r^l}(v_r - v_r^l) \quad \forall\ r \in r_{\mathrm{lin}} \quad l = 1, \dots, L \tag{22}$$

By combining eq 22 with eq 20, we obtain the following linear constraint (eq 23):

$$\ln v_r^l + \frac{1}{v_r^l}(v_r - v_r^l) \geq k_r + \ln(\gamma_r) + \sum_{j=1}^{n+m} f_{rj} x_j \quad \forall\ r \in r_{\text{lin}} \quad l = 1, ..., L \qquad (23)$$

Note that the quality of the relaxation depends on the number of linearizations added to the model.

On the other hand, the logarithmic term ln v_r in eq 21 is underestimated by means of a piecewise linear function[76−78] defined over H subintervals within the domain $[v_r^{\text{LB}}, v_r^{\text{UB}}]$ as follows:

$$\ln v_r \geq \begin{cases} a_r^1 v_r + b_r^1 & v_r^1 \leq v_r \leq v_r^2 \\ a_r^2 v_r + b_r^2 & v_r^2 \leq v_r \leq v_r^3 \\ \ldots & \\ a_r^h v_r + b_r^h & v_r^h \leq v_r \leq v_r^{h+1} \\ \ldots & \\ a_r^H v_r + b_r^H & v_r^H \leq v_r \leq v_r^{H+1} \end{cases} \qquad (24)$$

where a_r^h and b_r^h are the coefficients of the straight line that is active in the hth interval defined by the limits $v_r^1 = v_r^{\text{LB}}$ and $v_r^{H+1} = v_r^{\text{UB}}$. This can be modeled as a disjunction with h terms as follows:

$$\bigvee_{h=1}^{H} \begin{bmatrix} Z_r^h \\ v_r^h \leq v_r \leq v_r^{h+1} \\ a_r^h v_r + b_r^h \leq k_r + \ln(\gamma_r) + \sum_{j=1}^{n+m} f_{rj} x_j \end{bmatrix} \quad \forall r \in r_{\text{lin}}$$

$$Z_r^h \in \{\text{True}, \text{False}\} \qquad (25)$$

Here, the Boolean variable Z_r^h indicates whether the hth interval of the rth velocity is active or not. The last equation inside the disjunction is obtained by combining eq 21 and eq 24. The disjunction in eq 25 can be translated into linear equations through the convex hull reformulation.

$$v_r = \sum_{h=1}^{H} vh_r^h \quad \forall\ r \in r_{\text{lin}} \qquad (26)$$

$$v_r^h z_r^h \leq vh_r^h \leq v_r^{h+1} z_r^h \quad \forall\ r \in r_{\text{lin}} \quad h = 1, ..., H \qquad (27)$$

$$\sum_{h=1}^{H} z_r^h = 1 \quad \forall\ r \in r_{\text{lin}} \qquad (28)$$

$$\sum_{h=1}^{H} (a_r^h vh_r^h + b_r^h z_r^h) \leq k_r + \ln(\gamma_r) + \sum_{j=1}^{n+m} f_{rj} x_j \quad \forall\ r \in r_{\text{lin}} \qquad (29)$$

where vh_r^h is a disaggregated variable and z_r^h is a binary variable that takes the value of 1 if the hth interval of the rth velocity is active and 0 otherwise. Note that, in contrast with the supporting hyperplanes, the piecewise formulation does require the definition of binary variables. Hence, a proper balance should be found between the number of intervals and the quality of the relaxation, so that the computational burden of the model does not explode with the addition of a large number of binary variables.

Finally, eqs 7−10 are rewritten as follows:

$$k_r = k_{r1} + k_{r2} + k_{r3} \quad r = 1, ..., p \qquad (30)$$

$$\ln(K_r^{\text{LB}}) y_{r1} \leq k_{r1} \leq \ln(1 - \delta) y_{r1} \quad r = 1, ..., p \qquad (31)$$

$$\ln(1 - \delta) y_{r2} \leq k_{r2} \leq \ln(1 + \delta) y_{r2} \quad r = 1, ..., p \qquad (32)$$

$$\ln(1 + \delta) y_{r3} \leq k_{r3} \leq \ln(K_r^{\text{UB}}) y_{r3} \quad r = 1, ..., p \qquad (33)$$

Recall that bounds on variable X_i need to be expressed in the space of variables x_i as shown in eq 34.

$$\ln(X_i^{\text{LB}}) \leq x_i \leq \ln(X_i^{\text{UB}}) \quad i = 1, ..., n \qquad (34)$$

The lower bounding problem can be expressed in compact form as follows:

$$(\text{CMILP}) \ \min - \sum_{r=1}^{p} \mu_{ir} v_r$$

$$\text{s.t. eqs } 1,\ 11,\ 12,\ 18,\ 23,\ \text{and } 26{-}34$$

It should be clarified that the reformulation presented here is an opt-reformulation since all local and global optima of the original problem are mapped into local and global optima of the reformulated model.[63] Problem CMILP can be solved via standard methods for MILP problems such as the B&B.[46]

4.2. Customized Spatial Branch-and-Bound. The spatial branch-and-bound algorithm we propose to solve problem OMINLP exploits the particular features of the GMA model. The method is based on sequentially solving subproblems obtained by partitioning the original domain. A spatial branch-and-bound tree (sBB tree from here on) is used to represent the hierarchy of nodes.

Let OMINLP^k and CMILP^k denote the OMINLP and CMILP subproblems associated with node k of the sBB tree. The original problem, OMINLP, is allocated in the root node (k = 0). A convex relaxation of the original problem (model CMILP^0) is solved in order to obtain a valid lower bound on the global optimum of the original formulation.[42,44,79,80] An upper bound for the node can also be computed by optimizing locally the original model OMINLP^0 using the solution provided by CMILP^0 as starting point. If the optimality gap of the node is above the tolerance, then we generate subproblems OMINLP^1 and OMINLP^2 by splitting (branching) the domain of one of the p velocities v_r. This is equivalent to creating two descendant nodes in the sBB tree. Every time a subproblem OMINLP^k is created, it is added to a list T containing all of the active (i.e., yet to explore) nodes in the sBB tree. Each of these subproblems is then solved exactly in the same manner as OMINLP^0, in order to produce lower and upper bounds for each of the nodes. Recall that in these subproblems, we impose lower and upper limits on the variables according to the selected branching scheme. Every time a node is evaluated, the associated OMINLP^k problem is eliminated from T.

If at any node k of the sBB tree CMILP^k is infeasible, the node can be pruned, as it does not contain any feasible solution to OMINLP. If this happens at node 0, then OMINLP is infeasible. Similarly, if the optimal solution to CMILP^k, denoted by rOF^*, is above the overall upper bound OUB (i.e., the best bound considering all the nodes of the sBB tree), we can prune this node, as proceeding in this branch will only lead to worse solutions (note that as we go deeper in the tree, subproblems are more restricted). After updating OUB, we can prune those nodes in the active list with a lower bound greater than OUB.

Search trees are only finite for an ε-tolerance.[40] Hence, a node can be fathomed when the difference between the upper and lower bounds is smaller than the tolerance. We update OUB whenever the upper bound of the node is lower than the current OUB. The overall lower bound (OLB) corresponds to the lowest among the lower bounds of the active nodes in the sBB tree. The algorithm terminates when the gap between OUB and OLB is reduced below the ε-tolerance.

In the next few sections, we highlight some particular features of our sBB strategy.

4.2.1. Branching Strategy. An effective branching technique[50,81] aims at minimizing the size of the sBB tree and, thus, can strongly affect the performance of the algorithm.[63] In contrast with the application of B&B to MILP optimization, where the optimal solution of the relaxation is only infeasible in the original problem when integer variables take fractional values, in nonlinear optimization, infeasibilities may also be due to continuous variables violating constraints that have been relaxed. We must keep in mind that the termination criterion for the proposed strategy is achieving a sufficiently small optimality gap. A tight CMILP formulation capable of providing high-quality lower bounds plays a major role in the performance of the algorithm. Recall that eq 4 is the only equation of OMINLP that is relaxed to build CMILP. Hence, by deriving a tight approximation of the logarithmic function therein, it is possible to determine tight bounds on the global optimal solution of OMINLP. The proposed method branches on the velocities $v_r|r \in r_{\text{lin}}$. This is a common feature with the reduced space B&B[57] that only branches on a subset of variables. With this strategy, the linear estimators (i.e., piecewise linear functions and hyperplanes) concentrate on the lower region of the branching velocity in the left-hand descendant subproblem and in the upper region of the velocity in the right-hand one. This improves the quality of the relaxation without increasing the number of variables and the associated complexity.

At each node, the algorithm branches on one single velocity. Our branching strategy consists of branching on the velocity term with the worst relaxation (i.e., the one for which the difference between the solutions of the relaxed and original problem takes a maximum value). Let $v^{\text{CMILP}^{k^*}}$ be the vector containing the value of the p velocities v_r in the optimal solution of subproblem CMILP^k and $v^{\text{OMINLP}^{k^*}}$ be the equivalent vector for subproblem OMINLP^k. The branching velocity in node k is that with the largest distance between its optimal value in the original problem and the relaxation:

$$r^k = \arg \max_{r \in r_{\text{lin}}} (\text{abs}(v_r^{\text{CMILP}^{k^*}} - v_r^{\text{OMINLP}^{k^*}})) \tag{35}$$

If no optimal solution to OMINLP^k is avaiblable (i.e., OMINLP^k was found infeasible in a local search), the branching velocity is selected with the same equation but $v_r^{\text{OMINLP}^{k^*}}$ is then calculated as a function ϕ of the optimal values of $k_r^{\text{CMILP}^{k^*}}$ and $x^{\text{CMILP}^{k^*}}$:

$$\begin{aligned} v_r^{\text{OMINLP}^{k^*}} &= \phi(k_r^{\text{CMILP}^{k^*}}, x^{\text{CMILP}^{k^*}}) \\ &= \exp(k_r^{\text{CMILP}^{k^*}})\gamma_r \prod_{j=1}^{n+m} \exp(x_j^{\text{CMILP}^{k^*}})^{f_{rj}} \end{aligned} \tag{36}$$

Another important consideration when branching is the selection of the branching point, that is, the point in which the domain of the branching velocity will be split. One possible strategy consists of using the optimal solution to CMILP^k, $v_r^{\text{CMILP}^{k^*}}$, as the branching point. From numerical examples, we found that this strategy usually led to large CPU times, mainly because it produces the same solutions in both descendant nodes. In contrast, allocating this point close to one of the extreme points of $[v_r^{\text{LB},k}, v_r^{\text{UB},k}]$ is likely to produce a very easy subproblem and a very hard one. The same applies to the rule presented in ref 57, where the branching point is selected as $v_r^{\text{br},k} = 0.9v_r^{\text{LB},k} + 0.1v_r^{\text{UB},k}$ if $v_r^{\text{OMINLP}^{k^*}} \le v_r^{\text{mid},k}$ (with $v_r^{\text{mid},k} = (v_r^{\text{LB},k} + v_r^{\text{UB},k})/2$) and $v_r^{\text{br},k} = 0.1v_r^{\text{LB},k} + 0.9v_r^{\text{UB},k}$ otherwise. Another alternative, perhaps the most intuitive one, is using the bisecting rule, in which the interval is divided by its mid point, $v_r^{\text{mid},k}$. Particularly, we have obtained the best performance of the algorithm by applying one of the strategies presented in ref 82. This strategy relies on using a convex combination between the optimal solution $v_r^{\text{OMINLP}^{k^*}}$ and the midpoint of the interval $v_r^{\text{mid},k}$, as illustrated by eq 37:

$$v_r^{\text{br},k} = 0.5v_r^{\text{OMINLP}^{k^*}} + 0.5v_r^{\text{mid},k} \tag{37}$$

Again, if $v_r^{\text{OMINLP}^{k^*}}$ is not available, it is calculated as in eq 36. With this strategy, we concentrate the efforts around the optimal solution without compromising the balance between the complexity of the two subproblems.

4.2.2. Bound Contraction and Interval Analysis. The quality of the OLB strongly depends on the bounds imposed on the variables.[40] These bounds can be tightened during the performance of the algorithm using bound contraction techniques. In general, we can distinguish between two lines of bound tightening procedures: optimality-based bounds tightening (OBBT[55,58,83−87]) and feasibility-based bounds tightening (FBBT[55,66,83,87−91]).

OBBT derives tight bounds for n variables by solving $2n$ optimization problems, where each of the n variables is minimized and maximized subjected to the problem constraints. When n is large, this procedure becomes time-consuming. Consequently, OBBT is typically performed only in the root node prior to the global optimization procedure.[84] We implement the same strategy, using OBBT to improve the bounds of the p velocities v_r by solving subproblems OBLB and OBUB:

$$\begin{aligned} &(\text{OBLB}) \text{ for every r}: && \min v_r \\ & && \text{s.t. eqs } 1, 11, 12, 18, 23 \text{ and } 26\text{−}34 \\ &(\text{OBUB}) \text{ for every r}: && \max v_r \\ & && \text{s.t. eqs } 1, 11, 12, 18, 23 \text{ and } 26\text{−}34 \end{aligned}$$

To avoid cutting off feasible values of v_r, we use the linear relaxation CMILP to generate bounds on the variables. For those cases in which the computational burden of model OBLB/UB is large or the number of velocities is particularly high, we can relax the integer variables in these subproblems in order to expedite their solution. Note that this is done at the expense of obtaining weaker bounds for the velocities.

On the other hand, FBBT inherits the knowledge from recursive arithmetic intervals[47] in order to infer new bounds for the variables from the information provided by the problem constraints. Every time we branch in a node k, we modify the bounds for the branching velocity in the descendant subproblems as follows:

$$v_r^{\text{UB},k+1} = v_r^{\text{br},k} \tag{38}$$

$$v_r^{\text{LB},k+2} = v_r^{\text{br},k} \tag{39}$$

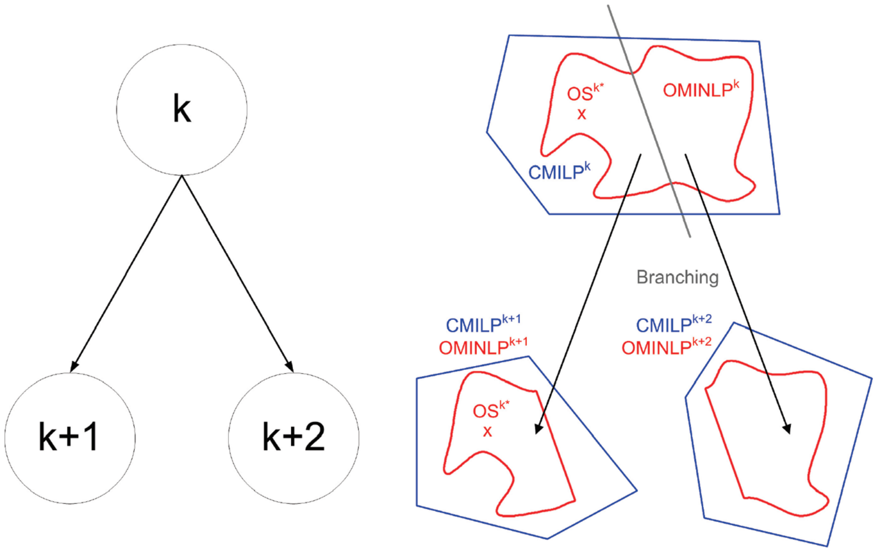

Figure 2. Scheme of sBB partitioning procedure. Solution OS^{k^*} belongs to the feasible space of subproblem OMINLP^{k+1}.

where $k+1$ and $k+2$ denote the left-hand side and right-hand side subproblems, respectively. Consider a hypothetical metabolite X_a for which the mass balance is described as follows:

$$\frac{dX_a}{dt} = 0 = 2v_1 + v_2 - 3v_3 \tag{40}$$

From this equation, we know that $v_1^{\mathrm{LB}} \geq (3v_3^{\mathrm{LB}} - v_2^{\mathrm{UB}})/2$ and $v_1^{\mathrm{UB}} \leq (3v_3^{\mathrm{UB}} - v_2^{\mathrm{LB}})/2$. Similar expressions can be derived to get bounds on v_2 and v_3. These equations improve the effect of the branching strategy by generating tighter bounds for variables others than the one on which we have branched. In general, the following expressions hold:

$$v_r^{\mathrm{LB},\,i} = \sum_{r' \left| \left(\frac{\mu_{ir'}}{-\mu_{ir}}\right) > 0\right.} \frac{\mu_{ir'}}{-\mu_{ir}} v_r^{\mathrm{LB}} + \sum_{r' \left| \left(\frac{\mu_{ir'}}{-\mu_{ir}}\right) < 0\right.} \frac{\mu_{ir'}}{-\mu_{ir}} v_r^{\mathrm{UB}}$$

$$r' \neq r \quad i = 1, ..., n \tag{41}$$

$$v_r^{\mathrm{UB},\,i} = \sum_{r' \left| \left(\frac{\mu_{ir'}}{-\mu_{ir}}\right) > 0\right.} \frac{\mu_{ir'}}{-\mu_{ir}} v_r^{\mathrm{UB}} + \sum_{r' \left| \left(\frac{\mu_{ir'}}{-\mu_{ir}}\right) < 0\right.} \frac{\mu_{ir'}}{-\mu_{ir}} v_r^{\mathrm{LB}}$$

$$r' \neq r \quad i = 1, ..., n \tag{42}$$

Note that each mass balance equation in which velocity r participates can potentially lead to new tighter bounds. To account for this, we introduce the index i in the bounds $v_r^{\mathrm{LB},i}$ and $v_r^{\mathrm{UB},i}$. The bounds obtained in each equation are finally compared in order to keep the tightest one:

$$v_r^{\mathrm{LB}} = \max(v_r^{\mathrm{LB,\,old}}, \max_{i=1}^{n}(v_r^{\mathrm{LB},\,i})) \tag{43}$$

$$v_r^{\mathrm{UB}} = \min(v_r^{\mathrm{UB,\,old}}, \min_{i=1}^{n}(v_r^{\mathrm{UB},\,i})) \tag{44}$$

Since during the FBBT procedure bounds may be updated, it may be worth it to repeat the process recursively in order to obtain tighter bounds. It is convenient to consider an iteration limit on the number of times that the procedure is performed. More sophisticated criteria (e.g., repeating the FBBT until the best improvement falls below a given tolerance) can also be used.

It is known that FBBT provides weaker bounds than OBBT.[63] However, it tends to be faster. One of the main advantages of FBBT is that it can detect infeasible subproblems prior to their optimization. A subproblem k is infeasible when $v_r^{\mathrm{LB},k} > v_r^{\mathrm{UB},k}$ for at least one r:

$$\exists r | v_r^{\mathrm{LB},\,k} > v_r^{\mathrm{UB},\,k} \rightarrow \mathrm{OMINLP}^k = \phi \tag{45}$$

OBBT and FBBT are thus valuable techniques for expediting the overall performance of the algorithm.

4.2.3. Strengthening Cuts. A special type of linear cuts that tighten the relaxation of OMINLP can be derived from the stoichiometric coefficients that relate the p velocities in the mass balance of every dependent metabolite i. Let us consider the example introduced in the previous section. Two cuts can be deduced from eq 40 as follows:

$$v_3 \geq \frac{2v_1}{3} \tag{46}$$

$$v_3 \geq \frac{v_2}{3} \tag{47}$$

In general, from any mass balance equation associated with metabolite i with $p(i)$ velocities in which only one μ_{ir} has a different sign than the remaining ones (i.e., $\exists r | \mu_{ir}\mu_{ir'} < 0 \ \forall r \neq r' \wedge \mu_{ir'}\mu_{ir''} > 0 \ \forall r', r'' \neq r, r' \neq r''$), it is possible to generate $p(i) - 1$ strengthening cuts according to eq 48:

$$v_r \geq \frac{\mu_{ir'}}{-\mu_{ir}} v_{r'} \quad \forall \ i, r' | \ \exists \ r | \mu_{ir}\mu_{ir'} < 0 \ \forall \ r \neq r' \ \wedge \ \mu_{ir'}\mu_{ir''} > 0$$

$$\forall \ r'' \neq r, r' \tag{48}$$

These inequalities can be obtained offline and added to CMILP before the optimization takes place. A major advantage of these cuts is that we can easily linearize them by applying the exponential transformation described before. Particularly, if we introduce eq 4 into eq 48 and replace the original X_i and K_r as

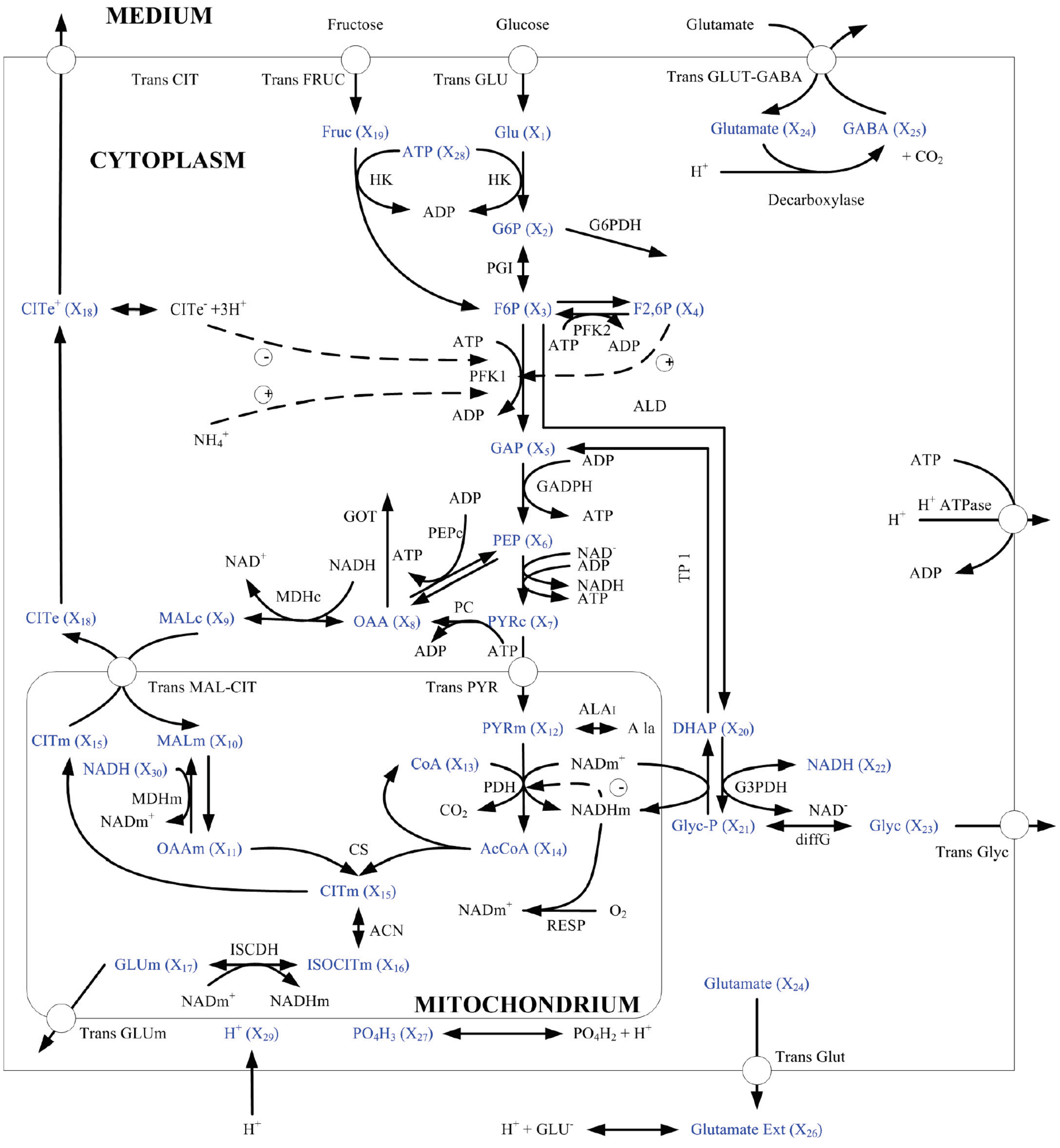

Figure 3. Citric acid production in *Aspergillus niger*. Adapted from ref 67. Dependent metabolites are highlighted.

described in eqs 15 and 16, we obtain eq 49:

$$(\exp k_r)\gamma_r \prod_{j=1}^{n+m} (\exp x_j)^{f_{rj}} \geq \frac{\mu_{ir'}(\exp k_r^{'})\gamma_{r'} \prod_{j=1}^{n+m} (\exp x_j)^{f_{r'j}}}{-\mu_{ir}} \qquad (49)$$

We can linearize these equations by taking logarithms in both sides of the inequality, similarly as we did in eq 18:

$$k_r + \ln(\gamma_r) + \sum_{j=1}^{n+m} f_{rj}x_j \geq \ln(\mu_{ir'}) + k_{r'} + \ln(\gamma_{r'}) - \ln(-\mu_{ir}) + \sum_{j=1}^{n+m} f_{r'j}x_j \qquad (50)$$

Note that the introduction of strengthening cuts does not require the addition of auxiliary variables in the model.

4.2.4. Bound Inheritance. In sBB algorithms, most of the time is spent in solving the lower bounding problem and identifying a good incumbent for CMILP^k. The customized sBB algorithm incorporates a strategy devised to alleviate the effect of this limitation.

Let OS$^{k^*}$ be the optimal solution of subproblem OMINLP^k. If none of the pruning criteria are met in this node, two descendant subproblems, OMINLP^{k+1} and OMINLP^{k+2}, will be created. Solution OS$^{k^*}$ must be feasible for at least one of these subproblems (see Figure 2). Let us assume, without loss of generality, that OS$^{k^*}$ belongs to the feasible space of OMINLP^{k+1}. Hence, we can obtain a good incumbent for CMILP^{k+1} by expressing

Table 1. Size of Citric Acid Models after Preprocessing[a]

	OMINLP			CMILP			
case	equations	CV	IV	PW_0	equations	CV	IV
B1	692	448	3	12	5339	1072	924
B2	692	448	3	10	5111	958	810
C1	692	445	6	10	5111	958	810
C2	692	445	6	30	7451	2098	1890
D1	692	442	9	15	5681	1243	1095
D2	692	442	9	10	5111	958	810
E1	692	436	15	12	5339	1072	924
E2	692	436	15	15	5681	1243	1095

[a] OMINLP: full-space problem. CMILP: MIP relaxation of OMINLP. CV: number of continuous variables. IV: number of integer variables. PW_0: number of piecewise sections in the initial iteration of the algorithm.

Table 2. Enzymes That Can Be Modifed in Each of the Instances of OMINLP

case	ME	subcase	modifiable enzymes	case	ME	subcase	modifiable enzymes
B	1	1	[40]	D	3	1	[1, 40, 60]
		2	[59]			2	[1, 40, 59]
C	2	1	[40, 59]	E	5	1	[1, 39, 40, 59, 60]
		2	[1, 40]			2	[1, 28, 40, 59, 60]

OS^{k*} in the space of variables of the linear relaxation. We accomplish this by applying a logarithmic transformation on the continuous variables, and by fixing the values of the binary variables according to the intervals of the piecewise approximation in which the original solution has fallen. This provides an integer feasible solution that is used as a starting value for the B&B solvers, thereby expediting the solution of the lower bounding problem in node $k + 1$. Note that this initialization scheme is only applicable to one descendant node.

5. COMPUTATIONAL RESULTS

The problem selected for testing the capabilities of our customized sBB algorithm is the maximization of the citric acid production in *Aspergillus niger* (see Figure 3). On the basis of the results of Polisetty et al.[67] and Pozo et al.,[68] we solve several instances of OMINLP, which differ in the number of reactions (ME) allowed for simultaneous modification. We assume that the 60 reactions included in the model can be modified by genetic manipulation. Note, however, that any practical solution should consider only a limited number of changes. In this specific case, Polisetty et al.[67] showed that by manipulating only 5 enzymes, it is possible to attain a solution close to the one found when all of the enzymes can be modified.

Here, we take the results from Pozo et al.[68] as a reference for comparison purposes. We focus on optimizing the system when only one, two, three, or five enzymes can be modified (case B, ME = 1; case C, ME = 2; case D, ME = 3; and case E, ME = 5). The nomenclature is the same used in Pozo et al.[68] The cases discussed in Table 8 of that paper are used to test the performance of the novel sBB method. A total of eight instances are solved with the customized sBB approach (see Tables 1 and 2). In all these cases, those enzymes that are not allowed for modification are fixed to their basal state. The maximum change for each enzyme is 5 fold over its basal state. The optimization constraints are the same as in the referenced paper.

Table 3. Parameters Setting in the sBB Algorithm

parameter	configuration
node selection	highest LB
CPLEX tolerance	0.00%
FBBT stop criterion	10 iterations
number of hyperplanes	50
branching point selection	see eq 37

Our results are compared with those obtained by the OA technique introduced by the authors in an earlier work[21,68] and also with the global optimization package BARON. With regard to the sBB and OA methods, it should be noted that both of them solve iteratively the same subproblems: the MILP-based relaxation CMILP and the bounded OMINLP. From numerical examples, we observed that both algorithms worked better when the binary variables associated with the genetic manipulations of the enzyme levels are fixed in the original problem according to the output of the linear MILP relaxation. For this reason, the lower bounds are generated by solving a bounded NLP instead of a bounded MINLP. [Note that the optimization task is posed as a maximization problem, so CMILP predicts upper bounds on the global optimum of OMINLP.]

In all of the examples, we used CPLEX 11.2.1 as MILP solver and CONOPT 3.14s for the NLPs, whereas BARON v.8.1.5[92] was employed to solve the full-space OMINLP problems. The algorithms were implemented in GAMS 23.0.2 on an Intel 1.2 GHz machine. An optimality tolerance of 2.00% was fixed in all of the cases.

The performance of the sBB algorithm depends on a series of factors that can be configured at will. The ones with the highest influence are the branching rule, the CPLEX tolerance, the stop criterion for the FBBT procedure, the selection of the branching point, the number of supporting hyperplanes, and the number of piecewise intervals. The particular configuration of the algorithm chosen to perform the calculations is given in Table 3. The only parameter that was particularly tuned for every single instance being solved was the number of piecewise intervals. The results obtained with the aforementioned sBB configuration are shown in Table 4, which also summarizes the performance of the other methods. Recall that the optimal solution reported corresponds to the best solution provided by the lower bounding problem when the termination criterion was met. Note also that all of the algorithms are compared on the basis of the CPU time required to attain a solution with an optimality gap of 2.00%. This is the same comparison criterion used in Pozo et al.[68] Other criteria could have been used instead. Nevertheless, we observed that the conclusions of the analysis are very similar for all of the cases.

As can be seen, the proposed methodology can solve all of the instances within the required tolerance. The same occurs with the OA, whereas BARON was not able to improve the starting point (which corresponds to the basal state solution) even after 3600 s of CPU time. This might be due to the use of generic techniques for building the relaxed upper bounding problem (when maximizing) that do not benefit from the particular structure of the GMA formalism.

The number of nodes explored in the sBB tree varies from one example to another without any clear tendency. However, the node in which the optimal solution is found is generally very close to the root node (20 first nodes) in all the instances except C1

Table 4. Comparison between the Best Results Obtained with the OA and the Customized sBB for Each Instance[a]

	sBB						BARON			OA				
case	PW_0[b]	nodes	NO	LB	UB	CPU	LB	UB	CPU	PW_0	PW_f	LB	UB	CPU
B1	12	34	10	25.82	26.05	36	12.36	—[c]	17	3	12	25.82	26.33	17
B2	10	198	12	12.37	12.57	229	12.36	—	9	3	15	12.35	12.51	45
C1	10	298	244	25.83	26.34	368	12.36	—	13	3	16	25.78	26.14	53
C2	30	26	15	25.82	26.34	61	12.36	—	39	3	12	25.82	26.33	18
D1	15	340	1	40.88	41.59	1155	12.36	—	52	3	18	40.88	41.42	167
D2	10	42	18	176.8	179.87	35	12.36	176.79	3600	3	12	176.79	180	18
E1	12	1	1	347.92	353.22	1	12.36	347.93	3600	7	9	347.93	353.35	6
E2	15	32	1	256.59	261.68	83	12.36	256.68	3600	3	18	256.59	261.32	1093

[a] PW_0: number of piecewise sections in the initial iteration of the algorithm. PW_f: number of piecewise sections in the last iteration of the algorithm. NO: node in which the optimal solution was found. LB: lower bound on the global optimum in mM min^{-1}. UB: upper bound on the global optimum in mM min^{-1}. CPU: CPU time in seconds. [b] Note that for the sBB algorithm, $PW_0 = PW_f$ as the number of piecewise sections is not modified throughout the algorithm. [c] BARON failed to provide a rigorous upper bound in cases with —.

(244). This is in consonance with the common observation that B&B algorithms may take a long time to verify optimality, although good (sometimes optimal) solutions are usually found in the early stages of the search.[48]

Regarding the optimal number of piecewise intervals, they range between 10 and 15. In only one case (C2) did the customized sBB perform better with a larger number of piecewise terms (i.e., 30). In contrast, in the OA, the optimal number of initial piecewise terms is small on average (i.e., 4), and even in the worst case, the algorithm works better with fewer intervals (i.e., 9) than in any instance of the sBB. Recall that, in each iteration of the OA algorithm, the total number of piecewise sections is increased by one, as the interval containing the optimal solution of CMILP is split into two subintervals. Since there is a binary variable associated with each of these new intervals, starting with too many sections is likely to lead to large instances that are hard to solve in short CPU times.

On the other hand, the final number of piecewise sections required by the OA exceeds the piecewise terms used in the sBB in all of the cases except two (i.e., C2 and E1). Let us note that in the OA method, the number of piecewise intervals is progressively increased to construct tighter relaxations, whereas in the customized sBB, bound tightening techniques and tailored branching rules allow reducing of the search space while keeping the number of piecewise terms constant in each subproblem. Henceforth, the sBB can produce a relaxation as tight as that of the OA with fewer piecewise sections.

With regard to the CPU time, the OA proved to be faster on average (177s compared to 253s). Specifically, it performed better than the sBB in more instances (6 vs 2). The reason for this to happen might be that at each iteration of the OA, we tighten the relaxation of the logarithm of all the velocities $|r_{\text{lin}}|$, whereas in the sBB, only one velocity is tightened at each branching point of the tree. This leads to more nodes and hence large CPU times. This finally results in a faster convergence of the lower and upper bounds in the case of the OA.

The customized sBB algorithm proved to be significantly faster than the OA in the two most difficult instances (i.e., those with a higher number of manipulations allowed). It should be noted that these results may vary according to the settings of each algorithm.

In order to better explore the advantages of the proposed method, we also solved instances of cases B, E, and F (the latter corresponding to ME = 4), where the best combination of enzymatic manipulations is searched. In these computations, no binary variables are fixed prior to the optimization. This exercise attempts to mimic the search for an optimal genetic modification in a biotechnological application.

As can be seen in Table 5, the solution identified for case B by both algorithms corresponds to the one obtained in case B1 (see Table 4). The same occurs with the solution obtained by the sBB for case E, which is the same as that of case E1. In contrast, the solution of case E found with the OA corresponds to a different combination of enzymes that leads to a lower (i.e., worse) value of the objective function. Solutions found for ME = 1 and ME = 5 did not improve those already reported in Polisetty et al.[67] and Pozo et al.[68] On the other hand, the solution obtained for case F, which implies modifying four enzymes, is indeed very close to the one found when all the enzymes are allowed to change. In addition, this solution can be attained modifying different combinations of enzymes.

Regarding the performance of the algorithms, both of them were capable of finding solutions with low optimality gaps in all of the instances, showing the sBB method the best performance. These results are partly due to the bound tightening techniques discussed in section 4.2.2, which were not included in the OA proposed in Pozo et al.[68] For instance, note that the customized sBB identifies the global optimum of cases E and F in just one node using 11 and 12 piecewise intervals respectively. One could think that the same result could be obtained with the OA using the same number of piecewise sections. However, when no bound contraction is performed, this setup does not allow for attaining the specified tolerance in one iteration of the OA algorithm. In fact, with this number of initial piecewise terms, the algorithm shows a worse performance than with 4 intervals (i. e., the final CPU times exceed those reported in Table 5 for four initial sections).

These results also indicate that good/optimal solutions can be found in the early stages of the search. Note that the global optimum of case B is identified in node 238 with sBB, but a solution with an optimality gap of 3% is found in node 1. This can be better seen in Figure 4: solutions very close to the global optimum are identified in the very first seconds of the execution of the algorithms, while the remaining time is spent in reducing the optimality gap. Particularly, the OA provides smaller optimality gaps than the sBB in the beginning of the search (i.e., in the

Table 5. Comparison between the Best Results Obtained with the Customized sBB and the OA for Each Instance[a]

	sBB								OA							
case	PW_0[b]	nodes	NO	OE	K_r	LB	UB	CPU	PW_0	PW_f	OE	K_r	LB	UB	CPU
B	3	296	238	[40]	[5.00]	25.82	26.25	484	2	12	[40]	[5.00]	25.82	26.19	864
E	11	1	1	[1, 39, 40, 59, 60]	[1.40, 0.92, 5.00, 5.00, 1.07]	347.92	353.81	91	4	7	[28, 39, 40, 59, 60]	[1.13, 1.45, 5.00, 5.00, 1.05]	347.26	353.75	203
F	12	1	1	[1, 39, 40, 59]	[1.23, 0.88, 5.00, 5.00]	347.25	351.41	50	4	7	[39, 40, 55, 59]	[1.53, 5.00, 1.10, 5.00]	347.26	351.71	78

[a] PW_0: number of piecewise sections in the initial iteration of the algorithm. NO: node in which the optimal solution was found. OE: enzymes r being modified in the optimal solution of the instance. K_r: optimal fold-change in activity of enzyme r. LB: lower bound on the global optimum in mM min^{-1}. UB: upper bound on the global optimum in mM min^{-1}. CPU: CPU time in seconds. PW_f: number of piecewise sections in the last iteration of the algorithm. [b] Note that for the sBB algorithm, $PW_0 = PW_f$ as the number of piecewise sections is not modified throughout the algorithm.

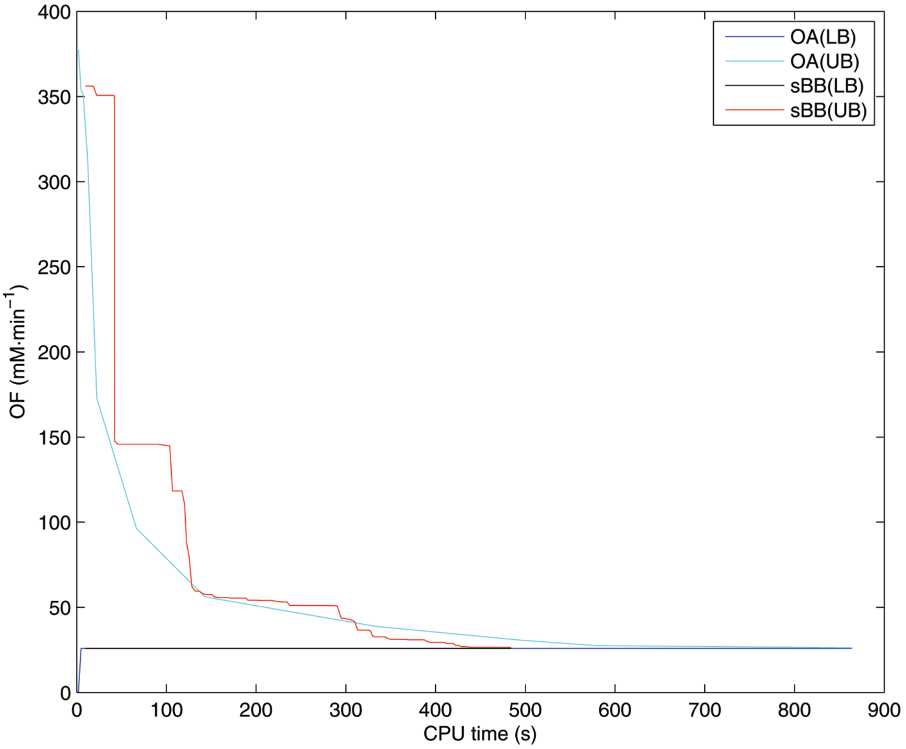

Figure 4. Evolution of the lower and the upper bounds of the global optimum of case B for the OA and the sBB algorithms.

first 300s). At this point, the tendency changes and the sBB shows better performance. This is due to the increase in the number of binary variables and hence in the complexity of the MILP subproblems solved by the OA. In contrast, the size of the MILP subproblems calculated in the sBB is kept constant in the nodes of the tree.

6. CONCLUSIONS

This paper has addressed the global optimization of metabolic networks described through the GMA formalism. A customized sBB algorithm that benefits from the specific structure of this type of model has been presented for this purpose. The optimization task was posed as a nonconvex MINLP in which integer variables denote the number of manipulations allowed. Tight bounds on the global optimum were obtained by constructing a linear MILP-based relaxation that exploits the mathematical structure of the GMA formalism. The method incorporates branching rules and bound contraction strategies devised to expedite the overall solution procedure.

Our strategy was compared against an outer approximation (OA) algorithm and the global optimization package BARON. Numerical results showed that the first two methods outperformed BARON in all of the instances under study. This is due to the quality of the MILP-based relaxation that is obtained by performing a logarithmic transformation on the power-law equations and approximating them by under- and overestimators. We also observed that none of these two methods (sBB and OA) proved to be superior in all of the cases. Nevertheless, the sBB showed a better performance in the most complicated instances, which is probably due to the ability of this strategy to reduce the problem domain without increasing the number of variables. Problems with a similar structure (i.e., with a large number of sigmoidal terms) may also benefit from the proposed strategy. Future work will focus on devising systematic tuning strategies that will improve the performance of the customized sBB algorithm.

The results obtained clearly show that we can tackle problems of moderate complexity when expressed as GMA models. One difficulty encountered when addressing the global optimization of complex metabolic networks is the current limited biological knowledge of some of these systems. While stoichiometric models can easily be constructed, GMA models require additional information that may not be available for large models. Although detailed GMA genome-wide models are far in the future, our results show that it is worth it to collect the required information, as we are able to obtain optimization results that go beyond those possible with stoichiometric models.

AUTHOR INFORMATION

Corresponding Author

*E-mail: gonzalo.guillen@urv.cat.

ACKNOWLEDGMENT

The authors wish to acknowledge support of this research work from the Spanish Ministry of Education and Science (projects DPI2008-04099, PHB2008-0090-PC, BFU2008-00196 and CTQ2009-14420-C02), the Spanish Ministry of External Affairs (projects HS2007-0006 and A/023551/09), and the Generalitat de Catalunya (FI programs).

NOMENCLATURE

Indexes

h = interval of the piecewise underestimation of the logarithm function
i = dependent metabolite
j = metabolite (dependent or independent)
l = supporting hyperplane
r = = flow, process, velocity

Sets

r_{lin} = set of processes r whose kinetic equations are linearized

Variables

K_r = fold-change in the basal state activity of enzyme governing process r
k_r = logarithm of the fold change in the basal state activity of enzyme governing process r
K_{r1} = auxiliary disaggregated variable associated with process r
K_{r2} = auxiliary disaggregated variable associated with process r
K_{r3} = auxiliary disaggregated variable associated with process r
t = = time
v_r = velocity of process r
vh_r^h = disaggregated variable associated with the hth term of the convex hull reformulation of the piecewise underestimator of velocity r
X_i = concentration of metabolite i
x_i = logarithm of the concentration of metabolite i
y_{r1} = binary variable associated with the first term of the convex hull of the disjunction of process r
y_{r2} = binary variable associated with the second term of the convex hull of the disjunction of process r
y_{r3} = binary variable associated with the third term of the convex hull of the disjunction of process r
z_r^h = binary variable associated with the hth term of the convex hull of the piecewise underestimation of velocity r

Parameters

$\boldsymbol{\delta}$ = sufficiently small parameter
γ_r = basal state activity of enzyme governing process r
μ_{ir} = stoichiometric coefficient of process r in the mass balance of metabolite i
a_r^h = slope of the segment used in interval h of the piecewise approximation of velocity r
b_r^h = vertical axis intercept of the segment used in interval h of the piecewise approximation of velocity r
f_{rj} = kinetic order of metabolite j in process r
H = total number of intervals in the piecewise underestimator of the logarithmic function
K_r^{LB} = lower bound on the fold change in the basal state activity of enzyme governing process r
K_r^{UB} = upper bound on the fold change in the basal state activity of enzyme governing process r
L = total number of supporting hyperplanes (linearization points)
m = total number of independent metabolites
ME = maximum number of enzymes allowed for modification
n = total number of dependent metabolites
p = total number of flows (processes) involved in the metabolic network under study
$p(i)$ = total number of flows (processes) involved in the mass balance of metabolite i
v_r^h = lower limit of interval h in the piecewise underestimator of velocity r
v_r^{h+1} = upper limit of interval h in the piecewise underestimator of velocity r
v_r^{LB} = lower bound on velocity r
v_r^{UB} = upper bound on velocity r
X_i^{LB} = lower bound on the concentration of metabolite i
X_i^{UB} = upper bound on the concentration of metabolite i

REFERENCES

(1) Guell, M.; van Noort, V.; Yus, E.; Chen, W. H.; Leigh-Bell, J.; Michalodimitrakis, K.; Ya-mada, T.; Arumugam, M.; Doerks, T.; Kuhner, S.; Rode, M.; Suyama, M.; Schmidt, S.; Gavin, A. C.; Bork, P.; et al. Transcriptome complexity in a genome-reduced bacterium. *Science (Washington, DC, U.S.)* **2009**, *326*, 1268–1271.

(2) Kuhner, S.; van Noort, V.; Betts, M. J.; Leo-Macias, A.; Batisse, C.; Rode, M.; Yamada, T.; Maier, T.; Bader, S.; Beltran-Alvarez, P.; Castano-Diez, D.; Chen, W. H.; Devos, D.; Guell, M.; Norambuena, T.; et al. Proteome organization in a genome-reduced bacterium. *Science (Washington, DC, U.S.)* **2009**, *326*, 1235–1240.

(3) Yus, E.; Maier, T.; Michalodimitrakis, K.; van Noort, V.; Yamada, T.; Chen, W. H.; Wodke, J. A.; Guell, M.; Martinez, S.; Bourgeois, R.; Kuhner, S.; Raineri, E.; Letunic, I.; Kalinina, O. V.; Rode, M.; et al. Impact of genome reduction on bacterial metabolism and its regulation. *Science (Washington, DC, U.S.)* **2009**, *326*, 1263–1268.

(4) Gibson, D. G.; Glass, J. I.; Lartigue, C.; Noskov, V. N.; Chuang, R. Y.; Algire, M. A.; Benders, G. A.; Montague, M. G.; Ma, L.; Moodie, M. M.; Merryman, C.; Vashee, S.; Krishnakumar, R.; Assad-Garcia, N.; Andrews-Pfannkoch, C.; et al. Creation of a Bacterial Cell Controlled by a Chemically Synthesized Genome. *Science (New York, N.Y.)* **2010**, .

(5) Vilaprinyo, E.; Alves, R.; Sorribas, A. Use of physiological constraints to identify quantitative design principles for gene expression in yeast adaptation to heat shock. *BMC Bioinf.* **2006**, *7*, 184.

(6) Feist, A. M.; Palsson, B. O. The growing scope of applications of genome-scale metabolic reconstructions using *Escherichia coli*. *Nat. Biotechnol.* **2008**, *26*, 659–667.

(7) Oberhardt, M. A.; Palsson, B. O.; Papin, J. A. Applications of genome-scale metabolic reconstructions. *Mol. Syst. Biol.* **2009**, *5*, 320.

(8) Vital-López, F.; Armaou, A.; Nikolaev, E.; Maranas, C. A. Computational Procedure for Optimal Engineering Interventions Using Kinetic Models of Metabolism. *Biotechnol. Prog.* **2006**, *22*, 1507–1517.

(9) Banga, J. Optimization in computational systems biology. *BMC Syst. Biol.* **2008**, 2–47.

(10) Bailey, J. Toward a science of metabolic engineering. *Science* **1991**, *252*, 1668–1675.

(11) Hatzimanikatis, V.; Floudas, C.; Bailey, J. Optimization of regulatory architectures in metabolic reaction networks. *Biotechnol. Bioeng.* **1996**, *52*, 485–500.

(12) Voit, E. Optimization in integrated biochemical systems. *Biotechnol. Bioeng.* **1992**, *40* (5), 572–582.

(13) Bailey, J. Lessons from metabolic engineering for functional genomics and drug discovery. *Nat. Biotechnol.* **1999**, *17*, 616–618.

(14) Marin-Sanguino, A.; Torres, N. Optimization of biochemical systems by linear programming and general mass action model representations. *Math. Biosci.* **2003**, *184* (2), 187–200.

(15) Marin-Sanguino, A.; Voit, E.; Gonzalez-Alzon, C.; Torres, N. Optimization of biotechnological systems through geometric programming. *Theor. Biol. Med. Model.* **2007**, 4–38.

(16) Alvarez-Vasquez, F.; Canovas, M.; Iborra, J.; Torres, N. Modeling, optimization and experimental assessment of continuous L-(−)-carnitine production by *Escherichia coli* cultures. *Biotechnol. Bioeng.* **2002**, *80* (7), 794–805.

(17) Torres, N. V.; Voit, E. O.; Gonzalez-Alcon, C. Optimization of nonlinear biotechnological processes with linear programming: Application to citric acid production by *Aspergillus niger*. *Biotechnol. Bioeng.* **1996**, *49*, 247–258.

(18) Alvarez-Vasquez, F.; Gonzalez-Alcon, C.; Torres, N. V. Metabolism of citric acid production by *Aspergillus niger*: model definition, steady-state analysis and constrained optimization of citric acid production rate. *Biotechnol. Bioeng.* **2000**, *70*, 82–108.

(19) Lin, H.; Bennett, G. N.; San, K. Y. Metabolic engineering of aerobic succinate production systems in *Escherichia coli* to improve process productivity and achieve the maximum theoretical succinate yield. *Metab. Eng.* **2005**, *7*, 116–127.

(20) Chang, Y.; Sahinidis, N. Optimization of metabolic pathways under stability considerations. *Comput. Chem. Eng.* **2005**, *29*, 467–479.

(21) Guillén-Gosálbez, G.; Sorribas, A. Identifying quantitaive operation principles in metabolic pathways: a systematic method for searching feasible enzyme activity patterns leading to cellular adaptive responses. *BMC Bioinf.* **2009**, *10*, 386.

(22) Sorribas, A.; Pozo, C.; Vilaprinyo, E.; Guillén-Gosálbez, G.; Jiménez, L.; Alves, R. Optimization and evolution in metabolic pathways: Global optimization techniques in Generalized Mass Action models. *J. Biotechnol.* **2010**, DOI: 10.1016/j.jbiotec.2010.01.026.

(23) Bower, J.; Bolouri, H. *Computational Modeling of Genetic and Biochemical Networks*; MIT Press: London, 2004.

(24) Orth, J. D.; Thiele, I.; Palsson, B. O. What is flux balance analysis? *Nat. Biotechnol.* **2010**, *28*, 245–248.

(25) Edwards, J.; Ibarra, R.; Palsson, B. In silico predictions of *Escherichia coli* metabolic capabilities are consistent with experimental data. *Nat. Biotechnol.* **2001**, *19* (2), 125–130.

(26) Forster, J.; Famili, I.; Fu, P.; Palsson, B.; Nielsen, J. Genome-scale reconstruction of the *Saccharomyces cerevisiae* metabolic network. *Genome Res.* **2003**, *13* (2), 244–253.

(27) Alper, H.; Jin, Y.; Moxley, J.; Stephanopoulos, G. Identifying gene targets for the metabolic engineering of lycopene biosynthesis in *Escherichia coli*. *Metab. Eng.* **2005**, *7* (3), 155–164.

(28) Cox, S.; Levanon, S.; Sanchez, A.; Lin, H.; Peercy, B.; Bennett, G.; San, K. Development of a metabolic network design and optimization framework incorporating implementation constraints: A succinate production case study. *Metab. Eng.* **2006**, *8* (1), 46–57.

(29) Pramanik, J.; Keasling, J. Stoichiometric model of *Escherichia coli* metabolism: Incorporation of growth-rate dependent biomass composition and mechanistic energy requirements. *Biotechnol. Bioeng.* **1997**, *56* (4), 398–421.

(30) Fong, S.; Burgard, A.; Herring, C.; Knight, E.; Blattner, F.; Maranas, C.; Palsson, B. In silico design and adaptative evolution of *E. coli* for production of lactic acid. *Biotechnol. Bioeng.* **2005**, *91* (5), 643–648.

(31) Voit, E. Design principles and operating principles: the yin and yang of optimal functioning. *Math. Biosci.* **2003**, *182*, 81–92.

(32) Voit, E. O.; Savageau, M. A. Accuracy of alternative representations for integrated biochemical systems. *Biochemistry* **1987**, *26*, 6869–6880.

(33) Sorribas, A.; Curto, R.; Cascante, M. Comparative characterization of the fermentation pathway of *Saccharomyces cerevisiae* using biochemical systems theory and metabolic control analysis: model validation and dynamic behavior. *Math. Biosci.* **1995**, *130*, 71–84.

(34) Cascante, M.; Curto, R.; Sorribas, A. Comparative characterization of the fermentation pathway of *Saccharomyces cerevisiae* using biochemical systems theory and metabolic control analysis: steady-state analysis. *Math. Biosci.* **1995**, *130*, 51–69.

(35) Curto, R.; Sorribas, A.; Cascante, M. Comparative characterization of the fermentation pathway of *Saccharomyces cerevisiae* using biochemical systems theory and metabolic control analysis: Model definition and nomenclature. *Math. Biosci.* **1995**, *130*, 25–50.

(36) Alves, R.; Vilaprinyo, E.; Hernandez-Bermejo, B.; Sorribas, A. Mathematical formalisms based on approximated kinetic representations for modeling genetic and metabolic pathways. *Biotechnol. Genet. Eng. Rev.* **2008**, *25*, 1–40.

(37) Alves, R.; Vilaprinyo, E.; Sorribas, A. Integrating Bioinformatics and Computational Biology: Perspectives and Possibilities for In Silico Network Reconstruction in Molecular Systems Biology. *Curr. Bioinf.* **2008**, *3*, 98–129.

(38) Smith, E.; Pantelides, C. A symbolic reformulation/spatial branch-and-bound algorithm for the global optimization of nonconvex MINLPs. *Comput. Chem. Eng.* **1999**, *23*, 457–478.

(39) Floudas, C.; Akrotirianakisa, I.; Caratzoulasa, S.; Meyera, C.; Kallrath, J. Global optimization in the 21st century: Advances and challenges. *Comput. Chem. Eng.* **2005**, *29*, 1185–1202.

(40) Grossmann, I.; Bigler, L. Part II. Future perspective on optimization. *Comput. Chem. Eng.* **2004**, *28*, 1193–1218.

(41) Horst, R.; Thoai, N.; Vries, L. D. A new simplicial cover technique in constrained global optimization. *J. Global Optimiz.* **1992**, *2*, 1–19.

(42) Falk, J.; Soland, R. An algorithm for separable nonconvex programming problems. *Manage. Sci.* **1969**, *15*, 550–569.

(43) Al-Khayyal, F. Generalized bilinear programming. Part I. Models, applications and linear programming relaxation. *Eur. J. Operat. Res.* **1992**, *60*, 306–314.

(44) Al-Khayyal, F.; Falk, F. Jointly constrained biconvex programming. *Math. Operat. Res.* **1983**, *8*, 273–286.

(45) R., H.; Tuy, H. On the convergence of global methods in multiextremal optimization. *J. Optimiz. Theory Appl.* **1987**, *54*, 253.

(46) Horst, R.; Tuy, H. *Global Optimization: Deterministic Approaches*, 2nd ed.; Springer-Verlag: Berlin, 1993.

(47) Ryoo, H.; Sahinidis, N. V. Global optimization of nonconvex NLPs and MINLPs with applications in process design. *Comput. Chem. Eng.* **1995**, *19* (5), 551–566.

(48) Ryoo, H.; Sahinidis, N. V. A branch-and-reduce approach to global optimization. *J. Global Optimiz.* **1996**, *8* (2), 107–138.

(49) Adjiman, C.; Androulakis, I.; Floudas, C. A global optimization method, αbb, for general twice-differentiable constrained NLPs: II. Implementation and computational results. *Comput. Chem. Eng.* **1998**, *22* (9), 1159–1179.

(50) Adjiman, C.; Androulakis, I.; Floudas, C. Global optimization of MINLP problems in process synthesis and design. *Comput. Chem. Eng.* **1997**, *21*, S445–S450.

(51) Adjiman, C.; Androulakis, I.; Maranas, C.; Floudas, C. A global optimization method, αBB, for process design. *Comput. Chem. Eng.* **1996**, *20*, S419–S424.

(52) Adjiman, C.; Dallwig, S.; Floudas, C.; Neumaier, A. A global optimization method, αBB, for general twice-differentiable constrained NLPs: I. Theoretical advances. *Comput. Chem. Eng.* **1998**, *22* (9), 1137–1158.

(53) Adjiman, C.; Floudas, C. Rigorous convex underestimators for general twice-differentiable problems. *J. Global Optimiz.* **1996**, *9* (1), 23–40.

(54) Adjiman, C.; Schweiger, C.; Floudas, C. Mixed-integer nonlinear optimization in process synthesis. *Handbook of Combinatorial Optimization*; Kluwer Academic Publishers: Norwell, MA, 1998.

(55) Smith, E. *On the Optimal Design of Continuous Processes*; Ph.D. thesis,Imperial College of Science, Technology and Medicine, University of London, 1996.

(56) Smith, E.; Pantelides, C. Global optimization of nonconvex MINLPs. *Comput. Chem. Eng.* **1997**, *21*, S791–S796.

(57) Epperly, T.; Pistikopoulos, E. A reduced space branch and bound algorithm for global optimization. *J. Global Optimiz.* **1997**, *11*, 287–311.

(58) Zamora, J.; Grossmann, I. A branch and contract algorithm for problems with concave univari-ate, bilinear and linear fractional terms. *J. Global Optimiz.* **1999**, *14*, 217–249.

(59) Kesavan, P.; Barton, P. Generalized branch-and-cut framework for mixed-integer nonlinear optimization problems. *Comput. Chem. Eng.* **2000**, *24*, 1361–1366.

(60) O'Grady, A.; Bogle, I.; Fraga, E. Interval analysis in automated design for bounded solutions. *Chem. Zvesti* **2001**, *55* (6), 376–381.

(61) Vaidyanathan, R.; El-Halwagi, M. Global optimization of nonconvex MINLPs by interval analysis. In *Global Optimization in Engineering Design*; Grossmann, I., Ed.; Kluwer Academic Publishers: Dordrecht, The Netherlands, 1996; pp 175−193.

(62) Zilinskas, J.; Bogle, I. Evaluation ranges of functions using balanced random interval arithmetic. *Informatica* **2003**, *14* (3), 403–416.

(63) Belotti, P.; Lee, J.; Liberti, L.; Margot, F.; Wächter, A. Branching and bounds tightening techniques for non-convex MINLP. *Optimiz. Methods Software* **2009**, *24* (4−5), 597–634.

(64) Sahinidis, N.; Grossmann, I. MINLP model for cyclic multiproduct scheduling on continuous parallel production lines. *Comput. Chem. Eng.* **1991**, *15*, 85–103.

(65) Sahinidis, N.; Grossmann, I. Reformulation of multi-period MINLP model for capacity expansion of chemical process. *Oper. Res.* **1992**, *40*, S127–S144.

(66) Carrizosa, E.; Hansen, P.; Messine, F. Improving interval analysis bounds by translations. *J. Global Optimiz.* **2004**, *29* (2), 157–172.

(67) Polisetty, P.; Gatzke, E.; Voit, E. Yield optimization of regulated metabolic systems using deterministic branch-and-reduce methods. *Biotechnol. Bioeng.* **2008**, *99* (5), 1154–1169.

(68) Pozo, C.; Guillén-Gosálbez, G.; Sorribas, A.; Jiménez, L. Outer approximation-based algorithm for biotechnology studies in systems biology. *Comput. Chem. Eng.* **2010**, DOI: 10.1016/j.compchemeng.2010.03.001.

(69) Savageau, M. Biochemical systems analysis. I. Some mathematical properties of the rate law for the component enzymatic reactions. *J. Theor. Biol.* **1969**, *25*, 365–369.

(70) Savageau, M. Biochemical systems analysis. II. The steady-state solutions for an n-pool system using a power-law approximation. *J. Theor. Biol.* **1969**, *25*, 370–379.

(71) Voit, E. *Computational Analysis of Biochemical Systems. A Practical Guide for Biochemists and Molecular Biologists*; Cambridge University Press: Cambridge, U.K., 2000.

(72) Vecchietti, A.; Sangbum, L.; Grossmann, I. Modeling of discrete/continuous optimization problems: characterization and formulation of disjunctions and their relaxations. *Comput. Chem. Eng.* **2003**, *27*, 433–448.

(73) Lee, S.; Grossmann, I. Global optimization of nonlinear generalized disjunctive programming with bilinear equality constraints: applications to process networks. *Comput. Chem. Eng.* **2003**, *27*, 1557–1575.

(74) Biegler, L.; Grossmann, I.; Margot, F.; Sahinidis, N.; Lee, J.; Waechter, A.; Belotti, P.; Castro, P.; Ruiz, J. CMU-IBM Cyber-Infrastructure for MINLP. http://www.minlp.org/ (accessed Oct 2010).

(75) Lu, H.-C.; Li, H.-L.; Gounaris, C.; Floudas, C. Convex relaxation for solving posynomial programs. *J. Global Optimiz.* **2010**, *46* (1), 147–154.

(76) Bergamini, M.; Scenna, N.; Aguirre, P. Global Optimal Structures of Heat Exchanger Networks by Piecewise Relaxation. *Ind. Eng. Chem. Res.* **2007**, *46*, 1752–1763.

(77) Bergamini, M.; Grossmann, I.; Scenna, N.; Aguirre, P. An improved piecewise outer-approximation algorithm for the global optimization of MINLP models involving concave and bilinear terms. *Comput. Chem. Eng.* **2008**, *32*, 477–493.

(78) Karuppiah, R.; Grossmann, I. Global optimization for the synthesis of integrated water systems in chemical processes. *Comput. Chem. Eng.* **2006**, *30*, 650–673.

(79) McCormick, G. *Nonlinear Programming, Theory, Algorithms, and Applications*; John Wiley & Sons: New York, 1983.

(80) Murtagh, B.; Saunders, M. *MINOS 5.1 user's guide. Technical Report SOL 83-20R*, Systems Optimization Laboratory, Stanford University: Palo Alto, CA, 1987.

(81) Schilling, G.; Pantelides, C. A simple continuous-time process scheduling formulation and a novel solution algorithm. *Comput. Chem. Eng.* **1996**, *20*, S1221–S1226.

(82) Tawarmalani, M.; Sahinidis, N. *Convexification and global optimization in continuous and mixed-integer nonlinear programming: Theory, algorithms, software and applications; Non-convex Optimization and Its Applications*; Kluwer Academic Publishers: Dordrecht, The Netherlands, 2002; Vol. 65.

(83) Hansen, P.; Jaumard, B.; Lu, S. An analytical approach to global optimization. *Math. Programming* **1991**, *52*, 227–254.

(84) Quesada, I.; Grossmann, I. Global optimization algorithm for heat exchanger networks. *Ind. Eng. Chem. Res.* **1993**, *32*, 487–499.

(85) Quesada, I.; Grossmann, I. A global optimization algorithm for linear fractional and bilinear programs. *J. Global Optimiz.* **1995**, *6*, 39–76.

(86) Quesada, I.; Grossmann, I. Global optimization of bilinear process networks and multicomponent flows. *Comput. Chem. Eng.* **1995**, *19* (12), 1219–1242.

(87) Liberti, L. Writing global optimization software. In *Global Optimization: From Theory to Implementation*; Liberti, L., Maculan, N., Eds.; Springer: Berlin, 2006; pp 211−262.

(88) Shectman, J.; Sahinidis, N. A finite algorithm for global minimization of separable concave programs. *J. Global Optimiz.* **1998**, 12, 1−36.

(89) Hentenryck, P. V.; Michel, L.; Deville, Y. Numerica, a Modeling Language for Global Optimization; MIT Press: Cambridge, MA, 1997; Vol. 65.

(90) Shectman, J.; Sahinidis, N. A finite algorithm for global minimization of separable concave programs. *J. Global Optimiz.* **1998**, *12*, 1–36.

(91) Messine, F. Deterministic global optimization using interval constraint propagation techniques. *RAIRO-RO* **2004**, *38* (4), 277–294.

(92) Sahinidis, N. V. BARON: a general purpose global optimization software package. *J. Global Optim.* **1996**, *8*, 201–205.