Global Optimization of Large-Scale Mixed-Integer Linear Fractional Programming Problems: A Reformulation-Linearization Method and Process Scheduling Applications

Dajun Yue
Dept. of Chemical and Biological Engineering, Northwestern University, Evanston, IL 60208

Gonzalo Guillén-Gosálbez
Dept. of Chemical Engineering, University Rovira i Virgili, Tarragona E-43007, Spain

Fengqi You
Dept. of Chemical and Biological Engineering, Northwestern University, Evanston, IL 60208

DOI 10.1002/aic.14185
Published online July 30, 2013 in Wiley Online Library (wileyonlinelibrary.com)

Mixed-integer linear fractional program (MILFP) is a class of mixed-integer nonlinear programs (MINLP) where the objective function is the ratio of two linear functions and all constraints are linear. Global optimization of large-scale MILFPs can be computationally intractable due to the presence of discrete variables and the pseudoconvex/pseudoconcave objective function. We propose a novel and efficient reformulation–linearization method, which integrates Charnes–Cooper transformation and Glover's linearization scheme, to transform general MILFPs into their equivalent mixed-integer linear programs (MILP), allowing MILFPs to be globally optimized effectively with MILP methods. Extensive computational studies are performed to demonstrate the efficiency of this method. To illustrate its applications, we consider two batch scheduling problems, which are modeled as MILFPs based on the continuous-time formulations. Computational results show that the proposed approach requires significantly shorter CPU times than various general-purpose MINLP methods and shows similar performance than the tailored parametric algorithm for solving large-scale MILFP problems. Specifically, it performs with respect to the CPU time roughly a half of the parametric algorithm for the scheduling applications. © 2013 American Institute of Chemical Engineers *AIChE J*, 59: 4255–4272, 2013

Keywords: mixed-integer nonlinear programs, mixed-integer fractional programming, reformulation, mixed-integer linear programs, linearization

Introduction

Mixed-integer linear fractional program (MILFP) is a class of special type of nonconvex mixed-integer nonlinear program (MINLP).[1,2] An MILFP includes both continuous and discrete variables and seeks to optimize an objective function that is expressed in general form as the ratio of two linear functions subject to linear constraints. In process systems engineering, MILFP models usually occur in the planning and scheduling optimization for chemical manufacturing, petroleum refinery, polymerization processes, and paper production.[3–9] When continuous-time formulations are used,[10,11] the objective function of a scheduling problem might involve maximizing the cyclic profit rate or productivity, both of which are usually expressed as the ratio of two linear functions with the numerator as the profit and the denominator as the cycle time or makespan.[12–15] With this type of objective function, the scheduling problems are formulated as MILFPs, when all the constraints are linear. Real-world scheduling problems can be quite large-scale and global optimization is important to achieve the best process performance. However, MILFPs are nonconvex MINLPs and Non-deterministic Polynomial-time hard (NP-hard) in nature. Therefore, MILFPs can be hard to optimize to global optimality, because of their combinatorial nature and the pseudoconvexity of the objective functions.[1,2] Therefore, it is the goal of this work to develop an efficient solution strategy for the global optimization of large-scale MILFPs.

In this article, we propose a novel approach to reformulate general MILFP problems into an equivalent mixed-integer linear programming (MILP) form. The proposed reformulation–linearization method is based on the integration of Charnes–Cooper transformation[16] and the Glover's linearization scheme,[12] presented as an extension of the seminal Charnes–Cooper procedure to the particular mixed-integer case. This approach requires the addition of auxiliary continuous variables and constraints, but allows for the global optimization of MILFP problems using standard

Correspondence concerning this article should be addressed to F. You at you@northwestern.edu.

MILP methods, such as the branch-and-cut algorithms.[17] To the best of our knowledge, this is the first method that presents an exact MILP reformulation of the general MILFP problem with both discrete and continuous variables. We note that Li[18] and Wu[19] applied the Charnes–Cooper transformation procedure and similar linearization approach to pure 0–1 fractional programming problems in their respective works. Billionnet proposed efficient approximation algorithms with detailed time complexity analysis for 0–1 fractional knapsack problems.[20,21] In this article, after presenting the major properties of MILFPs and the existing solution methods, we introduce the proposed reformulation–linearization method and discuss its main properties. We also perform a side-by-side comparison between the proposed method and the tailored parametric algorithm, which has been shown to perform more efficiently than general-purpose MINLP and global optimization methods.[6] Extensive computational examples are presented to demonstrate the efficiency of our method for solving large-scale MILFP problems with hundreds or thousands of binary variables. To illustrate the potential of this approach, we consider two scheduling problems of multipurpose batch plants. Both scheduling problems are modeled as MILFPs based on continuous-time formulations.[21–24] The results clearly show that the CPU time can be lowered in several orders of magnitude when using the proposed approach, taking as reference general-purpose MINLP solvers, including DICOPT and SBB, and the global optimizer BARON. In addition, the proposed method performs similarly to the tailored parametric algorithm.

The major novelties of this article are summarized below:

- An efficient reformulation–linearization method for transforming MILFPs into their equivalent MILP problems, allowing the global optimization of large-scale MILFPs using efficient MILP methods within modest computational times.
- A comprehensive comparison between the proposed approach and various MINLP and global optimization methods as well as the tailored parametric algorithm.
- Two applications of the proposed method on optimal scheduling of flexible batch chemical processes.

The rest part of this article is organized as follows. We first review the properties of MILFP problems, major MINLP and global optimization methods, and the tailored parametric algorithm. We then introduce and discuss the proposed reformulation–linearization method, followed by numerical examples and case studies on batch process scheduling. This article is concluded in the last section.

Mixed-Integer Linear Fractional Program

An MILFP includes both continuous and discrete variables. All the constraints of an MILFP are linear, and the objective function is expressed as the ratio of two linear functions. Mathematically, a general MILFP can be formulated as the following problem (P0)

$$(\mathbf{P0})\ \max\ \frac{A_0+\sum_{i\in I}A1_ix_i+\sum_{j\in J}A2_jy_j}{B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j} \tag{1}$$

$$\text{s.t. } C_{0k}+\sum_{i\in I}C1_{ik}x_i+\sum_{j\in J}C2_{jk}y_j=0,\ \forall k\in K, \tag{2}$$

$$x_i\geq 0,\ \forall i\in I \text{ and } y_j\in\{0,1\},\ \forall j\in J \tag{3}$$

where x_i are continuous variables and y_j are discrete variables. For problem (P0), it is assumed that the denominator $B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j>0$ for all feasible solutions and all inequalities are converted into equalities through the use of slack variables.[25] Thus, problem (P0) includes $|I|$ continuous variables, $|J|$ binary variables, and $|K|$ equality constraints.

Due to the nonconvexities/nonlinearities in the objective function and the combinatorial nature given by the existence of 0–1 variables, large-scale MILFP problems with hundreds or thousands of binary variables are hard to optimize globally with general-purpose solution methods. In the ensuing sections, we review the major properties of MILFP problems, as well as the main MINLP methods for solving them.

Properties of MILFP problems

We start talking about the nonlinear relaxation of an MILFP, in which all the binary variables $y_j\in\{0,1\},\ \forall j\in J$ are treated as continuous variables $0\leq y_j\leq 1,\ \forall j\in J$. The nonlinear relaxation of problem (P0) is given below, and it is denoted as problem (RMILFP).

$$(\mathbf{RMILFP})\ \max\ \frac{A_0+\sum_{i\in I}A1_ix_i+\sum_{j\in J}A2_jy_j}{B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j} \tag{4}$$

$$\text{s.t. } C_{0k}+\sum_{i\in I}C1_{ik}x_i+\sum_{j\in J}C2_{jk}y_j=0,\ \forall k\in K, \tag{5}$$

$$x_i\geq 0, \forall i\in I \text{ and } 0\leq y_j\leq 1,\ \forall j\in J \tag{6}$$

The problem (RMILFP) is a linear fractional program (LFP), in which all the variables are continuous, all the constraints are linear, and the objective function is the ratio of two linear functions. The LFP has been well-studied in the past decades, and its major properties are summarized below:

Property 1 *The objective function of problem (RMILFP) is both pseudoconvex and pseudoconcave.*

Property 2 *The objective function of problem (RMILFP) is strictly quasiconvex and strictly quasiconcave.*

Property 3 *Every local optimum of problem (RMILFP) is also its global solution.*

The proofs of these properties are discussed in Chapter 11.4, "Linear Fractional Programming" in Bazaraa et al.[26] These properties of problem (RMILFP) can be exploited in the efficient solution of the MILFP problem (P0) with general MINLP methods.

Based on Property 3, You et al.[6] concluded that the relaxation problem (RMILFP) or a nonlinear programming subproblem can be solved to global optimality by using a local NLP solver, if some of the binary 0–1 variables are fixed. The standard branch-and-bound method[27,28] relies on solving a sequence of relaxed NLP subproblems.[29] Hence, this solution method guarantees the global optimality of the solution found within a desired epsilon tolerance. The extended cutting plane method by definition can handle pseudoconvex/pseudoconcave functions,[30,31] and therefore it can also guarantee the global optimality of the final solution. In summary, we can obtain the global optimum of the MILFP problem (P0) with an MINLP method that can handle pseudoconvex/

pseudoconcave objective functions, such as the standard branch-and-bound algorithm[27,28] and the extended cutting plane methods.[30,31]

Lemma 1 *A global optimal solution of the MILFP problem (P0) can be obtained by solving it with MINLP methods that can handle pseudoconvex/pseudoconcave objective functions, such as standard branch-and-bound and extended cutting plane method.*

MINLP methods for solving MILFP problems

Because an MILFP has both continuous and discrete variables, and it includes a nonlinear objective function with nonconvexities, it is also a nonconvex MINLP problem. Thus, global optimization methods for nonconvex MINLP problems can be utilized for solving MILFP problems. Global optimization methods for MINLP problems include, but are not limited to, the branch and reduce method,[32] the α-BB method,[33] and the spatial branch-and-bound search method for bilinear and linear fractional terms.[34,35] All these methods rely on a branch and bound solution procedure. The difference among these methods lies on the definition of the convex envelopes for computing the lower bound, and on the strategy to perform the branching on the discrete and continuous variables. The computer code BARON is a general-purpose global optimizer based on the branch-and-reduce method,[36] and it is available in a number of optimization modeling systems, such as GAMS.[37]

In addition to global optimization methods, MINLP methods that rely on convexity assumptions can also be utilized. Lemma 1 in the previous section allows us to globally optimize MILFP problems with some convex MINLP methods, such as the standard branch-and-bound method[30] and the extended cutting plane method,[31,32] which are potentially faster than using a global optimizer. Note that the Generalized Benders Decomposition algorithm[38] and Outer-Approximation method[39,40] do not guarantee global optimality when solving MILFP problems. This is because both Generalized Benders Decomposition and Outer-Approximation rely on the assumption that the functions are convex when predicting the lower/upper bounds with the master problem. If there is a pseudoconvex function, these two methods could not guarantee that the function linearizations would always be valid supporting hyperplanes. Thus, the linearization of a pseudoconvex function might cut off a part of the feasible region, and may not underestimate the objective function (see Figure 1). A similar issue was discussed in the work by Jain and Grossmann.[41]

We next review the major computer codes for convex MINLPs. The code α-ECP implements the extended cutting plane method.[31,32] The code SBB, which is available in GAMS, implements the standard branch-and-bound method.[27,28] The program DICOPT is an MINLP solver that is based on the Outer-Approximation algorithm, and is also available in GAMS. The open source MINLP solver Bonmin[42] implements an extension of the branch-and-cut outer-approximation algorithm that was proposed by Quesada and Grossmann,[43] as well as the branch-and-bound and outer-approximation method.

Parametric algorithm for MILFP

Besides the general-purpose MINLP methods, an alternative to solving MILFP problems is to use the parametric algorithm[25,44,45] by successively solving a sequence of MILP subproblems.[6,13] This algorithm is based on a parametric programming approach for defining an optimal-value function $F(q)=\max\{N(x,y)-q\cdot\ D(x,y)|(x,y)\in S\}$, where q is a parameter, $N(x,y)=A_0+\sum_{i\in I}A1_ix_i+\sum_{j\in J}A2_jy_j$, $D(x,y)=B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j$, and S is the feasible region of problem (P0) given by $S:\{D(x,y)>0$ and $C_{0k}+\sum_{i\in I}C1_{ik}x_i+\sum_{j\in J}C2_{jk}y_j=0,\ \ \forall k$ and $x_i\geq 0,\ \forall i\in I$ and $y_j=\{0,1\},\ \forall j\in J\}$. You et al.[6] showed that the MILFP problem (P0) is equivalent to the parametric program $F(q)$, that is, $F(q^*)=\max\{N(x,y)-q^*\cdot D(x,y)|(x,y)\ \in S\}=0$ if and only if $q^*=N(x^*,y^*)/D(x^*,y^*)=\max\ \{N(x,y)/D(x,y)|(x,y)\in S\}$.

Thus, the global optimal solution of the MILFP problem (P0) can be obtained by using a generalized Newton's method to solve the equation $F(q)=0$. For parameter q_m the subgradient of the parametric program $F(q_m)$ is identified as $-D(x^m,y^m)$, which is the negative value of the denominator evaluated at the optimal solution of (x^m,y^m). The subgradient can then be utilized in the Newton's algorithm, resulting in the following iterations

$$\begin{aligned}q_{m+1}&=q_m-\frac{F(q_m)}{F'(q_m)}=q_m-\frac{F(q_m)}{-D(x^m,y^m)}\\&=q_m+\frac{N(x^m,y^m)-q_m\cdot D(x^m,y^m)}{D(x^m,y^m)}=\frac{N(x^m,y^m)}{D(x^m,y^m)}\end{aligned}$$

This iteration and updating procedure constitute the core of the parametric algorithm for solving MILFP problems. The major steps of this algorithm are summarized as follows:

STEP 1. *Set* $q_1=0$, *initialize m by setting m = 1.*

STEP 2. *Solve the MILP problem* $F(q_m)=\max\{N(x,y)-q_m\cdot D(x,y)|(x,y)\in S\}$, *and denote the optimal solution as* (x^m,y^m).

Step 3. *If* $0\leq F(q_m)<\delta$ *(optimality tolerance), stop and output* (x^m,y^m) *as the optimal solution; otherwise, let*

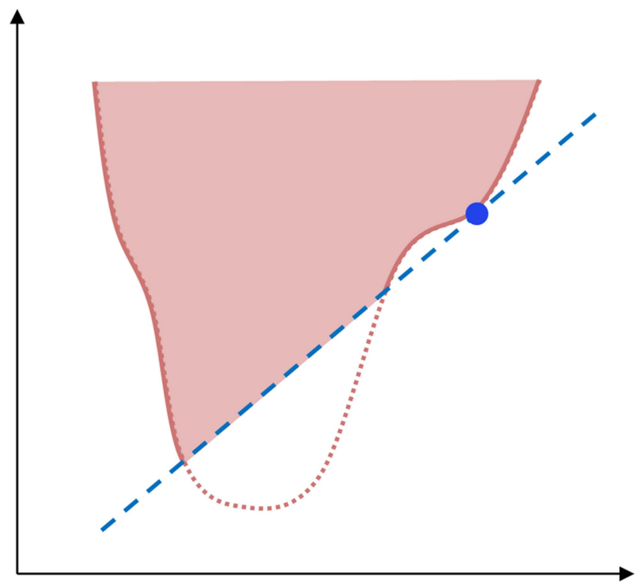

Figure 1. Linearization might cut off a part of the feasible region for a pseudoconvex function.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com]

$q_{m+1}=\frac{N(x^m,y^m)}{D(x^m,y^m)}$ and go to Step 2 to replace m with m+1 and q_m with q_{m+1}.

With this algorithm, we can solve a sequence of MILP subproblems to obtain the global optimal solution of an MILFP problem. Each MILP subproblem has the same size as the MILFP problem that includes $|I|$ continuous variables, $|J|$ binary variables, and $|K|$ constraints. The only difference is that the objective function is in a linear, parametric form $N(x,y)-q\cdot D(x,y)$. Parametric algorithm allows us to take advantage of the efficient MILP methods to globally optimize nonconvex MILFP problems. The memory tends to be rather small compared with general-purpose MINLP methods, especially for large-scale MILFP problems. Besides, this algorithm converges quadratically for MILFP problems, because it is based on the Newton's method.[9]

Reformulation–Linearization Method

Charnes and Cooper[16] proposed a method to reformulate a LFP into an equivalent linear program, by introducing an additional continuous variable and an additional constraint (see the Appendix). This method is known as the Charnes–Cooper transformation and has been shown to be very efficient for solving continuous LFPs.[14,15] However, Charnes–Cooper transformation cannot be directly applied to MILFP problems as it is restricted to the case where all the variables are continuous.

We propose a novel and generalized approach to reformulate MILFP problems into an equivalent MILP form. Our approach is based on the reformulation of Charnes–Cooper and the Glover's linearization scheme. The former allows converting the MILFP into an MINLP, whereas the latter transforms this MINLP into an equivalent MILP. Details of the reformulation–linearization method and a side-by-side comparison with the parametric algorithm are presented in the following sections.

Equivalent MILP reformulation

Our proposed reformulation–linearization method includes two steps. We consider again the MILFP problem in general form given as problem (P0)

$$(\mathbf{P0})\ \max\ \frac{A_0+\sum_{i\in I}A1_ix_i+\sum_{j\in J}A2_jy_j}{B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j} \tag{7}$$

$$\text{s.t. } C_{0k}+\sum_{i\in I}C1_{ik}x_i+\sum_{j\in J}C2_{jk}y_j=0,\quad \forall k\in K \tag{8}$$

$$x_i\geq 0,\forall i\in I\,\text{and}\,y_j\in\{0,1\},\quad \forall j\in J, \tag{9}$$

where $B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j>0$.

The first step of the reformulation is similar to the Charnes–Cooper transformation.[16] We introduce a new variable u and a set of variables z_i, such that $u=\frac{1}{B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j}$, and $z_i=\frac{x_i}{B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j}=u\cdot x_i$. The variable u is positive and the variables z_i are nonnegative, because $B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j>0$ for all feasible solutions of problem (P0).

The fractional objective function (7) can then be transformed into a linear function with the introduction of new variables,

$$\frac{A_0+\sum_{i\in I}A1_ix_i+\sum_{j\in J}A2_jy_j}{B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j}=A_0\cdot u+\sum_{i\in I}A1_iz_i+\sum_{j\in J}A2_j\cdot\left(y_j\cdot u\right) \tag{10}$$

This is similar to the Charnes–Cooper transformation except that we only apply the variable transformations to the continuous variables. Please note that we do not transform the binary variables y_j, so the reformulation results in bilinear terms $y_j\cdot u$ in the objective function.

Because variable u is positive, we multiply both sides of constraint (8) by u and take advantage of the relationship that $z_i=u\cdot x_i$. The resulting equality is given below,

$$\begin{aligned}&C_{0k}\cdot u+\sum_{i\in I}C1_{ik}\cdot x_i\cdot u+\sum_{j\in J}C2_{jk}\cdot\left(y_j\cdot u\right)\\&=C_{0k}\cdot u+\sum_{i\in I}C1_{ik}\cdot z_i+\sum_{j\in J}C2_{jk}\cdot\left(y_j\cdot u\right)=0,\forall k\in K,\end{aligned} \tag{11}$$

Please note that constraints (11) similarly include the bilinear terms $y_j\cdot u$.

Because $u=\frac{1}{B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j}$, we include the following constraint in the reformulation to define the variable u.

$$\begin{aligned}1&=u\cdot\left(B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j\right)=B_0\cdot u+\sum_{i\in I}B1_i\cdot x_i\cdot u\\&+\sum_{j\in J}B2_j\cdot\left(y_j\cdot u\right)=B_0\cdot u+\sum_{i\in I}B1_i\cdot z_i+\sum_{j\in J}B2_j\cdot\left(y_j\cdot u\right)\end{aligned} \tag{12}$$

Thus, the MILFP problem (P0) can be transformed into the following equivalent MINLP problem (P1)

$$(\mathbf{P1})\qquad \max A_0\cdot u+\sum_{i\in I}A1_iz_i+\sum_{j\in J}A2_j\cdot\left(y_j\cdot u\right) \tag{13}$$

$$\text{s.t. } C_{0k}\cdot u+\sum_{i\in I}C1_{ik}\cdot z_i+\sum_{j\in J}C2_{jk}\cdot\left(y_j\cdot u\right)=0,\ \forall k\in K \tag{14}$$

$$B_0\cdot u+\sum_{i\in I}B1_i\cdot z_i+\sum_{j\in J}B2_j\cdot\left(y_j\cdot u\right)=1 \tag{15}$$

$$u\geq 0, z_i\geq 0,\forall i\in I\ \text{and}\ y_j\in\{0,1\},\ \forall j\in J \tag{16}$$

The only nonlinear terms in the MINLP problem (P1) are the bilinear terms $y_j\cdot u$, which are the products of a binary variable y_j and a continuous variable u. This type of bilinear terms can be exactly linearized by introducing a number of auxiliary variables and constraints, following the Glover's linearization scheme.[12] We introduce a set of auxiliary variables w_j, such that $w_j=y_j\cdot u$. Thus, the MINLP problem (P1) can be further linearized into an equivalent MILP problem (PR), which is given below

$$(\mathbf{PR})\ \ \max\ A_0\cdot u+\sum_{i\in I}A1_iz_i+\sum_{j\in J}A2_j\cdot w_j \tag{17}$$

$$\text{s.t. } C_{0k}\cdot u+\sum_{i\in I}C1_{ik}\cdot z_i+\sum_{j\in J}C2_{jk}\cdot w_j=0,\ \forall k\in K \tag{18}$$

$$B_0\cdot u+\sum_{i\in I}B1_i\cdot z_i+\sum_{j\in J}B2_j\cdot w_j=1 \tag{19}$$

$$w_j \le u, \ \forall j \in J, \tag{20}$$

$$w_j \le M \cdot y_j, \ \forall j \in J, \tag{21}$$

$$w_j \ge u - M \cdot (1 - y_j), \ \forall j \in J, \tag{22}$$

$$u \ge 0, z_i \ge 0, \forall i \in I \text{ and } w_j \ge 0, \ y_j \in \{0,1\}, \ \forall j \in J \tag{23}$$

where M is a sufficiently large number. Constraint (21) implies that if y_j is zero, then w_j should be zero; constraints (20) and (22) impose that if y_j is one, then w_j should be equal to u. Thus, constraints (20)–(22) are linearization constraints for $w_j = y_j \cdot u$.

Unless otherwise specified, the value of M is estimated heuristically throughout the article. Ideally, M should be its tightest possible value, namely the upper bound of variable u, which could be obtained by maximizing u subject to the constraints of the original model [see problem (P-M) in Appendix]. However, we found that the value of M has little effect on the computational performance (numerical examples provided in Appendix), thus the solution of problem (P-M) is not necessary.

After using the reformulation–linearization method, we transform the MILFP problem (P0) into the MILP problem (PR), which has $|J|$ binary 0–1 variables, $|I| + |J| + 1$ continuous variables, and $3 \cdot |J| + |K| + 1$ constraints. Compared to the MILFP problem (P0), which includes $|I|$ continuous variables, $|J|$ binary variables, and $|K|$ constraints, the reformulation does not introduce any additional integer variable, but requires $|J| + 1$ additional continuous variables and $3 \cdot |J| + 1$ constraints.

Because the MILFP problem (P0) is equivalent to the MINLP problem (P1), which is also equivalent to the MILP problem (PR), the problem (PR) is equivalent to the problem (P0). The derivation of the reformulation–linearization method reveals the one-to-one mapping relationship between the solutions of the two problems, because only variable substitution and exact linearization are performed during the reformulation process. These properties are summarized in the following lemmas.

Lemma 2 *The MILP problem (PR) is equivalent to the MILFP problem (P0).*

Lemma 3 *If (u^0, w^0, z^0, y^0) is a feasible solution of the MILP reformulation problem (PR), then there is a feasible solution (x^0, y^0) of the general MILFP problem (P0), such that $x_i^0 = z_i^0 / u^0$ and $w_j^0 = y_j^0 \cdot u^0$; vice versa.*

Lemma 4 *If (u^*, w^*, z^*, y^*) is the global optimal solution of the MILP reformulation problem (PR), then the global optimal solution of the general MILFP problem (P0) is given by (x^*, y^*), which satisfies $x_i^* = z_i^* / u^*$ and $w_j^* = y_j^* \cdot u^*$; vice versa.*

We note that the Charnes–Cooper transformation can be considered as a special case of the proposed reformulation–linearization method. The reason is that if binary variables y_j are not included in the MILFP problem (P0), constraints (20), (21), and (22) can be eliminated, and all the terms with variables w_j in objective function (17) and in constraints (18) and (19) can be removed. As a result, the reformulated MILP problem (PR) would become the equivalent linear program given in (A6)–(A8), which is the one reformulated from the LFP with Charnes–Cooper transformation (see the Appendix for details). Therefore, the proposed reformulation–linearization method is a generalization of the Charnes–Cooper transformation to account for problems with integer variables.

Major properties and comparison with the parametric algorithm

The major advantage of the proposed reformulation–linearization approach is that we can globally optimize a nonconvex MILFP problem by solving a similar size MILP problem, only once. Similar to the parametric algorithm, the proposed reformulation–linearization method can take advantage of the state-of-the-art MILP solvers, which have become more and more efficient in the past decades.[46–48] Because no nonlinear programming subproblem needs to be solved, the memory usage during the computational process tends to be rather small compared with using general MINLP and global optimization methods, especially for large-scale MILFP problems.

Table 1 shows the problem sizes of the original MILFP problem (P0) and the reformulated MILP problem (PR). We note that the linearization does not increase the number of discrete variables, and only needs some auxiliary variables and additional constraints, of which the numbers are both proportional to the number of binary variables, $|J|$.

The parametric algorithm has been shown to be much more efficient than general-purpose MINLP methods for solving large-scale MILFP problems.[6] As discussed in the previous section, the parametric algorithm relies on the solution of a sequence of MILP problems to converge to the global optimal solution of the MILFP problem. As shown in Table 1, each MILP subproblem of the parametric algorithm has the same size of the original MILFP problem, and thus has the same number of binary variables as the MILP reformulation problem (PR). Because the computational complexity of an MILP problem largely depends on the number of integer variables and also because both MILP problems have similar structure, the reformulated MILP problem (PR) might be computationally a bit more expensive than the MILP subproblem of the parametric algorithm. However, the proposed reformulation–linearization approach only requires the solution of one MILP (the reformulated MILP), while the parametric algorithm usually needs multiple iterations despite its quadratic convergence rate. Therefore, the proposed reformulation–linearization method with a single iteration should have comparable computational performance as the parametric algorithm when

Table 1. Comparison in Terms of Problems Sizes and Iterations Numbers between the Proposed Reformulation-Linearization Method and the Parametric Algorithm

	General MILFP Problem (P0)	Reformulated MILP Problem (PR)	MILP Subproblem of Parametric Algorithm
Number of binary 0–1 variables	$\|J\|$	$\|J\|$	$\|J\|$
Number of continuous variables	$\|I\|$	$\|I\| + \|J\| + 1$	$\|I\|$
Number of constraints	$\|K\|$	$3 \cdot \|J\| + \|K\| + 1$	$\|K\|$
Number of iterations required	–	one	At least one

$|I|$, $|J|$, $|K|$ are the numbers of continuous variables, binary variables and constraints of the original MILFP problem, respectively.

the number of discrete variables, $|J|$ is not significantly large. In practice, there is a clear trade-off between the number of MILPs to solve and their size.

Note that the differences between the reformulated MILP problem (PR) and the MILP subproblems of the parametric algorithm reside not only on the number of equations and variables. These problems also have a different objective function and a different linear relaxation, which may also have nontrivial impacts on the solution process. For clarification, in this work, we consider the proposed reformulation–linearization method and the parametric algorithm as two equally-good alternatives for solving general MILFP problems, rather than saying the proposed method is the best in all instances. Practitioners are encouraged to try both approaches when dealing with MILFP problems. In the next two sections, we present computational results for solving large-scale MILFP problems with the proposed reformulation–linearization method, parametric algorithm and various MINLP and global optimization methods.

Computational Results

In this section, we present computational experiments for 25 large-scale, randomly generated MILFP problems to demonstrate the efficiency of the proposed reformulation–linearization method. The first 21 instances are MILFP problems involving both discrete variables and continuous variables, while the last four instances are integer linear fractional programming (ILFP) problems with only integer variables. Unless otherwise stated, all the computational experiments are carried out on a PC with Intel® Core$^{\text{TM}}$ i5–2400 CPU @ 3.10 GHz and 8.00-GB RAM. All models and solution procedures are coded in GAMS 23.9.1.[37] The MILP problems in the proposed reformulation–linearization method and parametric algorithm are solved using CPLEX 12. The MINLP solvers used in the computational experiments are SBB (standard branch-and-bound algorithm), DICOPT (Outer-Approximation algorithm), and the global optimizer BARON 11.[36] The absolute optimality tolerance is set to be 10^{-3}. Four cores of the PC are available for the solvers and the parallel mode of CPLEX 12 is allowed. Note that the one-thread CPLEX 12 computational results are also summarized in Appendix.

We consider the MILFP problem (TP) presented below.

$$(\mathbf{TP})\ \max\ \frac{A_0+\sum_{i\in I}A1_ix_i+\sum_{j\in J}A2_jy_j}{B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j} \tag{24}$$

$$\text{s.t.}\quad \sum_{i\in I}C1_{ik}x_i+\sum_{j\in J}C2_{jk}y_j\le D_k,\ \ \forall k\in K \tag{25}$$

$$B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j\ge 0.001 \tag{26}$$

$$x^L\le x_i\le x^U,\ \ \forall i\in I\quad \text{and}\quad y_j\in\{0,1\},\ \ \forall j\in J \tag{27}$$

In this testing problem (TP), all the constraints are linear and the objective function is the ratio of two linear functions. This problem consists of $|I|$ continuous variables, $|J|$ binary variables, and $|K|+1$ constraints. The values of $|I|$, $|J|$, and $|K|$ range from 100 to 3000. The upper and lower bounds of the continuous variables x_i are all set to 1 and 0.1, respectively. The other parameters are generated randomly using uniform distribution. A_0 is generated uniformly on $U[-10,10]$. $A1_i$ is generated uniformly on $U[0,10]$. $A2_j$ is generated uniformly on $U[0,10]$. B_0 is generated uniformly on $U[-10,10]$. $B1_i$ is generated uniformly on $U[0,10]$. $B2_j$ is generated uniformly on $U[0,10]$. $C1_{ik}$ is generated uniformly on $U[-10,10]$. $C2_{jk}$ is generated uniformly on $U[-20,15]$. D_k is generated uniformly on $U[-15,20]$.

By using the proposed reformulation–linearization method, the model can be transformed into its equivalent MILP problem (TPR) given as follows.

$$(\mathbf{TPR})\ \max\ A_0\cdot u+\sum_{i\in I}A1_iz_i+\sum_{j\in J}A2_j\cdot w_j \tag{28}$$

$$\text{s.t.}\quad \sum_{i\in I}C1_{ik}\cdot z_i+\sum_{j\in J}C2_{jk}\cdot w_j\le D_k\cdot u,\ \ \forall k\in K \tag{29}$$

$$B_0\cdot u+\sum_{i\in I}B1_i\cdot z_i+\sum_{j\in J}B2_j\cdot w_j=1 \tag{30}$$

$$u\cdot x^L\le z_i\le u\cdot x^U,\ \ \forall i\in I \tag{31}$$

$$w_j\le u,\ \ \forall j\in J \tag{32}$$

$$w_j\le M\cdot y_j,\ \ \forall j\in J \tag{33}$$

$$w_j\ge u-M\cdot(1-y_j),\ \ j\in J \tag{34}$$

$$\begin{aligned} &u\ge 0;\ \ z_i\ge 0,\ \ \forall i\in I\\ &w_j\ge 0,\quad y_j\in\{0,1\},\quad \forall j\in J\end{aligned} \tag{35}$$

where $u=1/\left(B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j\right)$, $z_i=u\cdot x_i$ and w_j are the auxiliary variables for the Glover's linearization scheme.

As a side-by-side comparison, we also solve the original problem (TP) using the parametric algorithm, in which a sequence of MILP subproblems are calculated. The model formulation of the MILP subproblem is presented as follows.

$$\begin{aligned}(\mathbf{TPD})\ \max\ &\left(A_0+\sum_{i\in I}A1_ix_i+\sum_{j\in J}A2_jy_j\right)\\ &-q\left(B_0+\sum_{i\in I}B1_ix_i+\sum_{j\in J}B2_jy_j\right)\\ &\text{s.t. constraints (25)-(27)}\end{aligned} \tag{36}$$

where q is a parameter. Unless otherwise stated, the initial value of q is set to 0. Comparing the objective functions of (TPD) and (TP), we can see that (36) is equal to the numerator of (24) minus its denominator times the parameter q. Other than the objective function, all the constraints are exactly the same. As for the comparisons between (TPR) and (TPD), they are both MILP problems, but the problem size of (TPR) is larger than that of (TPD). This is due to the introduction of the positive continuous variable u, a set of continuous variables w_j, and auxiliary constraints (32)–(34) for the Glover's linearization scheme. Therefore, the computational time for solving (TPR) could be longer than that associated with subproblem (TPD). We note that the parametric algorithm relies on an iterative procedure that solves a sequence of MILP subporblems (TPD), whereas the reformulation–linearization method solves only one MILP model (TPR). There is therefore a trade-off between solving a series of MILPs smaller in size and calculating a bigger MILP only once.

We next apply the reformulation–linearization method and the parametric algorithm to solve these problems. We also solve them with the general-purpose MINLP solvers SBB and DICOPT, as well as the global optimizer BARON 11. The computational results are given in Table 2.

Table 2. Computational Results for Random MILFP Problems with Five Solution Methods

No.	Continuous Variables	Discrete Variables	Constraints	Reformulation-Linearization		Parametric Algorithm			SBB		DICOPT		BARON 11	
	$\|I\|$	$\|J\|$	$\|K\|+1$	Objective	CPU Seconds	Iterations	Objective	CPU Seconds	Objective	CPU Seconds	objective	CPU Seconds	Objective	CPU Seconds
1	100	100	101	3.394	1	7	3.394	1	3.394	1	3.394	1	3.394	1
2	500	100	101	2.391	1	7	2.391	2	2.391	5	2.391	2	2.391	23
3	100	500	101	6.094	1	8	6.094	2	6.094	6	6.094	1	6.094	3
4	100	100	501	2.644	307	7	2.644	304	2.617–2.680	3600[a]	–	–[b]	2.328–3.036	3600[a]
5	500	500	101	3.546	1	7	3.546	2	3.546	1	3.546	1	3.546	6
6	500	500	501	3.507	8	7	3.507	15	3.507	122	3.507	63	3.507	156
7	1000	500	501	2.942	10	7	2.942	15	2.942	114	2.942	116	2.942	369
8	500	1000	501	4.114	4	7	4.114	16	4.114	73	4.114	76	4.114	191
9	500	500	1001	3.484	38	7	3.484	37	3.484	726	3.484	339	3.484	852
10	1000	1000	501	3.512	12	7	3.512	37	3.512	167	3.512	183	3.512	442
11	1000	1000	1001	3.512	25	7	3.512	89	3.512	825	3.512	850	3.512	1893
12	2000	1000	1001	2.956	32	7	2.956	37	2.956	1786	2.956	1875	2.956–14.312	3600[a]
13	1000	2000	1001	4.667	28	7	4.667	71	4.667	1310	4.667	1310	4.667	2030.74
14	1000	1000	2001	3.510	56	8	3.510	172	–	3600[c]	–	3600[c]	1.895–19.836	3600[a]
15	2000	2000	1001	3.469	143	7	3.469	310	3.469	2537	3.469	2538	2.674–19.297	3600[a]
16	2000	2000	2001	3.469	292	7	3.469	449	–	3600[c]	–	3600[c]	2.152–19.297	3600[a]
17	3000	2000	2001	3.199	119	6	3.199	97	–	3600[c]	–	3600[c]	–	3600[c]
18	2000	3000	2001	3.970	314	7	3.970	509	–	3600[c]	–	3600[c]	–	3600[c]
19	2000	2000	3001	3.469	326	7	3.469	670	–	3600[c]	–	3600[c]	–	3600[c]
20	3000	3000	2001	3.525	686	7	3.525	606	–	3600[c]	–	3600[c]	–	3600[c]
21	3000	3000	3001	3.525	1302	7	3.525	2196	–	3600[c]	–	3600[c]	–	3600[c]
22	0	500	250	18.250	111	8	18.250	56	–	3600[c]	18.25	1011	15.643–20.636	3600[a]
23	0	1000	500	64.571	2	7	64.571	8	64.571	524	64.571	21	64.571	41
24	0	2000	1000	114.714	8	7	114.714	26	–	3600[c]	114.714	204	114.714	423
25	0	3000	1500	174.286	6	7	174.286	60	174.286	948	174.286	939	174.286	1096

[a]Suboptimal solutions (lower and upper bounds) obtained after 1 h (3600 s).
[b]Solver failure encountered.
[c]No feasible solutions returned after 1 h (3600 s).

As for the first 21 MILFP problems, the number of discrete variables, continuous variables, and constraints range from 100 each to 3000 each. Note that the effective size of the reformulated MILP problem can be easily calculated via Table 1. Generally, for all solution procedures, as the problem size grows, the computational time increases. However, as an exception, Instance 4 costs far more computational time than its adjacent instances, which indicates that the computational efforts required may not only depend on the problems size but also on the input data structure. For all the instances, the proposed reformulation–linearization method is able to obtain the global optimum in 30 min. In particular, it globally optimizes Instances 1–14 within 1 min. Regarding the parametric algorithm, although in most cases it requires more computational time and iterations, it is faster than the proposed reformulation–linearization method for Instances 4, 9, 17, and 20. However, it is still fair to conclude that the proposed reformulation–linearization method performs better than parametric algorithm for this TP.

As for the three general purpose MINLP solvers, they perform worse than the two tailored algorithms discussed above in solving these MILFP problems. The computational performances of SBB and DICOPT are similar. For relatively small-size problems (Instances 1–3 and 5), both MINLP solvers perform similarly to the reformulation–linearization method and the parametric algorithm. For relatively medium-size problems (Instances 5–13 and 15), they are able to obtain global optimums within 1 h, but are slower than the reformulation–linearization methods and the parametric algorithm. However, for relatively large-size problems (Instances 16–21) and some difficult problems (Instances 4 and 14), they fail to return the global optimal solutions. Although in all instances, DICOPT returns the global optimal solutions, it cannot guarantee the global optimality of the solution as discussed in the second section. In general, the global optimizer BARON 11 is slower than the local MINLP solvers, SBB, and DICOPT. It obtains global optimums for Instances 1–3, 5–11, and 13, but it returns suboptimal solutions or fails to converge for Instances 4, 12, 14–16 within 1 h. However, for relatively large-size problems (Instances 17–21), BARON 11 is not able to return any feasible solutions within 1 h. Overall, for the first 21 MILFP problems, the tailored reformulation–linearization method and the parametric algorithm are shown to be more efficient than the general-purpose MINLP solvers, and in most cases the reformulation–linearization method exhibits better performance.

We further solve four pure ILFP problems which do not include any continuous variable, that is, $|I|=0$. These problems are denoted as Instances 22–25. The number of binary variables range from 500 to 3000, and the number of constraints range from 250 to 1500. The proposed reformulation–linearization method is able to solve Instances 22–25 in 2 min. However, the computational time is not proportional to the problem size. Instance 22, which is the smallest in size, leads to the largest CPU times. Parametric algorithm exhibits comparable performances as the reformulation–linearization method. For instance in 22, the parametric algorithm is faster than the reformulation–linearization method, but for Instances 23–25 the parametric algorithm is slower. DICOPT is the fastest among the three MINLP solvers and returns global optimums for all Instances 22–25 in 1 h, though it is not so efficient as the reformulation–linearization method and the parametric algorithm. BARON 11 obtains global optimal solutions for Instances 23–25, and returns a suboptimal solution for Instance 22. SBB returns global optimums for Instances 23 and 25, but fails to find any feasible solutions for Instances 22 and 24. In conclusion, the proposed reformulation–linearization method is also robust and efficient in solving ILFP problems compared to general-purpose MINLP solvers, and it is comparable with the tailored parametric algorithm.

Applications in Batch Process Scheduling

In this section, we present two case studies that address scheduling problems of multipurpose batch plants. The first example studies the cyclic production scheduling, and the second example optimizes the productivity over the scheduling makespan. Both problems are formulated as MILFP problems based on the continuous–time formulation.[11,49]

Case study 1: Cyclic scheduling

In this case study, we consider the cyclic scheduling problem[22,50,51] which is a variant of the most studied scheduling problem in the process systems engineering literature originally proposed by Kondili et al.[52] This problem deals with the scheduling of a multipurpose pharmaceutical batch plant. The resource-task network (RTN)[51] of this batch process is shown in Figure 2. It produces two products from three raw materials through a reaction-separation network. There are three intermediate products and three equipment units available. The model formulation considered in this example was proposed by Castro et al.[23] based on a uniform time-grid continuous-time representation. Because You et al. studied the performances of the parametric algorithm and some general-purpose MINLP solvers on this problem in their paper,[6] we use the identical input data here. Therefore, the optimal solutions are exactly the same as those reported in their paper, and the differences in computational performances for the corresponding solution methods are mainly due to the use of newer versions of the solvers and the improvements in hardware.

In the RTN representation, the set of tasks is denoted as $i \in I$, and the chemicals and processing units are treated as generic entities called resources $r \in R$. By using the wrap-around operator $\Omega(\cdot)$,

$$\Omega(t)=\begin{cases} t, & 1 \le t \le |T| \\ t+|T|, & t<1 \end{cases} \tag{37}$$

the cyclic scheduling formulation is given in a very compact form, (38)–(44). In addition, as mentioned in the work by Castro et al.,[23] an anchoring constraint can be introduced to reduce the solution degeneracy arising from cyclic scheduling permutations.

Therefore, the problem is formulated as an MILFP problem, which is given below and denoted as (C1-P).

$$(\mathbf{C1-P})\ \max \left(\frac{\sum_{r \in R^{FP}} pr_r Amt_r}{H} \right) \tag{38}$$

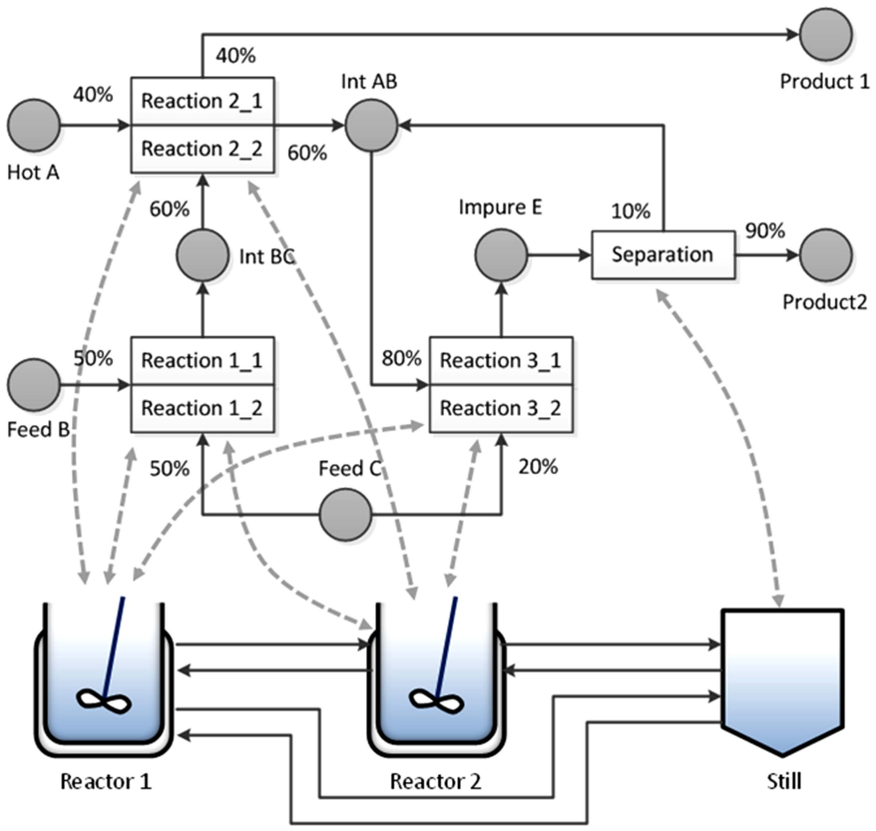

Figure 2. RTN representation of Case Study 1.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com]

s.t.

$$R_{r,t}=R_{r,\Omega(t-1)}+\left(Amt_{r|r\in R^{RM}}-Amt_{r|r\in R^{FP}}\right)_{|t=1}+\sum_{i\in I}\left[\sum_{\substack{t'\in T\\(t<t'\le t+\Delta t)\vee(t'\le t+\Delta t-|T|)}}\left(fcon_{r,i}N_{i,t,t'}+vcon_{r,i}B_{i,t,t'}\right)+\sum_{\substack{t'\in T\\(t-\Delta t\le t'<t)\vee(t'\ge t-\Delta t+|T|)}}\left(fpro_{r,i}N_{i,t',t}+vcon_{r,i}B_{i,t',t}\right)\right],\quad \forall r\in R, t\in T \tag{39}$$

$$R_{r,|T|}=ro_r+\sum_{i\in I}\sum_{t\in T}\left(\sum_{\substack{t'\in T\\(t<t'\le t+\Delta t)\vee(t'\le t+\Delta t-|T|)}} fcon_{r,i}N_{i,t,t'}+\sum_{\substack{t'\in T\\t<t'\le t+\Delta t}} fpro_{r,i}N_{i,t,t'}\right),\quad \forall r\in R^{EQ} \tag{40}$$

$$b_i^{\min}N_{i,t,t'}\le B_{i,t,t'}\le b_i^{\max}N_{i,t,t'},\quad \forall i\in I, t\in T, t'\in T, (t<t'\le t+\Delta t)\vee(t'\le t+\Delta t-|T|) \tag{41}$$

$$T_{t'}-T_t\ge ft_iN_{i,t,t'}+vt_iB_{i,t,t'},\quad \forall i\in I, t\in T, t'\in T, t<t'\le t+\Delta t \tag{42}$$

$$T_t-T_{t'}\ge H-ft_iN_{i,t,t'}+vt_iB_{i,t,t'},\quad \forall i\in I, t\in T, t'\in T, t'\le t+\Delta t-|T| \tag{43}$$

$$h^{\min}\le H\le h^{\max};\quad T_1=0;\quad 0\le T_t\le h^{\max};\quad N_{i,t,t'}\in\{0,1\};\quad B_{i,t,t'},R_{r,t},Amt_r\ge 0 \tag{44}$$

As shown in constraint (38), the objective is to maximize the revenue (the numerator) while minimizing the cycle time (the denominator), or in other words, to maximize the hourly revenue. Constraint (39) indicates the resource balance over the cycle and constraint (40) accounts for the start-up procedure. A set of time points is represented by $t\in T$, and Δt restricts the number of time points that a task can span. The binary variable $N_{i,t,t'}$ equals to 1 if task i starts at event point t and ends at/before time point t', while the continuous variable $B_{i,t,t'}$ denotes the corresponding batch size. The latter should lie between its upper bound ($b_i^{\max}$) and lower bound ($b_i^{\max}$) as given by constraint (41). In this formulation, the duration of a task consists of a fixed part (ft_i) related to the occurrence of the task and a variable part (vt_i) proportional to the batch size being processed. The continuous variable T_t denotes the time of event point t relative to the start of the cycle. Constraints (42) and (43) model the timing relations between the start and finishing of a task. At last, constraints (44) are the upper- and lower-bound constraints and the non-negative and integrity constraints.

By using the proposed reformulation–linearization method, the original MILFP problem (C1-P) can be transformed into the following equivalent MILP problem (C1-PR). For the simplicity of presentation, the reformulated variables retain the name of the original variables but add a letter G in the front.

$$(\mathbf{C1-PR})\quad \max\sum_{r\in R^{FP}} pr_rGAmt_r \tag{45}$$

s.t.

$$GR_{r,t}=GR_{r,\Omega(t-1)}+\left(GAmt_{r|r\in R^{RM}}-GAmt_{r|r\in R^{FP}}\right)_{|t=1}$$

$$+\sum_{i\in I}\left[\begin{array}{l}\displaystyle\sum_{\substack{t'\in T\\(t<t'\le t+\Delta t)\vee(t'\le t+\Delta t-|T|)}}\left(fcon_{r,i}GN_{i,t,t'}+vcon_{r,i}GB_{i,t,t'}\right)\\ \displaystyle+\sum_{\substack{t'\in T\\(t-\Delta t\le t'<t)\vee(t'\ge t-\Delta t+|T|)}}\left(fpro_{r,i}GN_{i,t',t}+vpro_{r,i}GB_{i,t',t}\right)\end{array}\right],\ \forall \mathrm{r}\in R,t\in T \tag{46}$$

$$GR_{r,|T|}=ro_r\cdot G$$
$$+\sum_{i\in I}\sum_{t\in T}\left(\sum_{\substack{t'\in T\\(t<t'\le t+\Delta t)\vee(t'\le t+\Delta t-|T|)}}fcon_{r,i}GN_{i,t,t'}+\sum_{\substack{t'\in T\\t<t'\le t+\Delta t}}fpro_{r,i}GN_{i,t,t'}\right),\ \forall \mathrm{r}\in R^{EQ} \tag{47}$$

$$b_i^{\min}GN_{i,t,t'}\le GB_{i,t,t'}\le b_i^{\max}GN_{i,t,t'},$$
$$\forall i\in I,t\in T,t'\in T,(t<t'\le t+\Delta t)\vee(t'\le t+\Delta t-|T|) \tag{48}$$

$$GT_{t'}-GT_t\ge ft_iGN_{i,t,t'}+vt_iGB_{i,t,t'},$$
$$\forall i\in I,t\in T,t'\in T,t<t'\le t+\Delta t \tag{49}$$

$$GT_t-GT_{t'}\ge GH-ft_iGN_{i,t,t'}+vt_iGB_{i,t,t'},$$
$$\forall i\in I,t\in T,t'\in T,t'\le t+\Delta t-|T| \tag{50}$$

$$GH=1 \tag{51}$$

$$GN_{i,t,t'}\le G,\ \forall i\in I,t\in T,t'\in T,$$
$$(t<t'\le t+\Delta t)\vee(t'\le t+\Delta t-|T|) \tag{52}$$

$$GN_{i,t,t'}\le g^{\max}\cdot N_{i,t,t'},\ \forall i\in I,t\in T,t'\in T,$$
$$(t<t'\le t+\Delta t)\vee(t'\le t+\Delta t-|T|) \tag{53}$$

$$GN_{i,t,t'}\ge G-g^{\max}\left(1-N_{i,t,t'}\right),$$
$$\forall i\in I,t\in T,t'\in T,(t<t'\le t+\Delta t)\vee(t'\le t+\Delta t-|T|) \tag{54}$$

$$h^{\min}\cdot G\le GH\le h^{\max}\cdot G;\quad GT_1=0;\quad 0\le GT_t\le h^{\max}\cdot G;$$
$$N_{i,t,t'}\in\{0,1\};\quad GB_{i,t,t'},GR_{r,t},GAmt_r,GH,G,GN_{i,t,t'}\ge 0 \tag{55}$$

Compared to the original formulation (C1-P), the objective function (45) becomes a linear function rather than a fractional term. Constraints (46)-(50) and constraint (55) correspond to constraints (39)-(44) in the original formulation, while constraints (51) to (54) are additional. Constraint (51) is crucial to maintain its equivalence with (C1-P). Constraints (52)-(54) relate to the linearization of bilinear terms. Also, G and $GN_{i,t,t'}$ are additional continuous variables. The former represents the reciprocal of cycle time H and the latter are the auxiliary variables for the Glover's linearization scheme. All the other continuous variables have a one-to-one correlation with the variables in (C1-P), that is $R_{r,t}=GR_{r,t}/G$.

The parametric algorithm involves solving a sequence of MILP problem (C1-PD). The difference from the original problem (C1-P) is only in the objective function, while all the constraints remain the same.

$$(\mathbf{C1-PD})\quad \max\sum_{r\in R^{FP}}pr_rAmt_r-q\cdot H$$
$$\text{s.t.}\quad \text{constraints}\quad (39)\text{-}(44) \tag{56}$$

where q is a parameter, which would be updated iteratively until reaching the optimal objective value for the original MILFP problem (C1-P).

As known, the solution returned using the time grid formulation is dependent on the number of event points $|T|$, as well as the number of intervals that a task can span Δt. As pointed out by Wu and Ierapetritou,[53] the feasible range of the cycle time $H\in\left[H^{\min},H^{\max}\right]$ would also affect the solution. As the goal of this article is to compare the computational performances of various algorithms for solving MILFP problems, we solve five instances with different patterns of these four parameters using the reformulation–linearization method, the parametric algorithm, and the general-purpose MINLP solvers, DICOPT, and SBB, and the global optimizer BARON 11. The results are given in Table 3.

As can be seen, the problem size grows as the number of event points $|T|$ and the number of intervals that a task can span Δt increase. The proposed reformulation–linearization method obtains the global optimal solution in 7 s for Instance 1, whereas it spends about 18 min to return the global optimum for Instance 5. The parametric algorithm also returns global optimal solutions within the computational time limit for all the five instances. For solving Instance 1, the parametric algorithm is slightly slower than the reformulation–linearization method, but for Instances 2–4, the parametric algorithm tends to be faster. BARON 11 obtains global optimums for Instance 2–4 within 1 h.

Table 3. Results for Case Study 1 with Five Solution Methods

No.	Number of event points	Step	H^{min}	H^{max}	Discrete Variables	Continuous Variables	Constraints	Objective	Iterations	Cycle Time	CPU Seconds	Solvers/Algorithms
					105	276	859	32.757		62.801	7	Reformulation–Linearization
					105	172	375	32.757	3	62.801	9	Parametric
1	5	3	40	70	105	172	375	32.757–40.004		62.801	3600[a]	SBB
					105	172	375	26.514		40	14	DICOPT
					105	172	375	32.757–40.780		62.801	3600[a]	BARON 11
					105	276	859	32.605		97.585	8	Reformulation–Linearization
					105	172	375	32.605	3	97.585	8	Parametric
2	5	3	70	100	105	172	375	32.605–35.301		97.585	3600[a]	SBB
					105	172	375	30.302		70	21	DICOPT
					105	172	375	32.605		97.585	1101	BARON 11
					126	330	1029	33.396		105.677	55	Reformulation–Linearization
					126	205	449	33.396	3	105.677	27	Parametric
3	6	3	100	140	126	205	449	33.396–37.921		105.677	3600[a]	SBB
					126	205	449	33.396		105.677	101	DICOPT
					126	205	449	33.396		105.677	2960	BARON 11
					168	414	1323	33.396		105.677	93	Reformulation–Linearization
					168	247	575	33.396	3	105.677	33	Parametric
4	6	4	100	140	168	247	575	33.396-38.783		105.677	3600[a]	SBB
					168	247	575	33.396		105.677	114	DICOPT
					168	247	575	33.396		105.677	2659	BARON 11
					196	482	1542	34.108		103.47	1047	Reformulation–Linearizaton
					196	287	670	34.108	3	103.47	667	Parametric
5	7	4	100	140	196	287	670	32.320–43.579		100	3600[a]	SBB
					196	287	670	34.108		103.47	2284	DICOPT
					196	287	670	34.108–41.912		103.47	3600[a]	BARON 11

[a]Suboptimal solutions (lower and upper bounds) obtained after 1 h (3600 s).

However, for Instances 1 and 5, BARON 11 finds the global optimal solutions (32.757 and 34.108, respectively), but fails to prove its global optimality within 1 h. SBB finds global optimums for Instances 1–4, but fails to prove their optimality within 1 h. For Instance 5, SBB returns a suboptimal solution (32.320) rather than the global optimal one (34.108) after 1 h. DICOPT is the fastest among the three general-purpose MINLP solvers. For Instances 3–5, DICOPT returns global optimal solutions in 1 h, but for Instances 1 and 2 only suboptimal solutions are obtained after 1 h. The gaps relative to the global optimums are 19.1 and 7.1%, respectively. Overall, the reformulation–linearization method and the parametric algorithm present better computational performances than the general-purpose MINLP solvers for the global optimization of these cyclic scheduling problems.

Case study 2: Productivity

In this case study, we consider a scheduling problem originally coming from The Dow Chemical Company.[54,55] The state-task network (STN) representation of this process is given in Figure 3. Similar to the previous example, it represents a multipurpose batch plant with a complicated network structure. This process involves four products, four raw materials, and six intermediates. Both batch mixing and splitting are involved in the process. The decisions include the assignment of eight tasks to five equipment units, as well as the batch sizing. Note that in this problem, changeover times are not considered.

The objective of this problem is to maximize the productivity, which is defined as the total profit divided by the corresponding makespan.[56] A minimum demand of each product is required to be met, and the total time horizon is restricted to be 2000 min. This problem was addressed recently by Chu et al.[55] but the two objectives (minimizing the operational cost or makespan) are optimized separately in their paper, while productivity possesses the features of both objectives. Therefore, the results and optimal solutions may be different due to the difference in the objective function, although we use the identical input data as those reported in their paper.

In this example, we adopt the continuous-time formulation using global time points proposed by Maravelias and Grossmann.[24] The resulting MILFP problem is presented below, and denoted as problem (C2-P). Notations can be found in the Notation section in the end of this article.

$$(\mathbf{C2\!-\!P})\quad \max\left[\frac{\sum_{s\in P} p_s S_{s|t|} - \sum_i \sum_t \left(Ws_{it}f_{ij} + Bs_{it}b_{ij}\right)}{T_{|t|}}\right] \tag{57}$$

$$\text{s.t.} \sum_t Ws_{it} = \sum_t Wf_{it}, \quad \forall i \tag{58}$$

$$\sum_{i\in I_j} Ws_{it} \le 1, \quad \forall j,t \tag{59}$$

$$\sum_{i\in I_j} Wf_{it} \le 1, \quad \forall j,t \tag{60}$$

$$\sum_{i\in I_j} \sum_{t'<t} (Ws_{it'} - Wf_{it'}) \le 1, \quad \forall j,t \tag{61}$$

$$Td_{it} = Ws_{it}pt_i^F + Bs_{it}pt_i^V, \quad \forall i,t \tag{62}$$

$$Tf_{it} \ge Ts_{it} + Td_{it} - ht(1 - Ws_{it}), \quad \forall i,t \tag{63}$$

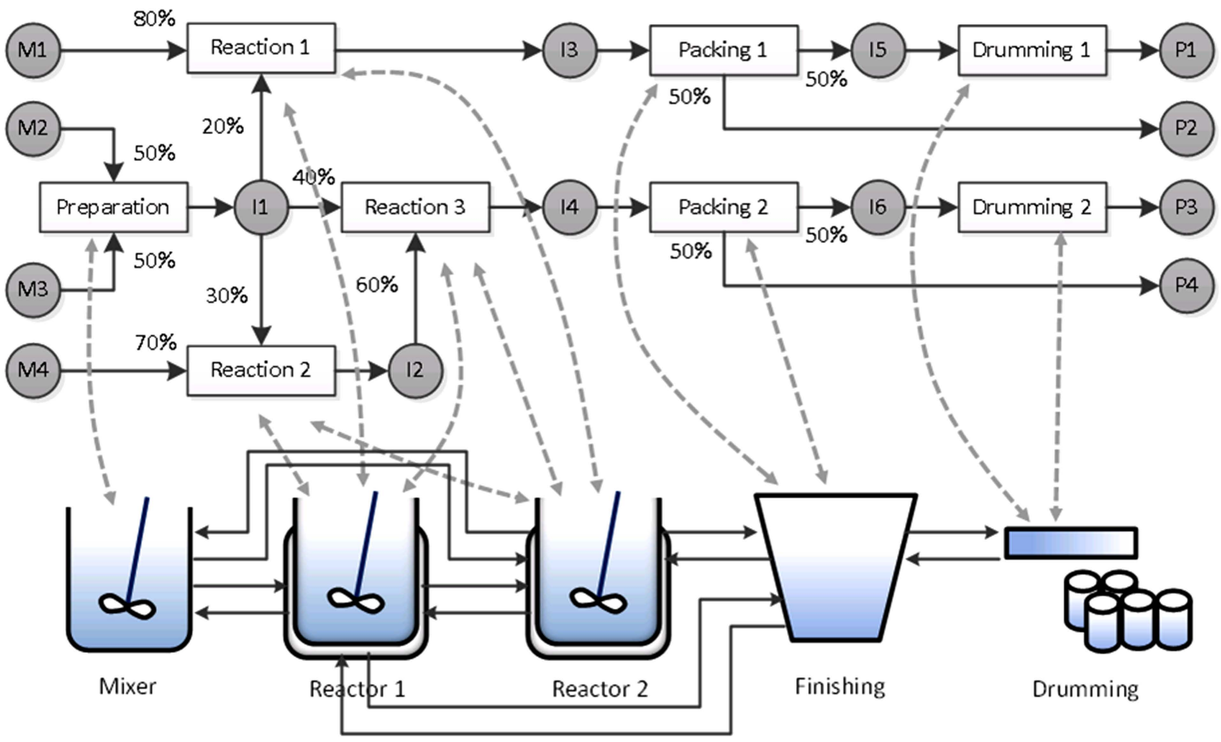

Figure 3. STN representation of Case Study 2.

[Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com]

$$Tf_{it} \le Ts_{it} + Td_{it} + ht(1 - Ws_{it}), \quad \forall i,t \tag{64}$$

$$Tf_{it} - Tf_{i(t-1)} \le Ws_{it} \cdot ht, \quad \forall i,t \tag{65}$$

$$Tf_{it} - Tf_{i(t-1)} \ge Td_{it}, \quad \forall i,t \tag{66}$$

$$Tf_{i(t-1)} \le T_t + ht(1 - Wf_{it}), \quad \forall i,t \tag{67}$$

$$Ts_{it} = T_t, \quad \forall i,t \tag{68}$$

$$T_t \ge T_{(t-1)}, \quad \forall t > 1 \tag{69}$$

$$T_1 = 0 \tag{70}$$

$$T_{|t|} \le ht \tag{71}$$

$$b_i^{\min} Ws_{it} \le Bs_{it} \le b_i^{\max} Ws_{it}, \quad \forall i,t \tag{72}$$

$$b_i^{\min} Wf_{it} \le Bf_{it} \le b_i^{\max} Wf_{it}, \quad \forall i,t \tag{73}$$

$$b_i^{\min}\left(\sum_{t'<t} Ws_{it'} - \sum_{t'<t} Wf_{it'}\right) \le Bp_{it} \le b_i^{\max}\left(\sum_{t'<t} Ws_{it'} - \sum_{t'<t} Wf_{it'}\right), \quad \forall i,t \tag{74}$$

$$Bs_{i(t-1)} + Bp_{i(t-1)} = Bp_{it} + Bf_{it}, \quad \forall i, t > 1 \tag{75}$$

$$S_{s|t|} \ge d_s, \quad \forall s \in P \tag{76}$$

$$B_{ist}^I = \rho_{is}^c Bs_{it}, \quad \forall i,s,t \tag{77}$$

$$B_{ist}^O = \rho_{is}^p Bf_{it}, \quad \forall i,s,t \tag{78}$$

$$B_{ist}^I \le b_i^{\max} \rho_{is}^c Ws_{it}, \quad \forall i,s,t \tag{79}$$

$$B_{ist}^O \le b_i^{\max} \rho_{is}^p Wf_{it}, \quad \forall i,s,t \tag{80}$$

$$S_{st} = S_{s(t-1)} + \sum_{i \in O_s} B_{ist}^O - \sum_{i \in I_s} B_{ist}^I, \quad \forall s,t \tag{81}$$

$$S_{st} \le s_s^{\max}, \quad \forall s,t \tag{82}$$

$$Bf_{it}, Bp_{it}, Bs_{it}, B_{ist}^I, B_{ist}^O, S_{st}, T_t, Td_{it}, Tf_{it}, Ts_{it} \ge 0 \tag{83}$$

$$Wf_{it}, Ws_{it} \in \{0,1\} \tag{84}$$

The objective function is given in (57) representing the productivity, where the numerator is the total profit and the denominator is the makespan. The inventory level of a product (S_{st}) at the end of the makespan equals to its sales amount. Given the market price of each product (p_s), the total revenue equals to the total income from selling all the products. The operational cost consists of a fixed part and a variable part, which is proportional to the batch size. The makespan is simply the absolute time at the last time point relative to the beginning of the production. The assignment constraints are given by constraints (58)–(61), where the binary variables Ws_{it} and Wf_{it} identify the start and finishing of a task ($i \in I$) related to the time points $t \in T$. The timing constraints for the start, duration, and finishing of each task are given by constraints (62)–(71). The capacity constraints (72)–(75) specify the upper and lower limits of the batch size for each task, and establish its correlation with the global time points. Constraint (76) states that the minimum demand must be satisfied. The material balance constraints given by (77)–(82) model the storage level of each state at all the time points. At last, constraints (83) and (84) are the nonnegative and integrity constraints.

By applying the reformulation–linearization method, the original MILFP problem (C1-P) can be reformulated into the following equivalent MILP problem. Similar to Case Study 1, the reformulated variables retain the name of the original variables with an additional letter G in the front of each symbol for the simplicity of presentation.

$$(\mathbf{C2-PR}) \quad \max \; p_s GS_{s|t|} - \sum_i \sum_t \left(GWs_{it} f_{ij} + GBs_{it} b_{ij}\right) \tag{85}$$

$$\text{s.t.} \quad \sum_t GWs_{it} = \sum_t GWf_{it}, \quad \forall i \tag{86}$$

$$\sum_{i \in I_j} GWs_{it} \le G, \quad \forall j,t \tag{87}$$

$$\sum_{i \in I_j} GWf_{it} \le G, \quad \forall j,t \tag{88}$$

$$\sum_{i \in I_j} \sum_{t'<t} \left(GWs_{it'} - GWf_{it'}\right) \le G, \quad \forall j,t \tag{89}$$

$$GTd_{it} = GWs_{it} pt_i^F + GBs_{it} pt_i^V, \quad \forall i,t \tag{90}$$

$$GTf_{it} \ge GTs_{it} + GTd_{it} - ht(G - GWs_{it}), \quad \forall i,t \tag{91}$$

$$GTf_{it} \le GTs_{it} + GTd_{it} + ht(G - GWs_{it}), \quad \forall i,t \tag{92}$$

$$GTf_{it} - GTf_{i(t-1)} \le GWs_{it} \cdot ht, \quad \forall i,t \tag{93}$$

$$GTf_{it} - GTf_{i(t-1)} \ge GTd_{it}, \quad \forall i,t \tag{94}$$

$$GTf_{i(t-1)} \le GT_t + ht(G - GWf_{it}), \quad \forall i,t \tag{95}$$

$$GTs_{it} = GT_t, \quad \forall i,t \tag{96}$$

$$GT_t \ge GT_{(t-1)}, \quad \forall t > 1 \tag{97}$$

$$GT_1 = 0 \tag{98}$$

$$GT_{|t|} \le G \cdot ht \tag{99}$$

$$b_i^{\min} GWs_{it} \le GBs_{it} \le b_i^{\max} GWs_{it}, \quad \forall i,t \tag{100}$$

$$b_i^{\min} GWf_{it} \le GBf_{it} \le b_i^{\max} GWf_{it}, \quad \forall i,t \tag{101}$$

$$b_i^{\min}\left(\sum_{t'<t} GWs_{it'} - G\sum_{t'<t} Wf_{it'}\right) \le GBp_{it} \le b_i^{\max}\left(\sum_{t'<t} GWs_{it'} - \sum_{t'<t} GWf_{it'}\right), \quad \forall i,t \tag{102}$$

$$GBs_{i(t-1)} + GBp_{i(t-1)} = GBp_{it} + GBf_{it}, \quad \forall i, t > 1 \tag{103}$$

$$GS_{s|t|} \ge G \cdot d_s, \quad \forall s \tag{104}$$

$$GB_{ist}^I = \rho_{is}^c GBs_{it}, \quad \forall i,s,t \tag{105}$$

$$GB_{ist}^O = \rho_{is}^p GBf_{it}, \quad \forall i,s,t \tag{106}$$

$$GB_{ist}^I \le b_i^{\max} \rho_{is}^c GWs_{it}, \quad \forall i,s,t \tag{107}$$

$$GB_{ist}^O \le b_i^{\max} \rho_{is}^p GWf_{it}, \quad \forall i,s,t \tag{108}$$

$$GS_{st} = GS_{s(t-1)} + \sum_{i \in O_s} GB_{ist}^O - \sum_{i \in I_s} GB_{ist}^I, \quad \forall s,t \tag{109}$$

$$GS_{st} \le G \cdot s_s^{\max}, \quad \forall s,t \tag{110}$$

$$GT_{|t|} = 1 \tag{111}$$

$$GWs_{it} \le G, \quad \forall i,t \tag{112}$$

$$GWs_{it} \le g^{\max} Ws_{it}, \quad \forall i,t \tag{113}$$

$$GWs_{it} \ge G - g^{\max}(1 - Ws_{it}), \quad \forall i,t \tag{114}$$

$$GWf_{it} \le G, \quad \forall i,t \tag{115}$$

$$GWf_{it} \le g^{\max} Wf_{it}, \quad \forall i,t \tag{116}$$

$$GWf_{it} \ge G - g^{\max}(1 - Wf_{it}), \quad \forall i,t \tag{117}$$

$$GBf_{it}, GBp_{it}, GBs_{it}, GB^{I}_{ist}, GB^{O}_{ist}, GS_{st}, GT_t, GTd_{it}, GTf_{it}, GTs_{it}, G, GWf_{it}, GWs_{it} \geq 0 \tag{118}$$

$$Wf_{it}, Ws_{it} \in \{0,1\} \tag{119}$$

where $G=1/T_{|t|}$, GWf_{it}, and GWs_{it} are the auxiliary variables for the Glover's linearization scheme. All the other continuous variables have a one-to-one correlation with the variables in (C2-P), that is, $S_{st}=GS_{st}/G$.

We also provide the MILP formulation (C2-PD) of the subproblem to be solved in the parametric algorithm as follows.

$$(\mathbf{C2-PD})\ \max \left[\sum_{s\in P} p_s S_{s|t|} - \sum_i \sum_t \left(Ws_{it} f_{ij} + Bs_{it} b_{ij} \right) \right] - q \cdot T_{|t|} \quad \text{s.t.} \quad \text{constraints} \quad (58)\text{-}(84) \tag{120}$$

We solve four instances of this problem with the number of time points ranging from 9 to 12, using the proposed reformulation–linearization method and the parametric algorithm, as well as using the general purpose MINLP solvers SBB, DICOPT, and the global optimizer BARON 11. The computational results are presented in Table 4. As the number of time points increases from 9 to 12, the computational time increases significantly. The computational time for the reformulation–linearization method increases from 11.28 to 1040.75 s, whereas for the parametric algorithm it increases from 11.89 to 751.72 s. Both methods are able to obtain global optimums within 20 min for all four instances. For Instance 1, the parametric algorithm is slightly slower than the reformulation–linearization method, whereas for Instances 2–4 the parametric algorithm tends to be relatively faster. Among the three general-purpose MINLP solvers, DICOPT appears to be the most efficient. However, as the outer-approximation method does not guarantee global optimality for pseudoconvex/pseudoconcave functions, DICOPT only returns suboptimal solutions for Instances 1–3, with the gaps relative to the global optimums ranging from 0.3 to 17.5%. It fails to obtain any feasible solutions for Instance 4. SBB returns suboptimal solutions for all four instances after 1 h, and the gaps relative to the global optimums range from 2.8 to 43.7%. BARON 11 returns suboptimal solutions for Instances 1, 2, and 4, and the gaps relative to the global optimums range from 0.3 to 102.9%. The suboptimal solution obtained by BARON 11 for Instance 4 is negative (−2.008), which means that the operational cost can be greater than the revenue in this case. For Instance 3, BARON 11 fails to obtain any feasible solutions within 1 h. Overall, for this scheduling problem, of which the objective is the maximization of productivity, the proposed reformulation–linearization method is shown to be much more efficient than DICOPT, SBB, and BARON.

Conclusions

In this article, we proposed a novel and efficient method for the global optimization of MILFP problems that consists of reformulating them into an equivalent MILP form. The reformulation–linearization approach is based on the integration of Charnes–Cooper transformation and Glover's linearization schemes. The detailed derivation of the proposed reformulation–linearization was presented, along with a rigorous analysis on the problem size and reformulation properties. We showed that reformulating MILFP problems into their equivalent MILP problems can be an efficient method for the global optimization of large-scale MILFP problems. It needs to be solved for only once and can reflect the gap information (upper bound and lower bound) on the fly. Extensive computational studies were performed to demonstrate the efficiency of this method, including two scheduling problems of multipurpose batch chemical processes. The results showed that the proposed reformulation–linearization method guarantees the global optimality of the solutions and requires significantly less computational effort than general-purpose MINLP solvers, such as DICOPT and SBB, and the global optimizer BARON. Similar computational performances were observed between the proposed

Table 4. Results for Case Study 2 with Five Solution Methods

No.	Number of Time Points	Discrete Variables	Continuous Variables	Constraints	Objective	Iterations	Make Span	CPU Seconds	Solvers/Algorithms
		176	1137	3206	49.81		1002	11	Reformulation–Linearization
		176	941	2209	49.81	4	1002	12	Parametric
1	9	176	941	2209	48.416–129.318		972	3600[a]	SBB
		176	941	2209	49.641		948	14	DICOPT
		176	941	2209	49.641–70.772		948	3600[a]	BARON 11
		198	1271	3565	57.441		1110	102	Reformulation–Linearization
		198	1053	2469	57.441	4	1110	38	Parametric
2	10	198	1053	2469	32.330–342.143		1056	3600[a]	SBB
		198	1053	2469	51.264		918	165	DICOPT
		198	1053	2469	36.719–220.316		972	3600[a]	BARON 11
		220	1405	3924	63.719		1218	467	Reformulation–Linearization
		220	1165	2729	63.719	4	1218	152	Parametric
3	11	220	1165	2729	55.278–548.425		1404	3600[a]	SBB
		220	1165	2729	52.64		894	1839	DICOPT
		220	1165	2729	–			3600[b]	BARON 11
		242	1476	4283	68.833		1326	1041	Reformulation–Linearization
		242	1277	2989	68.833	4	1326	752	Parametric
4	12	242	1277	2989	59.828–684.039		1512	3600[a]	SBB
		242	1277	2989	–		–	3600[b]	DICOPT
		242	1277	2989	−2.008–567.524		2000	3600[a]	BARON 11

[a]Suboptimal solutions (lower and upper bounds) obtained after 1 h (3600 s).
[b]No feasible solutions returned after 1 h (3600 s).

reformulation–linearization method and the parametric algorithm. Specifically, the proposed approach performs with respect to the CPU time roughly a half of the parametric algorithm for the two batch scheduling applications. However, it is not fully understood yet when it is expected that the proposed reformulation–linearization method would perform better or worse than the parametric algorithm. Therefore, further studies in broader applications are appreciated in future works.

Acknowledgments

The authors gratefully acknowledge financial support from the Initiative for Sustainability and Energy at Northwestern University (ISEN) and Argonne National Laboratory via a Northwestern-Argonne Early Career Investigator Award for Energy Research.

Notation

Case study 1

Set

I =	set of tasks
R =	set of resources
R^{EQ} =	set of equipment units
R^{FP} =	set of final products
R^{RM} =	set of raw materials
T =	set of event points

Parameters

$b_i^{\max}$ =	maximum batch size of task i
$b_i^{\min}$ =	minimum batch size of task i
$fcon_{r,i}$ =	fixed consumption of resource$r$at the start of task i
$fpro_{r,i}$ =	fixed production of resource r at the end of task i
ft_i =	fixed duration of task i
$g^{\max}$ =	upper bound for additional variable G
$h^{\max}$ =	upper bound of the cycle time
$h^{\min}$ =	lower bound of the cycle time
pr_r =	market value for resource r
ro_r =	total available amount of resource r
$vcon_{r,i}$ =	variable consumption of resource r at the start of task$i$
$vpro_{r,i}$ =	variable production of resource r at the end of task$i$
vt_i =	variable duration of task i
Δt =	the number of time intervals a task can span

Continuous variables

Amt_r =	net consumption/production of resource r over the cycle
$B_{i,t,t'}$ =	batch size task i starting at event point t and ending at event point t'
H =	cycle time
$R_{r,t}$ =	available amount of resource r over time interval t
T_t =	absolute time at event point t

Binary variables

$N_{i,t,t'}$ =	1 if task i starts at event point t and ends at event point t'

Case study 2

Sets

i =	set of tasks
j =	set of equipment units
s =	set of states
t =	set of time points
I_j =	subset of tasks that can be performed in unit j
I_s =	subset of tasks that consumes state s
O_s =	subset of tasks that produces state s
P =	subset of states that are final products

Parameters

b_{ij} =	variable cost for performing task i in unit j
$b_i^{\max}$ =	maximum batch size for task i
$b_i^{\min}$ =	minimum batch size for task i
d_s =	minimum demand of product s
f_{ij} =	fixed cost for performing task i in unit j
$g^{\max}$ =	upper bound for additional variable G
ht =	total time horizon
p_s =	market value of product s
pt_i^F =	fixed processing time of task i
pt_i^V =	variable processing time of task i
$s_s^{\max}$ =	maximum capacity of state s
ρ_{is}^c =	material balance coefficient regarding the consumption of state s by task i
ρ_{is}^p =	material balance coefficient regarding the consumption of state s by task i

Continuous variables

Bf_{it} =	batch size of task i that finishes at or before time point t
Bp_{it} =	batch size of task i that is active at time point t
Bs_{it} =	batch size of task i when it starts at time point t
B_{ist}^I =	amount of state s consumed by task i at time point t
B_{ist}^O =	amount of state s produced by task i at time point t
S_{st} =	storage level of state s at time point t
T_t =	absolute time at time point t
Td_{it} =	duration of task i that starts at time point t
Tf_{it} =	finishing time of task i that starts or is active at time point t
Ts_{it} =	starting time of task i at time point t

Binary variables

Wf_{it} =	1 if task i finishes at or before time point t
Ws_{it} =	1 if task i starts at time point t

Literature Cited

1. Floudas CA. *Deterministic Global Optimization: Theory, Methods and Applications*. Boston: Kluwer Academic Publishers, 1999.
2. Tawarmalani M, Sahinidis NV. *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software, and Applications*. Boston: Kluwer Academic Publishers, 2002.
3. Yue D, You F. Sustainable scheduling of batch processes under economic and environmental criteria with MINLP models and algorithms. *Comput Chem Eng*. 2013;54:44–59.
4. Sahinidis NV, Grossmann IE. MINLP model for cyclic multiproduct scheduling on continuous parallel lines. *Comput Chem Eng*. 1991; 15(2):85–103.
5. Pinto JM, Joly M, Moro LFL. Planning and scheduling models for refinery operations. *Comput Chem Eng*. 2000;24(9-10):2259–2276.
6. You FQ, Castro PM, Grossmann IE. Dinkelbach's algorithm as an efficient method to solve a class of minlp models for large-scale cyclic scheduling problems. *Comput Chem Eng*. 2009;33(11):1879–1889.
7. Yue D, Kim MA, You F. Design of sustainable product systems and supply chains with life cycle optimization based on functional unit: general modeling framework, mixed-integer nonlinear programming algorithms and case study on hydrocarbon biofuels. *ACS Sust Chem Eng*. 2013, DOI: 10.1021/sc400080x. In press.
8. Chu Y, You FQ. Integration of production sequencing and dynamic optimization for single multi-product CSTR: generalized benders decomposition coupled with global mixed-integer fractional programming. *Comput Chem Eng*. 2013, submitted).
9. Chu Y, You F. Integration of scheduling and control with online closed-loop implementation: fast computational strategy and large-scale global optimization algorithm. *Comput Chem Eng*. 2012;47:248–268.
10. Mendez CA, Cerda J, Grossmann IE, Harjunkoski I, Fahl M. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Comput Chem Eng*. 2006;30(6-7):913–946.
11. Floudas CA, Lin XX. Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review. *Comput Chem Eng*. 2004;28(11):2109–2129.
12. Glover F. Improved linear integer programming formulations of nonlinear integer problems. *Manag Sci*. 1975;22(4):455–460.
13. Warichet. *Scheduling of Mixed Batch-Continuous Production Lines*. Ph.D Thesis, Université Catholique de Louvain: Belgium, 2007.
14. Bajalinov EB. *Linear-Fractional Programming: Theory, Method, Applications and Software*. Boston: Kluwer Academic Publishers, 2003.
15. Barros AI. *Discrete and Fractional Programming Techniques for Location Models*. Boston: Kluwer Academic Publishers, 1998.
16. Charnes A, Cooper WW. Programming with linear fractional functionals. *Nav Res Log Q*. 1962;9(3-4):181–186.
17. Wolsey LA. *Integer Programming*. New York: John Wiley & Sons, Inc., 1998.
18. Li H-L. A global approach for general 0–1 fractional programming. *Euro J Oper Res*. 1994;73(3):590–596.
19. Wu T-H. A note on a global approach for general 0–1 fractional programming. *Euro J Oper Res*. 1997;101(1):220–223.
20. Billionnet A. Approximation algorithms for fractional knapsack problems. *Oper Res Lett*. 2002;30(5):336–342.

21. Billionnet A. Approximate and exact solution methods for the hyperbolic 0–1 knapsack problem. *Inform Syst Oper Res.* 2002;40(2):97–110.
22. Yue D, You F. Planning and scheduling of flexible process networks under uncertainty with stochastic inventory: MINLP models and algorithm. *AIChE J.* 2013;59(5):1511–1532.
23. Castro PM, Barbosa-Povoa AP, Novais AQ. Simultaneous design and scheduling of multipurpose plants using resource task network based continuous-time formulations. *Ind Eng Chem Res.* 2005;44(2):343–357.
24. Maravelias CT, Grossmann IE. New general continuous-time state-task network formulation for short-term scheduling of multipurpose batch plants. *Ind Eng Chem Res.* 2003;42(13):3056–3074.
25. Dinkelbach W. On Nonlinear fractional programming. *Manag Sci.* 1967;13(7):492–498.
26. Bazaraa MS, Sherali HD, Shetty CM. *Nonlinear Programming: Theory and Algorithms*. New York: Wiley, 2004.
27. Borchers B, Mitchell JE. An improved branch-and-bound algorithm for mixed-integer nonlinear programs. *Comput Oper Res.* 1994; 21(4):359–367.
28. Gupta OK, Ravindran A. branch and bound experiments in convex nonlinear integer programming. *Manag Sci.* 1985;31(12):1533–1546.
29. Bussieck MR, Drud AS. SBB: a new solver for mixed integer nonlinear programming. OR 2001, Duisburg, Germany.
30. Westerlund T, Pettersson F. An extended cutting plane method for solving convex MINLP problems. *Comput Chem Eng.*;19(Supplement 1):131–136.
31. Westerlund T, Porn R. Solving pseudo-convex mixed integer optimization problems by cutting plane techniques. *Optim Eng.* 2002;3(3):253–280.
32. Tawarmalani M, Sahinidis NV. Global optimization of mixed-integer nonlinear programs: a theoretical and computational study. *Math Program.* 2004;99(3):563–591.
33. Adjiman CS, Androulakis IP, Floudas CA. Global optimization of mixed-integer nonlinear problems. *AIChE J.* 2000;46(9):1769–1797.
34. Quesada I, Grossmann IE. A global optimization algorithm for linear fractional and bilinear programs. *J Glob Optim.* 1995;6(1):39–76.
35. Smith EMB, Pantelides CC. A symbolic reformulation/spatial branch-and-bound algorithm for the global optimisation of nonconvex MINLPs. *Comput Chem Eng.* 1999;23(4-5):457–478.
36. Sahinidis NV. BARON: A general purpose global optimization software package. *J Glob Optim.* 1996;8(2):201–205.
37. Rosenthal RE. *GAMS—A User's Manual*. GAMS Development Corporation: Washington, DC, 2008.
38. Geoffrion AM. Generalized benders decomposition. *J Optim Theor Appl.* 1972;10(4):237–260.
39. Duran MA, Grossmann IE. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Math Program.* 1986; 36(3):307–339.
40. Viswanathan J, Grossmann IE. A combined penalty-function and outer-approximation method for MINLP optimization. *Comput Chem Eng.* 1990;14(7):769–782.
41. Jain V, Grossmann IE. Cyclic scheduling of continuous parallel-process units with decaying performance. *AICHE J.* 1998;44(7): 1623–1636.
42. Bonami P, Biegler LT, Conna AR, Cornuéjols G, Grossmann IE, Laird CD, Lee J, Lodi A, Margot F, Sawaya N, Wächter A. An algorithmic framework for convex mixed integer nonlinear programs. *Discret Optim.* 2008;5(2):186–204.
43. Quesada I, Grossmann IE. An LP/NLP based branch and bound algorithm for convex MINLP optimization problems. *Comput Chem Eng.* 1992;16(10-11):937–947.
44. Schaible S. Fractional programming. II, on Dinkelbach's algorithm. *Manag Sci.* 1976;22(8):868–873.
45. Bixby RE. Solving real-world linear programs: a decade and more of progress. *Oper Res.* 2002;50(1):3–15.
46. Zhong Z, You FQ. Global convergent exact and inexact parametric algorithms for solving large-scale mixed-integer fractional programs and applications in process systems engineering. *Comput Chem Eng.* 2013, submitted).
47. Cornuejols G. Valid inequalities for mixed integer linear programs. *Math Program.* 2008;112(1):3–44.
48. Mitchell JE. Branch-and-cut algorithms for combinatorial optimization problems. In: Pardalos PM, Resende MGC, editors. *Handbook of Applied Optimization*: Oxford University Press Inc., 2002, pp. 65–77.
49. Pantelides CC. Unified frameworks for the optimal process planning and scheduling: In: Proceedings of the Second Conference on Foundations of Computer Aided Operations, Cache Publications: New York; 1994.
50. Schilling G, Pantelides CC. Optimal periodic scheduling of multipurpose plants. *Comput Chem Eng.* 1999;23(4-5):635–655.
51. Shah N, Pantelides CC, Sargent RWH. Optimal periodic scheduling of multipurpose batch plants. *Ann Oper Res.* 1993;42(1):193–228.
52. Kondili E, Pantelides CC, Sargent RWH. A general algorithm for short-term scheduling of batch-operations .1. MILP formulation. *Comput Chem Eng.* 1993;17(2):211–227.
53. Wu D, Ierapetritou M. Cyclic short-term scheduling of multiproduct batch plants using continuous-time representation. *Comput Chem Eng.* 2004;28(11):2271–2286.
54. Wassick JM, Agarwal A, Akiya N, Ferrio J, Bury S, You F. Addressing the operational challenges in the development, manufacture, and supply of advanced materials and performance products. *Comput Chem Eng.* 2012;47:157–169.
55. Chu Y, Wassick JM, You F. Efficient scheduling method of complex batch processes with general network structure via agent-based modeling. *AIChE J.* 2013;59:2884–2906.
56. Capon-Garcia E, Bojarski AD, Espuna A, Puigjaner L. Multiobjective optimization of multiproduct batch plants scheduling under environmental and economic concerns. *AICHE J.* 2011;57(10):2766–2782.

Appendix

Charnes-Cooper transformation for continuous LFPs

Charnes and Cooper[15] proposed a method to reformulate a LFP to an equivalent linear program. This is known as the Charnes–Cooper transformation.

Consider a general LFP given below,

$$(\mathbf{LFP}) \quad \max \quad \frac{A_0+\sum_{i\in I} A1_i x_i}{B_0+\sum_{i\in I} B1_i x_i} \tag{A1}$$

$$\text{s.t.}\, C_{0k}+\sum_{i\in I} C1_{ik}x_i=0, \quad \forall k\in K \tag{A2}$$

$$x_i\geq 0, \forall i\in I$$

Charnes–Cooper transformation first introduces a new variable u and a set of variables z_i, such that $u=\frac{1}{B_0+\sum_{i\in I} B1_i x_i}$ and $z_i=\frac{x_i}{B_0+\sum_{i\in I} B1_i x_i}=x_i\cdot u$.

With the variables u and z_i, the objective function (A1) can be transformed to the following function

$$\frac{A_0+\sum_{i\in I} A1_i x_i}{B_0+\sum_{i\in I} B1_i x_i}=A_0\cdot u+\sum_{i\in I} A1_i\cdot z_i \tag{A3}$$

Multiplying both sides of constraint (A2) by u, which is nonzero, it leads to the following equality,

$$C_{0k}\cdot u+\sum_{i\in I} C1_{ik}\cdot x_i\cdot u=C_{0k}\cdot u+\sum_{i\in I} C1_{ik}\cdot z_i=0, \forall k\in K, \tag{A4}$$

By definition, the variable u should satisfy the following constraint,

$$1=u\cdot\left(B_0+\sum_{i\in I} B1_i x_i\right)=B_0\cdot u+\sum_{i\in I} B1_i\cdot x_i\cdot u=B_0\cdot u+\sum_{i\in I} B1_i\cdot z_i \tag{A5}$$

Therefore, the problem (LFP) can be transformed to the following equivalent linear program (LFP-R)

$$(\mathbf{LFP-R}) \ \max A_0\cdot u+\sum_{i\in I} A1_i z_i \tag{A6}$$

$$\text{s.t.} C_{0k} \cdot u + \sum_{i \in I} C1_{ik} \cdot z_i = 0, \forall k \in K \tag{A7}$$

$$B_0 \cdot u + \sum_{i \in I} B1_i \cdot z_i = 1 \tag{A8}$$

$$u \geq 0, z_i \geq 0, \forall i \in I$$

Charnes–Cooper transformation reformulates a linear fraction program into a linear program by introducing an additional variable u, and an additional constraint (A8) to transform the ratio function in the objective function into a linear function. The original variables x_i are eliminated and replaced by z_i, which has the one-to-one map through the relationship of $z_i = x_i \cdot u$.

Tightest possible BigM value

Note that the tightest possible value of M can be obtained by maximizing u subject to the constraints of the original model. More precisely, we can obtain a rigorous value of M by solving the following problem

$$(\mathbf{P-M}) \quad \min B_0 + \sum_{i \in I} B1_i x_i + \sum_{j \in J} B2_j y_j \tag{A9}$$

$$\text{s.t. } C_{0k} + \sum_{i \in I} C1_{ik} x_i + \sum_{j \in J} C2_{jk} y_j = 0, \quad \forall k \in K \tag{A10}$$

$$x_i \geq 0, \forall i \in I \quad \text{and} \quad y_j \in \{0,1\}, \quad \forall j \in J \tag{A11}$$

where $B_0 + \sum_{i \in I} B1_i x_i + \sum_{j \in J} B2_j y_j > 0$.

The tightest value of M corresponds to the inverse of the optimal value of this formulation. Note that this model does not include auxiliary variables, so its computational burden might be lower than the one associated with the reformulated MILP.

Numerical examples of different BigM values

The following table shows several numerical examples using different BigM values. As can be seen, the BigM value has little effect on the CPU times for the proposed reformulation–linearization method. This is probably because the bounds automatically generated by CPLEX may be more efficient than that specified manually.

One-thread CPU times for CPLEX 12

The one-thread CPU times using CPLEX 12 for both the proposed reformulation–linearization method and the parametric method are given below. Table A2 is for the random MILFP problems. Table A3 is for case study 1. Table A4 is for case study 2. As can be observed, the one-thread CPU times roughly range from one time to four times of the four-thread CPU times shown in Table 2, 3, and 4. This suggests that the use of parallel mode can significantly improve the solution efficiency.

Table A1. CPU Times for Different M Values

	Continuous Variables	Discrete Variables	Constraints	Reformulation–Linearization	CPU Seconds		
No.	$\lvert I\rvert$	$\lvert J\rvert$	$\lvert K\rvert+1$	Objective	$M = 10^8$	$M = 10^4$	$M = 1$
20	3000	3000	2001	3.525	686	694	695
21	3000	3000	3001	3.525	1302	1309	1309

Random MILFP problems

Table A2. One-Thread Times for Random MILFP Problems

	Continuous Variables	Discrete Variables	Constraints	Reformulation–Linearization		Parametric Algorithm		
No.	$\lvert I\rvert$	$\lvert J\rvert$	$\lvert K\rvert+1$	Objective	1-Thread CPU Seconds	Iterations	Objective	1-Thread CPU Seconds
1	100	100	101	3.394	1	7	3.394	1
2	500	100	101	2.391	2	7	2.391	3
3	100	500	101	6.094	1	8	6.094	2
4	100	100	501	2.644	926	7	2.644	1099
5	500	500	101	3.546	1	7	3.546	2
6	500	500	501	3.507	8	7	3.507	17
7	1000	500	501	2.942	8	7	2.942	15
8	500	1000	501	4.114	3	7	4.114	17
9	500	500	1001	3.484	94	7	3.484	56
10	1000	1000	501	3.512	12	7	3.512	38
11	1000	1000	1001	3.512	22	7	3.512	77
12	2000	1000	1001	2.956	29	7	2.956	35
13	1000	2000	1001	4.667	22	7	4.667	73
14	1000	1000	2001	3.510	45	8	3.510	162
15	2000	2000	1001	3.469	131	7	3.469	345
16	2000	2000	2001	3.469	201	7	3.469	497
17	3000	2000	2001	3.199	104	6	3.199	94
18	2000	3000	2001	3.970	217	7	3.970	560
19	2000	2000	3001	3.469	226	7	3.469	556
20	3000	3000	2001	3.525	715	7	3.525	1808
21	3000	3000	3001	3.525	822	7	3.525	1616
22	0	500	250	18.250	111	8	18.250	193
23	0	1000	500	64.571	1	7	64.571	6
24	0	2000	1000	114.714	9	7	114.714	26
25	0	3000	1500	174.286	6	7	174.286	58

EXAMPLE 1

Table A3. One-Thread CPU Times for Case Study 1

No.	Number of event points	Step	$H^{\min}$	$H^{\max}$	Discrete variables	Continuous variables	Constraints	Objective	Iterations	Cycle Time	CPU Seconds	Solvers/ Algorithms
1	5	3	40	70	105	276	859	32.757	3	62.801	18	1-thread R-L
					105	172	375	32.757		62.801	24	1-thread parametric
2	5	3	70	100	105	276	859	32.605	3	97.585	19	1-thread R-L
					105	172	375	32.605		97.585	17	1-thread parametric
3	6	3	100	140	126	330	1029	33.396	3	105.677	222	1-thread R-L
					126	205	449	33.396		105.677	97	1-thread parametric
4	6	4	100	140	168	414	1323	33.396	3	105.677	233	1-thread R-L
					168	247	575	33.396		105.677	114	1-thread parametric
5	7	4	100	140	196	482	1542	34.108	3	103.47	3881	1-thread R-L
					196	287	670	34.108		103.47	1944	1-thread parametric

EXAMPLE 2

Table A4. One-Thread CPU Times for Case Study 2

No.	Number of Time Points	Discrete Variables	Continuous Variables	Constraints	Objective	Iterations	Make Span	CPU Seconds	Solvers/ Algorithms
1	9	176	1137	3206	49.81	4	1002	25	1-thread R-L
		176	941	2209	49.81		1002	23	1-thread parametric
2	10	198	1271	3565	57.441	4	1110	439	1-thread R-L
		198	1053	2469	57.441		1110	159	1-thread parametric
3	11	220	1405	3924	63.719	4	1218	1363	1-thread R-L
		220	1165	2729	63.719		1218	666	1-thread parametric
4	12	242	1476	4283	68.833	4	1326	4105	1-thread R-L
		242	1277	2989	68.833		1326	4323	1-thread parametric

Manuscript received Feb. 17, 2013, and revision received May. 15, 2013.