
Zhixia Zhong and Fengqi You

Abstract—In this work, we proposed novel parametric algorithms for solving large-scale mixed-integer linear and nonlinear fractional programming problems, and illustrate their application in process systems engineering. By developing an equivalent parametric formulation of the general mixed-integer fractional program (MIFP), we propose four exact parametric algorithms based on the root-finding methods, including bisection method, Newton’s method, secant method and false position method, respectively, for the global optimization of MIFPs. We also propose an inexact parametric algorithm that can potentially outperform the exact parametric algorithms for some types of MIFPs. Extensive computational studies are performed to demonstrate the efficiency of these parametric algorithms and to compare them with the general-purpose mixed-integer nonlinear programming methods. The applications of the proposed algorithms are illustrated through a case study on process scheduling. Computational results show that the proposed exact and inexact parametric algorithms are more computationally efficient than several general-purpose solvers for solving MIFPs.

I. INTRODUCTION

A mixed-integer fractional programming (MIFP) problem has an objective as a ratio of two functions, and includes both discrete and continuous variables. A general form of MIFP can be stated as the following problem (P):

\[
(P) \max \left\{ \frac{Q(x)}{D(x)} \mid x \in S \right\}
\]

where the variables \(x\) contain both continuous and discrete variables, the feasible region \(S\) is nonempty, compact, bounded and the denominator function \(D(x)\) is always positive in \(S\), i.e. \(D(x) > 0\) for \(x \in S\). The numerator function \(N(x)\) and the denominator function \(D(x)\) can be linear or nonlinear.

MIFP problems arise from a variety of real world applications. Major MIFP applications in process systems engineering can be generally categorized into three types.

The first one is to optimize the productivity of a process system, which can usually be measured by the performance per unit of time or the process output per input. These include, but are not limited to overall cost or profit over the makespan, the resource generation/consumption rate, or the product yield per raw material or utility consumption. Production scheduling problems can be formulated as a mixed-integer linear fractional programming (MILFP) problem by optimizing the productivity subject to the mixed-integer linear constraints. Cyclic process operations problems [1, 2] might involve the tradeoffs between inventory cost and fixed cost in the objective function that would lead to a mixed-integer quadratic fractional program (MIQFP).

Another important MIFP application is optimization for sustainability. Although overall environmental impact is usually used as the objective function for process optimization problems, environmental impact per functional unit could sometimes be a more appropriate objective from the life cycle assessment perspective, especially for the problems focusing on the performance of products, rather than that of the entire process. Recent MIFP applications in this area include environmental-conscious sustainable scheduling of batch processes [3, 4] and sustainable design of supply chains [5].

The third MIFP application field is on the optimization for return rate, such as return on investment, return on cost, return on risk, etc. The return rate is usually modeled by profit dividing the capital, revenue, asset or risk. One example is that capacity planning problems can be formulated as an MILFP problem to simultaneously optimize the decisions of capacity investment, inventory management, and production planning by maximizing the return over operating assets [6]. Portfolio selection problems can use financial return over risk as the objective function. Supply chain design problems can also be formulated as an MILFP by optimizing the total return over capital investment or market share capture per unit cost [7].

Most tailored MIFP solution algorithms in the literature focus on the special case of MILFP problems. Granot & Granot [8] developed valid cutting planes based on the Charnes-Cooper transformation for solving MILFP problems. A global optimization approach based on the branch-and-bound method and variable transformation to solve 0-1 fractional programs was proposed by [9]. Yue et al. [10] proposed an exact mixed-integer linear programming (MILP) reformulation for MILFP problems, although it cannot be applied to mixed-integer nonlinear fractional programs. An algorithm based on the parametric approach was proposed to solve integer linear fractional programming problems by Ishii et al. [11]. Pochet & Warichet [12] and You et al. [13] showed that the parametric approach is very efficient for solving MILFP models for cyclic scheduling, but their studies were not extended to the general mixed-integer nonlinear fractional programs to be addressed in this paper.

The goal of this paper is to propose novel and efficient algorithms based on the parametric approach for solving MIFP problems, and to illustrate their effectiveness of solving process scheduling problems. We first show that the
parametric approach for continuous fractional programs is applicable to the mixed-integer linear and nonlinear fractional programs. To solve the resulting parametric problem, which is equivalent to the original MIFP problem, we consider four one-dimension root-finding algorithms, including the bisection method, the Newton’s method, the secant method and the false position method. These are all global convergent exact methods with at least linear convergence rates. We further propose a novel, inexact parametric algorithm based on the Newton’s method for solving the equivalent parametric problem. We show that this new algorithm has a linear convergence rate, but it could be much more computationally efficient in each iteration than the exact parametric algorithms. Thus, the inexact parametric algorithm can potentially require shorter total computational times than the exact parametric algorithms for some types of MIFPs.

II. PARAMETRIC ALGORITHMS FOR MIFP PROBLEMS

A. Equivalent parametric formulation and its properties

Considering the following parametric problem \((P_q)\):

\[
(P_q) \quad F(q) = \max \{ N(x) - q \cdot D(x) \mid x \in S \}
\]

where \(x\) contains continuous variables and integer variables, \(S\) is compact and bounded, \(D(x)\geq 0\), and \(q\) is a parameter.

This parametric problem \((P_q)\) has some special properties that can be utilized for solving the original fractional program \((P)\). It is easy to show that the function \(F(q)\) is convex, strictly monotonic decreasing, continuous and has bounded subgradients for \(q \in \mathbb{I}\), and that \(F(q) = 0\) has a unique solution \(q^*\) which is exactly the global optimal objective value of the problem \((P)\). Thus, we have the following proposition for the equivalence between the parametric problem and the fractional program.

**Lemma:** \(q^* = N(x^*)/D(x^*) = \max \{ N(x)/D(x) \mid x \in S \}\) if \(F(q^*) = F(q^*,x^*) = \max \{ N(x) - q^* \cdot D(x) \mid x \in S \} = 0\), i.e. the variable \(x^*\) is a global optimal solution to the fractional programming problem \((P)\) if and only if \(x^*\) is a global optimal solution to the parametric problem \((P_q)\) with the parameter \(q^*\) such that \(F(q^*) = 0\).

We note that the equivalence proposition only requires that the denominator is always positive, i.e. \(D(x) > 0\). The proposition holds true when the problem contains both continuous and discrete variables, no matter whether the functions \(N(x)\) and \(D(x)\) are linear or nonlinear. Though there are no assumptions on the convexity of the two functions \(N(x)\) and \(D(x)\), the equivalence is only valid based on the global optimal solutions of both problems. In other words, the equivalence proposition might be invalid for local optimal solutions of the problems.

Because the ratio function in the objective is transformed into the difference, the parametric problem \((P_q)\) has fewer nonlinearities than the MIFP problem \((P)\), and might be easier to solve. For example, if \((P)\) is an MILFP problem, \((P_q)\) is a (parametric) mixed-integer linear program (MILP); if \((P)\) is an MIQFP problem with a convex \(N(x)\) and a concave \(D(x)\), \((P_q)\) is a convex mixed-integer quadratic program (MIQP). MILP and convex MIQP problems can be solved very efficiently using state-of-the-art branch-and-cut methods implemented in solvers like CPLEX. Thus, the parametric approach might be much more computationally efficient than the general-purpose MINLP methods for solving these types of MIFP problems.

Thus, solving MIFP problem \((P)\) ends up with finding the root of equation \(F(q) = \max \{ N(x) - q \cdot D(x) \mid x \in S \} = 0\). Although \(F(q)\) does not have a close-form analytical expression, there are several numerical root-finding algorithms for solving nonlinear equations, such as the bisection method, Newton’s method, secant method, false position method and others. In principle, all root-finding methods are applicable for solving the parametric problem \((P_q)\). In this paper, we only apply these four well-known root-finding algorithms to solve the equivalent parametric form of MIFP. The iteration procedures of these methods are shown in Figure 1.

![Figure 1. Iteration procedures of solving the equivalent parametric problem by using the bisection method (superscript “B”), the exact Newton’s method (superscript “N”), the secant method (superscript “S”) and the false position method (superscript “F”). q’ and q” are the starting points, and q* is the optimal solution.](image)

Due to the space limit, we only introduce the Newton’s method in details in this paper. Other parametric algorithms can be easily derived from the standard root-finding methods.

In Newton’s method, \(q_{n+1}\) is defined by,

\[
q_{n+1} = q_n - F'(q_n)/F(q_n)
\]

As shown by [13], we can use the approximated subgradient at point \(q_n\) to estimate the derivate, \(F'(q_n) = -D(x_n')\), which is the negative value of the denominator evaluated at \(x_n'\), a global optimal solution of \(\max \{ N(x) - q_n \cdot D(x) \mid x \in S \}\). Thus, we have:

3610
\[ q_{n+1} = q_n - \frac{F(q_n)}{-D(x_n^*)} = q_n + \frac{N(x_n^*) - q_n \cdot D(x_n^*)}{D(x_n^*)} = \frac{N(x_n^*)}{D(x_n^*)} \]

The full procedure of the Newton’s method for solving (P_q) is as follows:

**Step 1**: Set \( q_0 = 0 \), initialize \( n \) by \( n = 1 \).

**Step 2**: Solve \( F(q_n) = \max \{ N(x) - q_n \cdot D(x) \mid x \in S \} \) and denote the optimal solution as \( x_n^* \).

**Step 3**: If \( |F(q_n)| < \delta \) (optimality tolerance), stop and output \( x_n^* \) as the optimal solution and \( q_n \) as optimal objective. If \( |F(q_n)| \geq \delta \), let \( q_{n+1} = N(x_n^*)/D(x_n^*) \), update \( n \) with \( n + 1 \) and update \( q_n \) with \( q_{n+1} \). Go to Step 2.

In this method, the function is approximated by its tangent line and one computes the q-intercept of this tangent line. The method can iterate and converge to \( q^* \) as shown in Figure 1.

We note that this is an exact Newton’s method that requires each parametric subproblem to be solved to the global optimum. This algorithm converges superlinearly at a rate of \( \left[ 1 - D(x^*)/D(x^1) \right] \).

**B. Performance profiles of the parametric algorithms**

In order to test the computational efficiency of the proposed solution algorithms for solving large-scale MILFP problems, we conduct computational experiments based on two testing problems. The MILFP testing problem (TP1) is presented as follows:

\[(TP1) \quad \max \quad A_0 + \sum_{i \in I} A_1 x_i + \sum_{j \in J} A_2 x_j y_j \]
\[\text{s.t.} \quad \sum_{i \in I} C_{i,k} x_i + \sum_{j \in J} C_{j,k} y_j \leq D_k, \quad \forall k \in K \]
\[B_0 + \sum_{i \in I} B_1 x_i + \sum_{j \in J} B_2 y_j \geq 0.001 \]
\[x^i \leq x_i \leq x^U, \quad \forall i \in I \text{ and } y_j \in \{0,1\}, \quad \forall j \in J \]

where all the constraints are linear and the objective function has a concave quadratic function in the numerator and a convex quadratic function in the denominator.

We note that the second constraint in both testing problems is to ensure that the denominator is positive. Both problems consists of \(|I| \) continuous variables, \(|J| \) binary variables, and \(|K|+1 \) constraints. The values of \(|I|, \; |J| \) and \(|K| \) range from 100 to 2,000.

For each testing problem, we solve 45 large-scale, randomly generated instances to demonstrate the efficiency of the solution algorithms, including the exact Newton’s method, bisection method, secant method, false position method and the general MINLP solution methods via SBB solver (simple branch-and-bound algorithm), DICOPT (outer-approximation algorithm) and the global optimizer BARON 11.3. A tailored MILFP solution approach, the reformulation linearization method [10], is also used to solve the MILFP problem (TP1) for comparison. We note that this reformulation method cannot be used to solve mixed-integer nonlinear fractional programs like the MIQFP problem (TP2). Default initialization options by GAMS are used for the MINLP solvers. The parametric algorithms are initialized as described in Section 2.2.5. All the computational experiments are carried out on a computer with Intel® Core™ 2 Due CPU P8600@ 2.4 GHz 1.58 GHz and 2.98 GB RAM. Only one core is used for all computation. All models and solution procedures are coded in GAMS 23.9. The MILP and MIQFP problems in the parametric algorithms are solved using CPLEX 12, which was called in cold-start at each iteration of the parametric algorithm. The relative optimality tolerance of the MILP/MIQP solvers is set to 0 and the absolute optimality tolerance for the root-finding algorithms are set as \( \delta = 10^{-5} \). The maximum solution time for each instance is limited to 2 hours or 7,200 CPU seconds.

Figure 2 shows the performance profiles of solving the MILFP testing problems (TP1) using the aforementioned algorithms, based on the performance analysis and benchmarking methods of optimization algorithms proposed by [14]. The x-axis is the maximum computational time needed for solving the problem and the y-axis is the number of instances. If a solution method has a performance profile towards the upper left corner, it implies that this method can solve more problems within shorter time, i.e., better computational performance. We can see from Figure 2 that the performances of the parametric algorithms and reformulation-linearization method is in general better than those tested MINLP solvers in solving the MILFP problems. Among the parametric algorithms, Newton’s method and secant method, which both converge quadratically, have better performance than the bisection method and the false position method with linear convergence rates. Generally speaking, among the four tested parametric algorithms, both exact Newton’s method and the secant method are good choices in solving large-scale MILFP problems. Reformation-linearization method has comparable performance with the exact Newton’s method and the secant method for solving MILFP problems.
In this section, we present an inexact solution approach based on the Newton’s method for solving MIFPs, of which both the numerator and denominator functions are non-negative, i.e. $N(x) \geq 0$ and $D(x) > 0$. The main idea is that we start the iterations with the initial value of $q = 0$, and apply the same procedure as the Newton’s method, except that the parametric subproblem in each iteration does not need to be solved to the global optimality, i.e. 0% gap. Instead, we show that the algorithm will still converge to $q^*$, as long as that we set the initial value of $q = 0$ and that each subproblem is solved to a pre-defined relative optimality criterion less than 100%. Here we consider the same definition of relative optimality criterion as used by GAMS, i.e. $(|BP - BF|)/|BP| < \varepsilon$, where BP is the best possible solution, BF is the best solution found, and $\varepsilon$ is the relative optimality criterion [15].

The full procedure of the inexact parametric algorithm based on the Newton’s method for solving (P_q) is as follows:

**Step 1**: Set $q_1 = 0$ and initialize $n$ by setting $n = 1$.

**Step 2**: Solve $F(q_n) = \max \{ N(x) - q_n \cdot D(x) | x \in S \}$ to a relative optimality gap $\varepsilon$, where $\varepsilon < 100\%$. Denote the near-optimal solution as $x_n^\ast$.

**Step 3**: If $|F(q_n)| < \delta$ (tolerance), stop and output $x_n^\ast$ as the optimal solution and $q_n$ as optimal objective. If $|F(q_n)| \geq \delta$ , let $q_{n+1} = N(x_n^\ast)/D(x_n^\ast)$, update $n$ with $n + 1$ and update $q_n$ with $q_{n+1}$. Go to Step 2.

The main difference between this algorithm and the one presented in the previous section is the Step 2, where each subproblem does not need to be solved to global optimum. It is worth noting that in the last step, the problem will still be solved to the global optimality, because the optimal objective value in the final iteration should be zero and a solution satisfying the relative optimality gap would also lead to the objective value of zero, i.e. the global optimum. This can potentially reduce the computational time required for each iteration, although the number of iterations might increase. It is easy to prove that this inexact algorithm converges linearly with a rate of $\left[ 1 - D(x') / \eta \cdot D(x_n^\ast) \right]$, where $x_n^\ast$ is the near-optimal solution with a relative optimality gap $\varepsilon$ obtained in iteration $k$ and $\eta = 1/(1 - \varepsilon)$. Although the convergence rate of this inexact algorithm is slower than that of the exact Newton’s method, it may need shorter computational time in each iteration due to the lower requirement of the solution quality for each subproblem. Thus, this inexact method can potentially outperform the exact parametric algorithms as will be shown in the next section.
proposed, more efficient inexact parametric solution algorithms for large-scale MIFP problems. All the computational environment is the same as that described in Section II. The detailed description of this scheduling problem are given in Figure 3 and the model formulation is listed below. The model is a mixed-integer linear fractional program which maximizes the productivity that is represented by the total profit over the makespan. The main tradeoff in this problem is that more time points |t| would potentially lead to better objective value, i.e. higher productivity, but it could also lead to larger problem with more variable and constraints that might be computationally intractable.

\[
\begin{align*}
\text{max} & \quad \sum_{i \in P} \rho_i S_{\alpha(i)} - \sum_{i \in P} (w_{s,i} + w_{f,i}) \\
\text{s.t.} & \quad \sum_{i \in P} W_{s,i} = \sum_{i \in P} W_{f,i}, \quad \forall i \\
& \quad \sum_{i \in P} W_{s,i} \leq 1, \quad \forall i, t \\
& \quad \sum_{i \in P} W_{f,i} \leq 1, \quad \forall j, t \\
& \quad \sum_{i \in P} \sum_{v \in T} (W_{s,i} - W_{f,i}) \leq 1, \quad \forall j, t \\
& \quad T_{d,i} = W_{s,i} + W_{f,i}, \quad \forall i, t \\
& \quad T_{f,i} \geq T_{s,i} + T_{d,i} - ht(1 - W_{s,i}), \quad \forall i, t \\
& \quad T_{f,i} \leq T_{s,i} + T_{d,i} + ht(1 - W_{s,i}), \quad \forall i, t \\
& \quad T_{f,i} - T_{f,i-t} \geq T_{d,i}, \quad \forall i, t \\
& \quad T_{f,i-t} \leq T_{i} + ht(1 - W_{f,i}), \quad \forall i, t \\
& \quad T_{s,i} = T_{i}, \quad \forall i, t \\
& \quad T_i \geq T_{i-t}, \quad \forall t > 1 \\
& \quad T_i = 0 \quad (1) \\
& \quad T_{ij} \leq ht \quad (2) \\
& \quad b_{i}^{\text{min}} W_{s,i} \leq B_{s,i} \leq b_{i}^{\text{max}} W_{s,i}, \quad \forall i, t \\
& \quad b_{i}^{\text{min}} W_{f,i} \leq B_{f,i} \leq b_{i}^{\text{max}} W_{f,i}, \quad \forall i, t \\
& \quad S_{i} \geq d_{i}, \quad \forall s \in P \\
& \quad B_{i}^{\text{min}} = \rho_{i} B_{s,i}, \quad \forall i, s, t
\end{align*}
\]

Figure 3. State-task network representation of the case study.

Figure 4. The comparison of the computational results and efficiency of using inexact Newton’s method in the scheduling problem at different relative optimality gaps (opter).

We specifically compare the exact Newton’s method with the inexact parametric algorithm proposed. The results are shown in Figure 4. By using the inexact parametric algorithm, each MILP subproblem only needs to be solved to a pre-defined relative optimality criterion between 0 and 100%. We implemented this inexact algorithm to solve the scheduling problem with different values of the relative optimality gap. We note that the case of 0% gap corresponds to the exact Newton’s method, while the rest cases with 1%, 10% and 90% gaps are for the inexact parametric algorithm. The results show that the inexact algorithm is more efficient than the exact Newton’s method for solving this scheduling problem. For all the instances, the inexact algorithm returns the same optimal solution as the exact one, but only requires around 50%–70% of the CPU times of the exact Newton’s algorithm. The time-saving is more significant for large instance with 10 time points. From the number of iterations listed in Table 1, it is interesting to see that the inexact algorithms with 1% and 10% gap require the same number of iterations as the exact algorithm, i.e. 0% gap. Because solving each MILP subproblem to 1% or 10% gap requires less computational times than solving it to 0%, it is not surprising that the inexact algorithms with 1% and 10% have better computational performance than the exact algorithm. The inexact approach with 90% gap, on the other hand, usually
requires another iteration to reach the tolerance, so it might not be as efficient as similar algorithms with 1% or 10% gap. The high efficiency of the inexact algorithm is due to the problem structure. For this particular scheduling problem, it is much easier to solve the MILP subproblem to a pre-defined optimality gap, e.g. 10%, than closing the gap to 0%. Over all, the inexact Newton’s method with a proper gap can use less time than exact Newton’s method.

V. CONCLUSION

In this paper, we presented parametric algorithms for solving the general MIFP problems. We first showed the equivalence between the parametric problem and the original MIFP problem. To solve the equivalent parametric problem we considered four one-dimension root-finding algorithms, including the bisection method, the Newton’s method, the secant method and the false position method, respectively. These are all global convergent exact algorithms. We further propose an inexact parametric algorithm, which is a variation of exact algorithm based on the Newton’s method, for the global optimization of MIFPs problem. We solved randomly generated instances to test the efficiency of the four parametric algorithms for solving large-scale MILFP and MIQFP problems, and also to compare their performance with the general-purpose global optimization methods. The results showed that the proposed algorithms had higher computational efficiency than the general-purpose solvers. Through a case study on process scheduling, the applicability and performance of the proposed algorithms were further illustrated and discussed. The results show that the inexact parametric algorithms based on Newton’s method can outperform other parametric algorithms, in solving large-scale MILFP problems, and would lead to economic benefits when they are applied to process scheduling problems.

REFERENCES