MÜHENDİSLİK ve DOĞA BİLİMLERİ FAKÜLTESİ

Uluslararası Mühendislik Araştırma ve Geliştirme Dergisi
International Journal of Engineering Research and Development

UMAGD, (2024) 16(1), 20-29.

10.29137/umagd.1323701

Cilt/Volume:16 Sayı/Issue:01 Ocak/January 2024

Araştırma Makalesi / Research Article

Mixed-Integer Second-Order Cone Programming Reformulations of a Fractional 0-1 Program for Task Assignment

İş Atama İçin Kesirli Bir 0-1 Programın Kısmi Tam Sayılı İkinci Mertebeden Koni Programlama Biçimlendirmeleri

Murat Güngör [1]

[1] *Industrial Engineering Department, Istanbul Medeniyet University, 34700 Istanbul, TURKEY*

Başvuru/Received:*08/07/2023* ***Kabul/Accepted***: *08/11/2023* ***Çevrimiçi Basım/Published Online:*** *31/01/2024*
Son Versiyon/Final Version: *31/01/2024*

Abstract
Fractional 0-1 programming is a subfield of nonlinear integer optimization in which the objective is to optimize the sum of ratios of affine functions subject to a set of linear constraints. It is well-known that fractional 0-1 programs can be formulated as mixed-integer linear programs. Recently, several alternative mixed-integer second-order cone programming reformulations have been proposed for fractional 0-1 programs. These reformulations, which can be solved directly by standard commercial solvers, have been reported to be efficient for certain types of problems. In this paper, we consider a task assignment problem with respect to preferences, where the goal is to maximize total weighted satisfaction while maintaining a fair distribution. The problem's mathematical model is naturally a fractional 0-1 program. We investigate three mixed-integer second-order cone programming reformulations thereof, and we compare, by means of a computational study, the performance of these reformulations with a benchmark mixed-integer linear programming formulation that was proposed and analyzed in the literature before. The latter, namely the mixed-integer linear programming formulation, turns out to be significantly better for the problem in question.

Key Words
"fractional 0-1 programming, hyperbolic 0-1 programming, mixed-integer conic quadratic programming, task assignment, preferences"

Öz
Kesirli 0-1 programlama, doğrusal olmayan tam sayılı en iyilemenin bir alt alanıdır. Amaç, afin fonksiyonlardan oluşan bir kesirler toplamının doğrusal kısıtlar altında en iyilenmesidir. Kesirli 0-1 programların kısmi tam sayılı doğrusal programlar olarak biçimlendirilebildiği iyi bilinmektedir. Yakın zamanda, kesirli 0-1 programlar için çeşitli alternatif kısmi tam sayılı ikinci mertebeden koni programlama biçimlendirmeleri önerilmiştir. Bu biçimlendirmeler, standart ticari çözücülerle doğrudan çözülebilmektedir ve bunların bazı problem tipleri için verimlilikleri bildirilmiştir. Bu makalede, amacın adil bir yük dağıtımı altında toplam ağırlıklı memnuniyeti en büyütmek olduğu, tercihleri dikkate alan bir iş atama problemi ele alınmaktadır. Problemin matematik modeli doğal olarak kesirli bir 0-1 programdır. Bu programın üç kısmi tam sayılı ikinci mertebeden koni programlama biçimlendirmesi incelenmiş ve bunlar, bilgisayar deneyleri yardımıyla, daha önce literatürde önerilen ve incelenen denek taşı bir kısmi tam sayılı doğrusal programlama biçimlendirmesi ile kıyaslanmıştır. Kısmi tam sayılı doğrusal programlama biçimlendirmesinin söz konusu iş atama problemi için kayda değer derecede daha iyi sonuçlar verdiği görülmüştür.

Anahtar Kelimeler
"kesirli 0-1 programlama, hiperbolik 0-1 programlama, kısmi tam sayılı konik karesel programlama, kısmi tam sayılı ikinci mertebeden koni programlama, iş atama, tercihler"

***Responsible Author:** murat.gungor@medeniyet.edu.tr

1. Introduction

Second-order cone programming (SOCP) is a subfield of nonlinear convex optimization in which a linear function is minimized over the intersection of an affine set and the product of second-order (quadratic) cones (Alizadeh and Goldfarb, 2003; Lobo et al., 1998; Boyd and Vandenberghe, 2004). SOCP, also known as conic quadratic programming, includes linear and convex quadratic programming as special cases, but is less general than semidefinite programming (Lobo et al., 1998). Like the problems in these classes, SOCPs can be solved in polynomial time by interior point methods (Alizadeh and Goldfarb, 2003). (We use the acronym SOCP for *second-order cone programming* as well as *second-order cone program*—the meaning shall be clear from context. Similarly, for other such acronyms.) Mixed-integer second-order cone programming (MISOCP) is an extension of SOCP, where some of the decision variables are bound to take on integer values (Benson and Sağlam, 2014).

Fractional 0-1 programming (FP) problems are nonlinear integer optimization problems in which the goal is to minimize or maximize the sum of ratios of affine functions subject to a set of linear constraints (Borrero et al., 2017). FP is also referred to as hyperbolic 0-1 programming. Borrero et al. (2017) overview the literature on FPs including their uses, NP-hardness, and solution techniques.

It is possible to express FPs as equivalent mixed-integer linear programs (MILPs) in several ways (Li, 1994; Wu, 1997; Tawarmalani et al., 2002; Borrero et al., 2016). Recently, MISOCP reformulations are proposed for FPs. Şen et al. (2018) give such a reformulation for a special FP, namely assortment optimization. Later, Mehmanchi et al. (2019) develop a number of MISOCP reformulations for general FPs, and explore the relationship between equivalent MILP and MISOCP reformulations. They show that gluing the ideas underlying these two reformulation types lets one to push the boundaries of the current state-of-the-art results and handle problems of larger size.

Güngör (2019) gives a novel application of FP: a task assignment problem with respect to preferences, where the objective is to maximize total weighed satisfaction while maintaining a fair distribution of loads. He shows that the problem is NP-complete, gives three equivalent MILP formulations, and provides a brief discussion of some practical issues. Also he suggests an MILP-based heuristic, and performs experiments on randomly generated instances.

In this paper, we reformulate the task assignment problem with respect to preferences as MISOCPs, and by means of a computational study we compare the performance of these reformulations with a benchmark MILP formulation. Outline of the paper is as follows: We recall the precise definition of the problem in Section 2, and the benchmark MILP formulation in Section 3. Next, we give three equivalent MISOCP reformulations in Section 4. The computational study is provided in Section 5. Finally, we summarize our main conclusions in Section 6.

2. Problem Definition

For the task assignment problem with respect to preferences, we use the same notation as in Güngör (2019): Let I, J denote the number of people and tasks, and i, j the respective indices. For each pair (i, j) there is a positive number a_{ij} that shows the preference of person i over task j. The larger the a_{ij} the more preferred is the task j for person i. Every task has a specific load b_j. A parameter d represents the maximum acceptable difference from the mean load $\mu = (1/I) \times \sum_j b_j$ per person. We assume $d < \mu$. All data are nonnegative integers. A weight may be associated with each person, but we will not do so for the sake of simplicity. Let x_{ij} be a 0-1 variable defined as 1 if task j is assigned to person i, and 0 otherwise. A list of indexes, parameters, and decision variables for the problem with short explanations can be seen in Table 1.

Table 1. Indexes, parameters, and decision variables for the problem.

Symbol(s)	Explanation
i, j	indexes for people and tasks
I, J	number of people and tasks
a_{ij}	preference of person i over task j
b_j	load of task j
d	maximum acceptable difference from average load
x_{ij}	1 if task j is assigned to person i, and 0 otherwise

Satisfaction of person i is defined as the ratio $\sum_j a_{ij} b_j x_{ij} \,/\, \sum_j b_j x_{ij}$. Let X be the set of all $x = (x_{ij}) \in \{0, 1\}^{I \times J}$ such that

- $\sum_i x_{ij} = 1$ for all j, and
- $\mu - d \leq \sum_j b_j x_{ij} \leq \mu + d$ for all i.

Task assignment problem with respect to preferences can be stated as an FP:

$$\max_{x\in X}\sum_i \frac{\sum_j a_{ij}b_jx_{ij}}{\sum_j b_jx_{ij}}. \tag{1}$$

Denote the maximum possible preference level by $a_{\max}$. Then the problem can be equivalently stated with a minimization objective:

$$\min_{x\in X}\sum_i \frac{\sum_j (a_{\max}-a_{ij})b_jx_{ij}}{\sum_j b_jx_{ij}}. \tag{2}$$

Relationship between the optimal objective values $z^*_{\max}$ and $z^*_{\min}$ of (1) and (2), respectively, is given by

$$z^*_{\max}+z^*_{\min} = a_{\max}I.$$

3. Benchmark Mixed-Integer Linear Formulation

Let t_i be defined by

$$t_i = \frac{\sum_j \left(a_{\max}-a_{ij}\right)b_jx_{ij}}{\sum_j b_jx_{ij}} \tag{3}$$

If a_{min} denotes the minimum possible preference level, then $0 \le t_i \le a_{\max}-a_{\min}$ for all i. Introduce new binary variables w_{ik} by the equalities $\sum_{j=1}^{J} b_jx_{ij} = \sum_{k=1}^{K} 2^{k-1}w_{ik}$ for each i, where K is $\log_2(1+\sum_j b_j)$ rounded up to the nearest integer. The relation $\sum_k 2^{k-1}w_{ik}t_i = \sum_j a_{ij}b_jx_{ij}$ guarantees that (3) holds true, so it suffices to linearize the products $w_{ik}t_i$ in order to obtain a linear program. In general, for $x \in \{0,1\}$ and $y^L \le y \le y^U$, the expression xy necessarily equals z provided that the following four inequalities are satisfied (Adams and Forrester, 2005; Glover, 1975):

$$y^Lx \le z \le y^Ux, \tag{4}$$
$$y + y^U(x-1) \le z \le y + y^L(x-1). \tag{5}$$

Indeed, if $x=0$, then $z=0$ by virtue of (4), and (5) is redundant; if $x=1$, then $z=y$ by virtue of (5), and (4) is redundant—in any case, $z=xy$. (Note that y^L need not be nonnegative. If it is negative, then y as well as z are free variables.) Therefore, letting $t^L=0$ and $t^U=a_{\max}-a_{\min}$, the FP (2) can be written equivalently as an MILP, where the z_{ik} represent $w_{ik}t_i$:

min	$\sum_i t_i$		(6a)
s.t.	$\sum_i x_{ij} = 1$	for all j	(6b)
	$\mu - d \le \sum_j b_jx_{ij} \le \mu + d$	for all i	(6c)
	$\sum_j b_jx_{ij} = \sum_k 2^{k-1}w_{ik}$	for all i	(6d)
	$\sum_k 2^{k-1}z_{ik} = \sum_j a_{ij}b_jx_{ij}$	for all i	(6e)
	$z_{ik} \le t^Uw_{ik}$	for all i,k	(6f)
	$z_{ik} \le t_i + t^L(w_{ik}-1)$	for all i,k	(6g)
	$z_{ik} \ge t^Lw_{ik}$	for all i,k	(6h)
	$z_{ik} \ge t_i + t^U(w_{ik}-1)$	for all i,k	(6i)
	$t^L \le t_i \le t^U$	for all i	(6j)
	$x_{ij}, w_{ik} \in \{0,1\},\quad t_i, z_{ik} \ge 0$	for all i,j,k.	(6k)

Constraint (6d) defines the w_{ik}. Inequalities (6f)-(6i) ensure that $z_{ik} = w_{ik}t_i$ for any feasible solution. Consequently, equality (6e) together with (6d) imply that t_i is given by (3). Formulation (6) can be regarded as the minimization version of the best of the three MILP formulations investigated in Güngör (2019). Hence, it will be used as a benchmark for assessing the performance of the MISOCP reformulations to be given below.

4. Mixed-Integer Second-Order Cone Programming Reformulations

In this section, we reformulate (2), namely the minimization version of the task assignment problem with respect to preferences, as MISOCPs. It will be convenient to make the following definitions:

$$r_i = \sum_j b_jx_{ij},\quad y_i = \frac{1}{\sum_j b_jx_{ij}} = \frac{1}{r_i}.$$

Clearly, for $x \in X$, we have $\mu - d \leq r_i \leq \mu + d$ and $1/(\mu + d) \leq y_i \leq 1/(\mu - d)$ for all i. We let $y^L = 1/(\mu + d)$ and $y^U = 1/(\mu - d)$ for brevity.

4.1. First Reformulation

Problem (2) can be written as

$$\begin{array}{lll} \min & \sum_i t_i & \\ \text{s.t.} & t_i \geq \frac{\sum_j (a_{\max} - a_{ij}) b_j x_{ij}}{\sum_j b_j x_{ij}} & \text{for all } i \\ & x \in X, \quad t_i \geq 0 & \text{for all } i. \end{array}$$

The inequality above holds as equality at any optimal solution. Since the x_{ij} are binary, we have $x_{ij}^2 = x_{ij}$ so that $\sum_j (a_{\max} - a_{ij}) b_j x_{ij} = \sum_j (a_{\max} - a_{ij}) b_j x_{ij}^2$, and the above mathematical program can be cast as

$$\begin{array}{lll} \min & \sum_i t_i & \\ \text{s.t.} & t_i r_i \geq \sum_j (a_{\max} - a_{ij}) b_j x_{ij}^2 & \text{for all } i \\ & r_i = \sum_j b_j x_{ij} & \text{for all } i \\ & x \in X, \quad r_i, t_i \geq 0 & \text{for all } i. \end{array}$$

The nonlinear constraint $t_i r_i \geq \sum_j (a_{\max} - a_{ij}) b_j x_{ij}^2$ is a rotated cone constraint, which can be readily used with standard commercial solvers for MISOCP. Thus, our first conic reformulation of (2) can be explicitly stated as follows—we call it MISOCP1:

$$\begin{array}{llll} \min & \sum_i t_i & & (7a) \\ \text{s.t.} & t_i r_i \geq \sum_j (a_{\max} - a_{ij}) b_j x_{ij}^2 & \text{for all } i & (7b) \\ & r_i = \sum_j b_j x_{ij} & \text{for all } i & (7c) \\ & \sum_i x_{ij} = 1 & \text{for all } j & (7d) \\ & \mu - d \leq r_i \leq \mu + d & \text{for all } i & (7e) \\ & x_{ij} \in \{0, 1\}, \quad r_i, t_i \geq 0 & \text{for all } i, j. & (7f) \end{array}$$

The above MISOCP was originally proposed by Atamtürk and Gomez (2020). We refer to it as the compact reformulation.

4.2. Second Reformulation

Next, we rewrite (2) as

$$\begin{array}{lll} \min & \sum_i t_i & \\ \text{s.t.} & t_i = \sum_j (a_{\max} - a_{ij}) b_j x_{ij} \, y_i & \text{for all } i \\ & \sum_j b_j x_{ij} \, y_i = 1 & \text{for all } i \\ & x \in X, \quad t_i, y_i \geq 0 & \text{for all } i. \end{array}$$

As in Section 3, the above mathematical program can be linearized as follows, where the z_{ij} represent $x_{ij} y_i$:

$$\begin{array}{lll} \min & \sum_i t_i & \\ \text{s.t.} & t_i = \sum_j (a_{\max} - a_{ij}) b_j z_{ij} & \text{for all } i \\ & \sum_j b_j z_{ij} = 1 & \text{for all } i \\ & z_{ij} \leq y^U x_{ij} & \text{for all } i, j \\ & z_{ij} \leq y_i + y^L (x_{ij} - 1) & \text{for all } i, j \\ & z_{ij} \geq y^L x_{ij} & \text{for all } i, j \\ & z_{ij} \geq y_i + y^U (x_{ij} - 1) & \text{for all } i, j \\ & x \in X, \quad t_i, y_i, z_{ij} \geq 0 & \text{for all } i, j. \end{array}$$

For $x_{ij} \in \{0, 1\}$, we have $z_{ij} r_i = z_{ij} / y_i = x_{ij} = x_{ij}^2$, so the rotated cone constraint $z_{ij} r_i \geq x_{ij}^2$ is valid for the above formulation, and can be used to strengthen it. Adding also the second-order cone representable inequality $y_i r_i \geq 1$, we obtain another conic formulation of (2)—we call it MISOCP2:

$$\begin{array}{lll}
\min & \sum_i t_i & \quad (8a)\\
\text{s.t.} & t_i = \sum_j (a_{\max} - a_{ij}) b_j z_{ij} \quad \text{for all } i & \quad (8b)\\
& \sum_j b_j z_{ij} = 1 \quad \text{for all } i & \quad (8c)\\
& z_{ij} \le y^U x_{ij} \quad \text{for all } i,j & \quad (8d)\\
& z_{ij} \le y_i + y^L (x_{ij} - 1) \quad \text{for all } i,j & \quad (8e)\\
& z_{ij} \ge y^L x_{ij} \quad \text{for all } i,j & \quad (8f)\\
& z_{ij} \ge y_i + y^U (x_{ij} - 1) \quad \text{for all } i,j & \quad (8g)\\
& r_i = \sum_j b_j x_{ij} \quad \text{for all } i & \quad (8h)\\
& z_{ij} r_i \ge x_{ij}^2 \quad \text{for all } i,j & \quad (8i)\\
& y_i r_i \ge 1 \quad \text{for all } i & \quad (8j)\\
& \sum_i x_{ij} = 1 \quad \text{for all } j & \quad (8k)\\
& \mu - d \le r_i \le \mu + d \quad \text{for all } i & \quad (8l)\\
& x_{ij} \in \{0,1\},\ r_i, t_i, y_i, z_{ij} \ge 0 \quad \text{for all } i,j. & \quad (8m)
\end{array}$$

The above MISOCP was originally proposed by Şen et al. (2018) in the context of assortment optimization. We refer to it as the extended reformulation.

We shall prove that MISOCP2 has a stronger relaxation than MISOCP1. In other words, when the constraints $x_{ij} \in \{0,1\}$ are replaced by $0 \le x_{ij} \le 1$ in the two formulations, the optimal objective value of the SOCP resulting from (8) is greater than or equal to that of (7). The proof is adapted from Mehmanchi et al. (2019). We start with MISOCP1: Constraint (7b) can be written as

$$t_i \ge \sum_j (a_{\max} - a_{ij}) b_j \frac{x_{ij}^2}{r_i}.$$

Substituting x_{ij}^2/r_i with a new variable $\overline{z_{ij}}$ and adding the inequality $\overline{z_{ij}} \ge x_{ij}^2/r_i$, we obtain a mathematical program equivalent to MISOCP1:

$$\begin{array}{lll}
\min & \sum_i t_i & \\
\text{s.t.} & t_i \ge \sum_j (a_{\max} - a_{ij}) b_j \overline{z_{ij}} & \text{for all } i\\
& \overline{z_{ij}} r_i \ge x_{ij}^2 & \text{for all } i,j\\
& r_i = \sum_j b_j x_{ij} & \text{for all } i\\
& \sum_i x_{ij} = 1 & \text{for all } j\\
& \mu - d \le r_i \le \mu + d & \text{for all } i\\
& x_{ij} \in \{0,1\}, \quad r_i, t_i \ge 0 & \text{for all } i,j.
\end{array}$$

The first inequality must hold as an equality at any optimal solution, so the greater-than-or-equal-to sign there can be replaced by an equality sign. Then the formulation MISOCP2 has all the constraints above together with some additional ones. It follows that the relaxation of MISOCP2 is tighter than that of MISOCP1.

4.3. Third Reformulation

Finally, we consider an enhancement of MISOCP2 based on binary expansions as in Section 3. Let w_{ik} be defined by the equalities $\sum_{j=1}^{J} (a_{\max} - a_{ij}) b_j x_{ij} = \sum_{k=1}^{K_i} 2^{k-1} w_{ik}$ for each i, where K_i is $\log_2(1 + \sum_j (a_{\max} - a_{ij}) b_j)$ rounded up to the nearest integer (note that w_{ik} was defined differently in Section 3). Then (2) can be written as

$$\begin{array}{lll}
\min & \sum_i t_i & \\
\text{s.t.} & t_i \ge \sum_{k=1}^{K_i} 2^{k-1} w_{ik} y_i & \text{for all } i\\
& \sum_j (a_{\max} - a_{ij}) b_j x_{ij} = \sum_{k=1}^{K_i} 2^{k-1} w_{ik} & \text{for all } i\\
& r_i = \sum_j b_j x_{ij} & \text{for all } i\\
& y_i r_i \ge 1 & \text{for all } i\\
& x \in X,\ w_{ik} \in \{0,1\},\ r_i, t_i, y_i \ge 0 & \text{for all } i,k.
\end{array}$$

Introducing new variables $z_{ik} = w_{ik} y_i$, and using the fact that $w_{ik}^2 = w_{ik}$ for $w_{ik} \in \{0,1\}$, we obtain the convexification (note that z_{ik} was defined differently in Section 3)

$$
\begin{array}{lll}
\min & \sum_i t_i & \\
\text{s.t.} & t_i \geq \sum_{k=1}^{K_i} 2^{k-1} z_{ik} & \text{for all } i \\
& \sum_j \left(a_{\max} - a_{ij}\right) b_j x_{ij} = \sum_{k=1}^{K_i} 2^{k-1} w_{ik} & \text{for all } i \\
& z_{ik} r_i \geq w_{ik}^2 & \text{for all } i, k \\
& r_i = \sum_j b_j x_{ij} & \text{for all } i \\
& y_i r_i \geq 1 & \text{for all } i \\
& x \in X,\ w_{ik} \in \{0,1\},\ r_i, t_i, y_i, z_{ik} \geq 0 & \text{for all } i, k.
\end{array}
$$

The above formulation can be strengthened by including the linearization constraints $z_{ik} \leq y^L w_{ik}$ and $z_{ik} \geq y_i + y^U(w_{ik} - 1)$. The resulting MISOCP is our third formulation—we call it MISOCP3:

$$
\begin{array}{llll}
\min & \sum_i t_i & & (9a) \\
\text{s.t.} & t_i \geq \sum_{k=1}^{K_i} 2^{k-1} z_{ik} & \text{for all } i & (9b) \\
& \sum_j \left(a_{\max} - a_{ij}\right) b_j x_{ij} = \sum_{k=1}^{K_i} 2^{k-1} w_{ik} & \text{for all } i & (9c) \\
& z_{ik} r_i \geq w_{ik}^2 & \text{for all } i, k & (9d) \\
& r_i = \sum_j b_j x_{ij} & \text{for all } i & (9e) \\
& y_i r_i \geq 1 & \text{for all } i & (9f) \\
& z_{ik} \leq y^L w_{ik} & \text{for all } i, k & (9g) \\
& z_{ik} \geq y_i + y^U(w_{ik} - 1) & \text{for all } i, k & (9h) \\
& \sum_i x_{ij} = 1 & \text{for all } j & (9i) \\
& \mu - d \leq r_i \leq \mu + d & \text{for all } i & (9j) \\
& x_{ij}, w_{ik} \in \{0,1\},\quad r_i, t_i, y_i, z_{ik} \geq 0 & \text{for all } i, j, k. & (9k)
\end{array}
$$

Mehmanchi et al. (2019) were the first to propose the above formulation. We note that MISOCP3 has a reduced number of rotated cone constraints, but an increased number of binary variables when compared to MISOCP2.

5. Computational Study

Task loads b_j are created as random integers between $b_{\min} = 1$ and $b_{\max} = 10$. We analyze ten parameter combinations resulting from taking $I \in \{5, 15, 30, 50, 75\}$ and $J/I \in \{3, 5\}$, and we let $d \in \{1, 2\}$. We suppose that there are coefficients w_j of attractiveness, and everybody selects m tasks with respect to the discrete probability distribution defined by these coefficients. Random integers between $a_{\min} + 1$ and $a_{\max}$ are appointed as preference levels for the selected tasks. If needed, we multiply these numbers by an appropriate factor and then perform a rounding so that their maximum is precisely $a_{\max}$. To all the remaining tasks, we take $a_{\min}$ as preference level. Let $w_j = j^2$, $a_{\min} = 1$, $a_{\max} = 10$, and let m be $J/10$ rounded up to the nearest integer. This is the same instance generation procedure used in Güngör (2019).

We executed the mathematical programming formulations with C# using CPLEX 12.7 as solver, on a PC with Intel(R) Core(TM) i5-2450M CPU (2.50GHz) processor and 4 GB RAM, running a 64-bit Windows 7 operating system. For all the parameter combinations for (I,J), namely (5, 15), (5, 25), (15, 45), (15, 75), (30, 90), (30, 150), (50, 150), (50, 250), (75, 225), (75, 375), we randomly created five instances as explained in the previous paragraph, and ran the benchmark MILP formulation (6) and the three MISOCP formulations (7)-(9) in Section 4 for both $d = 1$ and $d = 2$. We set a time limit of 600 seconds. The results are given in Tables 2-5. Also, we solved the relaxed versions of these four formulations. The results are given in Tables 6 and 7.

The Feasible and Optimal columns in Tables 2-5 show respectively the number of instances for which the solver was able to find a feasible and an optimal solution. Let x be the best feasible solution and b the best bound found by the solver within the time limit. Thus, b is the least upper bound for a maximization objective, and the greatest lower bound for a minimization objective. (Note that the benchmark MILP formulation as well as all the three MISOCP reformulations have minimization as objective.) Let $z(x)$ denote the relevant objective function evaluated at x. The column Gap shows the average percent ratios $|z(x) - b|/z(x)$. A dash shows that this ratio cannot be calculated for some cases because no feasible solution was found. Finally, the column Time shows the mean CPU time in seconds.

Table 2. Solution quality and performance of the benchmark MILP formulation (6).

		$d = 1$				$d = 2$			
I	J	Feasible	Optimal	Gap (%)	Time (s)	Feasible	Optimal	Gap (%)	Time (s)
5	15	5	5	0.00	0.3	5	5	0.00	0.4
	25	5	5	0.00	0.7	5	5	0.00	0.8
15	45	5	5	0.00	4.1	5	5	0.01	77.7
	75	5	5	0.01	4.3	5	5	0.01	36.9
30	90	5	5	0.01	36.1	5	1	2.96	485
	150	5	5	0.00	71.3	5	4	0.09	358
50	150	5	3	0.77	293	5	0	20.0	600
	250	5	1	2.33	522	5	0	3.42	600
75	225	5	2	4.34	465	5	0	42.5	600
	375	4	0	-	600	5	0	48.7	600

Table 3. Solution quality and performance of the first reformulation MISOCP1 (7).

		$d = 1$				$d = 2$			
I	J	Feasible	Optimal	Gap (%)	Time (s)	Feasible	Optimal	Gap (%)	Time (s)
5	15	5	5	0.00	4.2	5	5	0.01	8.4
	25	5	0	7.51	600	5	0	7.09	600
15	45	5	0	90.0	600	5	0	89.2	600
	75	5	0	92.1	600	5	0	91.3	600
30	90	0	0	-	600	4	0	-	600
	150	0	0	-	600	5	0	96.7	600
50	150	2	0	-	600	4	0	-	600
	250	0	0	-	600	2	0	-	600
75	225	0	0	-	600	0	0	-	600
	375	0	0	-	600	0	0	-	600

Table 4. Solution quality and performance of the second reformulation MISOCP2 (8).

		$d = 1$				$d = 2$			
I	J	Feasible	Optimal	Gap (%)	Time (s)	Feasible	Optimal	Gap (%)	Time (s)
5	15	5	5	0.00	0.2	5	5	0.00	0.3
	25	5	5	0.01	176	5	0	0.01	189
15	45	5	1	0.61	493	5	0	1.99	600
	75	5	0	0.97	600	5	0	1.73	600
30	90	5	0	2.61	600	5	0	5.28	600
	150	4	0	-	600	5	0	3.01	600
50	150	2	0	-	600	5	0	7.86	600
	250	0	0	-	600	3	0	-	600
75	225	0	0	-	600	1	0	-	600
	375	0	0	-	600	0	0	-	600

Table 5. Solution quality and performance of the third reformulation MISOCP3 (9).

		$d = 1$				$d = 2$			
I	J	Feasible	Optimal	Gap (%)	Time (s)	Feasible	Optimal	Gap (%)	Time (s)
5	15	5	5	0.00	0.3	5	5	0.00	0.4
	25	5	5	0.01	0.4	5	5	0.00	0.3
15	45	5	5	0.01	77.9	5	2	2.45	478
	75	5	3	0.51	488	5	2	1.25	444
30	90	3	0	-	600	5	0	16.7	600
	150	1	0	-	600	5	0	17.1	600
50	150	1	0	-	600	4	0	-	600
	250	0	0	-	600	4	0	-	600
75	225	0	0	-	600	3	0	-	600
	375	0	0	-	600	2	0	-	600

The second column in Table 6 shows the number of instances for which the benchmark MILP formulation (6) has found the optimum within the time limit. For such instances, we computed the ratio $|z^* - z_r|/z^*$, where z^* and z_r denote the optimal objective values of the original and the relaxed problems, respectively. The last four columns in the table show the average of these ratios for given

Table 6. Average gaps for the relaxations of the benchmark MILP formulation (6) and the three MISOCP reformulations (7)-(9) with respect to the optimal objective value.

(I, J, d)	Number of instances used	Average gap for relaxation (%)			
		MILP	MISOCP1	MISOCP2	MISOCP3
(5, 15, 1)	5	71.3	78.7	2.5	4.8
(5, 15, 2)	5	70.6	78.3	3.7	7.5
(5, 25, 1)	5	69.4	77.7	1.1	3.4
(5, 25, 2)	5	69.0	77.5	1.5	5.8
(15, 45, 1)	5	88.6	91.4	3.9	5.5
(15, 45, 2)	5	88.2	91.2	5.1	7.7
(15, 75, 1)	5	89.1	91.1	1.6	3.0
(15, 75, 2)	5	88.9	91.0	2.3	5.0
(30, 90, 1)	5	93.9	95.1	6.8	8.4
(30, 90, 2)	1	93.3	95.0	6.4	9.4
(30, 150, 1)	5	94.6	94.9	2.5	3.5
(30, 150, 2)	4	94.4	94.9	2.8	4.6
(50, 150, 1)	3	96.8	96.7	7.0	8.2
(50, 150, 2)	0	-	-	-	-
(50, 250, 1)	1	97.2	96.5	2.4	2.9
(50, 250, 2)	0	-	-	-	-
(75, 225, 1)	2	98.4	97.6	7.0	7.8
(75, 225, 2)	0	-	-	-	-
(75, 375, 1)	0	-	-	-	-
(75, 375, 2)	0	-	-	-	-
Average	-	86.9	89.8	3.8	5.8

Table 7. Average solution times in seconds for the relaxations of the benchmark MILP formulation (6) and the three MISOCP reformulations (7)-(9).

(I, J, d)	Average solution time for relaxation (s)			
	MILP	MISOCP1	MISOCP2	MISOCP3
(5, 15, 1)	0.0	0.0	0.0	0.0
(5, 15, 2)	0.0	0.0	0.0	0.0
(5, 25, 1)	0.0	0.0	0.1	0.0
(5, 25, 2)	0.0	0.0	0.1	0.0
(15, 45, 1)	0.0	0.1	0.4	0.1
(15, 45, 2)	0.0	0.1	0.4	0.1
(15, 75, 1)	0.0	0.1	0.7	0.1
(15, 75, 2)	0.0	0.1	0.7	0.1
(30, 90, 1)	0.1	0.4	2.8	0.3
(30, 90, 2)	0.1	0.4	2.8	0.3
(30, 150, 1)	0.1	0.7	4.5	0.4
(30, 150, 2)	0.1	0.7	4.5	0.4
(50, 150, 1)	0.2	2.1	11.0	0.8
(50, 150, 2)	0.2	2.0	11.4	0.8
(50, 250, 1)	0.3	4.8	17.9	1.3
(50, 250, 2)	0.3	4.4	18.7	1.2
(75, 225, 1)	0.6	9.4	32.8	1.7
(75, 225, 2)	0.6	9.8	34.1	1.7
(75, 375, 1)	1.0	20.2	55.0	3.4
(75, 375, 2)	0.9	19.6	56.6	3.5

formulations. None of the five instances were solved to optimality for some parameter combinations such as $(I, J, d) = (50, 150, 2)$; this is indicated by a dash. Table 7 shows the average solution times in seconds for relaxations.

According to Tables 3-5, the reformulation MISOCP1 shows definitely the worst performance among the conic formulations. Indeed, MISOCP1 manages to find optimal solutions only for the easiest parameter combination (I,J)=(5,15). The overall average gap for $I = 15$ is about 90%, while it is less than 1.5% for both MISOCP2 and MISOCP3. In view of Tables 4 and 5, MISOCP3 shows a somewhat better performance on small instances ($I \leq 15$), whereas MISOCP2 seems more promising on large ones $I \leq 30$). Particularly, for $I = 15$, MISOCP3 has a smaller average gap in three of the four, and a smaller average time for all of the four parameter combinations. On the other hand, MISOCP2 yields notably better gaps for $I = 30$. Also, it is more successful at finding feasible solutions when $30 \leq I \leq 50$; however, for $I = 75$, MISOCP3 has found four more feasible solutions than MISOCP2 in total.

We had proved in Section 4.2 that MISOCP2 has a stronger relaxation than MISOCP1. Indeed, according to Table 6, overall average gap for the relaxation of the former is 3.8% while that of the latter is 89.8%. Moreover, in line with Mehmanchi et al. (2019), the same table shows that empirically the relaxation of MISOCP2 is also stronger than that of MISOCP3. In fact, not only on average, but in all instances we generated in our experiments, this happened to be the case.

Large gaps associated with MISOCP1 provides a partial explanation of the poor performance of this reformulation. For MISOCP3, in comparison with MISOCP2, the reduction in the number of rotated cone constraints is reflected in smaller average solution times for relaxations (Table 7), which generally compensates for the increase in the number of binary variables. Nevertheless, as Table 2 demonstrates, the benchmark MILP formulation (6) surpasses all three MISOCP reformulations. Although its relaxations have a large average gap like MISOCP1, they are solved very fast compared to even the best conic reformulation. This implies that nodes in a branch-and-bound tree for (6) are processed much more quickly, resulting in better overall performance.

We conclude this section with a comparison of formulation (6) with the best MILP formulation denoted TAPL3 in Güngör [12]. As we have noted in Section 3, formulation (6) can be regarded as the minimization version of TAPL3. Consequently, it is not surprising that the solution quality and performance of (6) demonstrated by Table 2 is generally similar to that of TAPL3. However, there are considerable differences at some specific instances. For example, regarding the combination $(I, J, d) = (75, 225, 1)$, the average gap for (6) is 4.34%, while it is 76.7% for TAPL3. This stems from actually one instance out of five, for which the gap is 19% for (6) and 373% for TAPL3 at the end of ten-minute time limit. On the other hand, concerning the triple $(I, J, d) = (75, 375, 2)$, the average gap is 48.7% for (6) and 3.97% for TAPL3. In this case, two instances end up with a gap larger than 90% for (6). These examples suggest that a seemingly trivial change of the objective sense—from maximization to minimization—may result in significant differences on the solution performance of some instances.

6. Conclusion

In this paper, we investigated three second-order cone programming reformulations of a fractional 0-1 program for task assignment that was formulated and solved in the literature before by mixed-integer linear programming. We carried out a computational study on randomly generated instances. The compact conic formulation performed the worst, whereas the extended formulation and its enhancement based on binary expansions yielded better and similar results. Nevertheless, the benchmark mixed-integer linear formulation gives the best performance for task assignment with respect to preferences.

References

Adams, W. P., & Forrester, R. J. (2005). A simple recipe for concise mixed 0-1 linearizations. Operations Research Letters, 33, 55–61. doi:10.1016/j.orl.2004.05.001

Alizadeh, F., & Goldfarb, D. (2003). Second-order cone programming. Mathematical Programming, Series B, 95, 3–51. doi:10.1007/s10107-002-0339-5

Atamtürk, A., & Gómez, A. (2020). Submodularity in conic quadratic mixed 0-1 optimization. Operations Research, 68, 609–630. doi:10.1287/opre.2019.1888

Benson, H. Y., & Sağlam, Ü. (2014). Mixed-Integer Second-Order Cone Programming: A Survey. INFORMS Tutorials in Operations Research, 13–36. doi:10.1287/educ.2013.0115

Borrero, J. S., Gillen, C., & Prokopyev, O. A. (2016). A simple technique to improve linearized reformulations of fractional (hyperbolic) 0-1 programming problems. Operations Research Letters, 44, 479–486. doi:10.1016/j.orl.2016.03.015

Borrero, J. S., Gillen, C., & Prokopyev, O. A. (2017). Fractional 0-1 programming: applications and algorithms. Journal of Global Optimization, 69, 255–282. doi:10.1007/s10898-016-0487-4

Boyd, S., & Vandenberghe, L. (2004). Convex Optimization.

Chi, C.-Y., Li, W.-C., & Lin, C.-H. (2017). Convex Optimization Problems. doi:10.1201/9781315366920-5

Glover, F. (1975). Improved Linear Integer Programming Formulations of Nonlinear Integer Problems. Management Science, 22, 455–460. doi:10.1287/mnsc.22.4.455

Güngör, M. (2019). A fractional 0-1 program for task assignment with respect to preferences. Computers and Industrial Engineering, 131, 263–268. doi:10.1016/j.cie.2019.03.048

Li, H. (1994). A global approach for general 0-1 fractional programming. European Journal of Operational Research, 73, 590–596. doi:10.1016/0377-2217(94)90257-7

Lobo, M. S., Vandenberghe, L., Boyd, S., & Lebret, H. (1998). Applications of second-order cone programming. Linear Algebra and Its Applications, 284, 193–228. doi:10.1016/S0024-3795(98)10032-0

Mehmanchi, E., Gómez, A., & Prokopyev, O. A. (2019). Fractional 0–1 programs: links between mixed-integer linear and conic quadratic formulations. Journal of Global Optimization, 75, 273–339. doi:10.1007/s10898-019-00817-7

Şen, A., Atamtürk, A., & Kaminsky, P. (2018). A conic integer optimization approach to the constrained assortment problem under the mixed multinomial logit model. Operations Research, 66, 994–1003. doi:10.1287/opre.2017.1703

Tawarmalani, M., Ahmed, S., & Nikolaos, V. (2002). Global Optimization of 0-1 Hyperbolic Programs. Journal of Global Optimization, 24, 385–416. doi:10.1023/A:1021279918708

Wu, T. (1997). A note on a global approach for general 0–1 fractional programming. European Journal of Operational Research, 101, 220–223. doi:10.1016/S0377-2217(96)00258-5