Theory and Methodology

Using approximate gradients in developing an interactive interior primal–dual multiobjective linear programming algorithm

Ami Arbel a,*, Shmuel S. Oren b

a Department of Industrial Engineering, Tel-Aviv University, Tel-Aviv 69978, Israel
b Department of Industrial Engineering and Operations Research, University of California at Berkeley, Berkeley, CA 94720, USA

Received April 1993; revised July 1994

Abstract

We present a new interactive multiobjective linear programming algorithm that is based on one variant of Karmarkar’s algorithm known as the path-following primal–dual algorithm. The modification of this single-objective linear programming algorithm to the multiobjective case is done by deriving an approximate gradient to the implicitly-known utility function. By interacting with the decision maker, locally-relevant preference information are elicited and the approximated gradient can therefore be continuously updated. The interior step direction is then generated by projecting the approximate gradient and taking an interior step from the current iterate to the new one along this projection.

Keywords: Multiobjective Linear Programming (MOLP); Multicriteria optimization; path-following primal–dual algorithm; interior-point algorithms

1. Introduction

We present a new interactive multiobjective linear programming algorithm (MOLP) that is based on one variant of Karmarkar’s algorithm known as the path-following primal–dual algorithm. The modification of this single-objective linear programming algorithm to the multiobjective case is done by deriving an approximate gradient to the implicitly-known utility function. By interacting with the decision maker (DM), locally-relevant preference information are elicited and the approximated gradient can, therefore, be continuously updated. The interior step direction is generated by projecting this approximate gradient and taking an interior step from the current iterate to the new one along this projection.

The area of Multicriteria Decision Making (MCDM) in general, and that of Multiobjective Linear Programming (MOLP) have been very active in recent years (see, e.g., [4], [10], [12]) which resulted in various solution approaches to such problems. In limiting our attention to MOLP algorithms, it is noticeable that a number of different approaches were developed over the years. While these approaches differ in the way they generate their stepping mechanism, they are mostly based on the simplex algorithm introduced in 1947 by Dantzig. With the introduction of an...
interior algorithm in 1984 by Karmarkar [5], a new class of algorithms became available for linear programming problems. Following the original paper with its interior projective algorithm, a number of additional variants were reported. The use of the so-called affine-scaling primal algorithm to MOLP problems was reported in [2 and 3]. It is our intention in this paper to present a modification of another variant, known as the path-following primal–dual algorithm [1, 6–8] to linear programming problems with multiple objectives.

The paper is arranged as follows. Section 2 describes the MOLP problem and some of the difficulties encountered in attempting its solution. Section 3 describes a way to derive an approximate gradient of the implicitly-known utility function. Section 4 uses this gradient to generate the interior search direction and develops our proposed algorithm. Section 5 illustrates it through an example and Section 6 provides a summary and concluding remarks.

2. Statement of the problem

A multiobjective linear programming (MOLP) problem is described through the following formulation:

\[
\begin{align*}
\text{max } & \quad C^Tx \\
\text{s.t. } & \quad Ax = b, \\
& \quad x \geq 0,
\end{align*}
\]

(1)

where \( A \) is an \( m \times n \) constraint matrix, \( C \) is an \( n \times q \) objective matrix, \( x \in \mathbb{R}^n \) is the solution (decision) vector, and \( b \in \mathbb{R}^m \). A naïve approach at solving the problem in (1) can be attempted by reducing the multiple objective problem to a single objective problem through a scaling operation such as shown in (2):

\[
\begin{align*}
\text{max } & \quad \lambda^TC^Tx \\
\text{s.t. } & \quad Ax = b, \\
& \quad x \geq 0,
\end{align*}
\]

(2)

where \( \lambda \in \mathbb{R}^q \) is supposed to reflect the relative importance of the \( q \) objectives. This scaling operation ignores the fact that the relative importance of the various objectives is usually dependent on the current solution point in decision space and, therefore, changes from one point to another. To use scaling coefficients one should derive them in a way that is locally-relevant and, therefore, (2) with its static set of weights does not represent a general valid solution approach to the MOLP problem.

A more realistic representation to the problem in (1) is offered by introducing a utility function over the range of decision values, \( u(x) \). With this function, the vector-valued optimization problem of (1) can be replaced by the scalar optimization problem

\[
\begin{align*}
\text{max } & \quad u(x) \\
\text{s.t. } & \quad Ax = b, \\
& \quad x \geq 0.
\end{align*}
\]

(3)

Conceptually, the problem shown in (3) can now be solved by using an iterative approach where one starts at a feasible point, and steps along the gradient of the utility function to the next iterate. Unfortunately, the assumption concerning the availability of the utility function is hardly ever justified even in small problems, let alone when one considers large-scale MOLP problems. The difficulty with this multiobjective optimization problem is, therefore, how to obtain a solution to the vector-valued optimization problem of (3) without actually evaluating the multiattribute utility function in an explicit manner.

Many approaches were developed over the last 20 years to address the MOLP problem of (1). Regardless of the specific algorithm, one has to provide the means to generate a step that moves the current iterate from its present position to the next one. When preference for the various possible directions is elicited by interacting with the Decision Maker, we refer to this class of algorithms as an interactive MOLP algorithm. Until recently, the stepping mechanism was mostly done through the simplex algorithm. With the introduction of Karmarkar's algorithm in 1984 a new class of algorithms became available for MOLP problem. The use of the affine-scaling primal algorithm for such problems was reported in [2] and [3]. Here we attempt to use another interior-point algorithm and a new step generation mechanism.
Using an interior algorithm for solving MOLP problems can be done in more ways than one. One class of approaches is offered by considering each of the individual cost vectors and use an interior algorithm to generate individual step direction vectors that optimize each of the single objectives separately. Then, somehow, use these individual step directions to derive a single combined step along which to move to the next iterate. We describe this general approach next.

Letting the objective matrix, \( C \), be written as \( C = [c_1, c_2, \ldots, c_q] \), we apply the interior algorithm of our choice by considering each column of the objective matrix separately. That is, we consider \( q \) single-objective problems to generate \( q \) step direction vectors. For example, when using the affine-scaling primal algorithm, the step direction vector, \( dx_i \), that provides the ascent direction for the \( i \)-th objective given by

\[
 dx_i = \alpha_i \nabla c_i(x),
\]

is obtained by stepping in the direction of the scaled gradient projected on the null space of the scaled constraints matrix \( A \). When using the path-following primal–dual algorithm, as we propose in this paper (see the Appendix for more details), the \( i \)-th primal problem corresponding to the \( i \)-th objective vector is given by

\[
 \min \ c_i^T x \\
 \text{s.t.} \quad Ax = b, \\
 \quad x \geq 0,
\]

and adding the slack variables, its corresponding dual problem is given by

\[
 \max \ b^T y \\
 \text{s.t.} \quad A^T y + z^i = c^i, \\
 \quad z^i \geq 0, \quad 1 \leq i \leq q.
\]

For these \( q \) pairs of primal and dual problems, the set of step direction vectors \( \{dx^i\} \) is generated from

\[
 dx^i = Z_i^{-1} w^i(\bar{\mu}_i) - Z_i^{-1} X d z^i, \quad 1 \leq i \leq q,
\]

and where

\[
 dy^i = -(A Z_i^{-1} X A^T)^{-1} A Z_i^{-1} w^i(\bar{\mu}_i), \quad (7)
\]

\[
 dz^i = -A^T dy^i. \quad (8)
\]

The vector \( w^i(\bar{\mu}_i) \) is given through

\[
 w^i(\bar{\mu}_i) = \bar{\mu}_i e - X Z_i e, \quad w^i(\bar{\mu}) \in \mathbb{R}^n,
\]

the barrier parameter \( \bar{\mu}_i \) is fixed through

\[
 \bar{\mu}_i = \sigma e^T X Z_i e / n,
\]

where \( 0 < \sigma < 1 \), and where \( X \) and \( Z_i \) are diagonal \( n \times n \) matrices whose diagonal is composed of the respective components of the current iterates of \( x \) and \( z^i (1 \leq i \leq q) \).

Once the set \( \{dx^i\} (1 \leq i \leq q) \) of \( q \) interior step direction vectors for each of the objectives is derived, the next question is associated with combining them somehow to generate an interior single step direction vector, \( dx \), along which one steps from the current iterate toward the next iterate. We refer to this phase as the step generation phase. One approach for generating this combined step vector is available by using the individual step direction vectors in a convex combination given through

\[
 dx = \sum_{i=0}^{q} \lambda_i \ dx_i, \quad (11)
\]

where

\[
 \sum_{i=0}^{q} \lambda_i = 1, \quad \lambda_i > 0, \quad (12)
\]

and where the scaling coefficients \( \{\lambda_i\} \) are supposed to reflect the locally-relevant preference for each of the directions offered through \( \{dx^i\} \). Another approach is offered by trying to find an approximation to the gradient of the implicitly-known utility function and use it to generate the interior step direction along which to step to the next interior iterate. We discuss this issue next and develop it into our proposed algorithm.

3. Approximate gradients

The vector-valued optimization problem in (1) is ambiguous since usually the objectives are con-
flitting and pursuing the optimum with respect to each objective will lead to different solutions. This ambiguity may be resolved by introducing a utility function $u(x)$. This function, defined in decision (solution) space, can be replaced by a real-valued function $U(v)$ defined over the space of objectives. If $u(v)$ was explicitly available we could have used the fact that $v = C^T x$ (that is, $v_i = c_i^T x$ where $1 \leq i \leq q$) to obtain a real-valued function $u(x)$ and use is to arrive at the true optimal solution vector, $x$, by solving the non-linear optimization problem shown by (3). Since the dimension of $v$ is significantly smaller than that of $x$, it is cheaper to elicit preference information about $U(v)$ in the objective space, than about $u(x)$ in the decision space. Therefore, the problem we address in this paper is in the form shown by

$$\max \ u[v_1(x), v_2(x), \ldots, v_q(x)]$$
\[ (13) \]

subject to

$$Ax = b,$$
\[ x \geq 0, \]

and where the value of the $i$-th objective function is given through $v_i = c_i^T x$.

To consider the multiobjective optimization problem we have to find a way to approximate the utility function at the current iterate, and take a step in a direction that results in an improvement of the value of the objective function. Since the utility function depends on the objective functions, we have

$$u(v_1, v_2, \ldots, v_q) = u(c_1^T x, c_2^T x, \ldots, c_q^T x)$$
\[ (14) \]

and from this, the gradient with respect to the solution vector is found as

$$\nabla_u u = \frac{\partial u}{\partial x} = \frac{\partial u}{\partial v_1} \frac{\partial v_1}{\partial x} + \ldots + \frac{\partial u}{\partial v_q} \frac{\partial v_q}{\partial x}$$
\[ = \frac{\partial u}{\partial v_1} c_1^T + \ldots + \frac{\partial u}{\partial v_q} c_q^T, \]
\[ (15) \]

which, in matrix form, is written as

$$\nabla_u u = (\nabla_v u) C^T.$$  
\[ (16) \]

To find the approximate gradient in decision space, we have to evaluate the gradient of the utility function in the objective space, that is, find $\nabla_v u$.

Considering each of the $q$ objective functions by themselves, results in stepping from the current iterate $x_0$, along $q$ specific step direction vectors, $dx_i$, to $q$ end points with their own specific values for the $q$ objective functions. Letting $\Delta v_{ij}$ denote the change in the value of the $i$-th objective function that results from stepping away from the initial point $x_0$ along the step direction vector $dx_i$, the change in the utility function, initial point $u(x)$, in stepping from the initial point $x_0$ to a set of $q$ new iterates $\{x_i\}$ can be approximated through a first order Taylor's expansion as follows:

$$u(x^1) = u(x_0) + \frac{\partial u}{\partial v_1} \Delta v_{11} + \frac{\partial u}{\partial v_2} \Delta v_{21} + \ldots + \frac{\partial u}{\partial v_q} \Delta v_{q1},$$
\[ (17) \]

$$u(x^2) = u(x_0) + \frac{\partial u}{\partial v_1} \Delta v_{12} + \frac{\partial u}{\partial v_2} \Delta v_{22} + \ldots + \frac{\partial u}{\partial v_q} \Delta v_{q2},$$
\[ \vdots \]

$$u(x^q) = u(x_0) + \frac{\partial u}{\partial v_1} \Delta v_{1q} + \frac{\partial u}{\partial v_2} \Delta v_{2q} + \ldots + \frac{\partial u}{\partial v_q} \Delta v_{qq}. \]
\[ (17) \]

In matrix form, these $q$ approximations are written as

$$\Delta u = \Delta[u(x)] = (\nabla_v u)(\Delta V),$$
\[ (18) \]

where the $i$-th component of the vector $\Delta u$ is given by $u(x^i) - u(x_0)$, the $q$-dimensional row vector $\nabla_v u$ is the gradient of the utility function defined by

$$\nabla_v u = \left[ \frac{\partial u}{\partial v_1}, \frac{\partial u}{\partial v_2}, \ldots, \frac{\partial u}{\partial v_q} \right].$$
\[ (19) \]
and the \( q \times q \) matrix \( \Delta V \) is given by

\[
\Delta V = \begin{bmatrix}
\Delta v_{11} & \Delta v_{12} & \cdots & \Delta v_{1q} \\
\Delta v_{21} & \Delta v_{22} & \cdots & \Delta v_{2q} \\
\vdots & \vdots & \ddots & \vdots \\
\Delta v_{q1} & \Delta v_{q2} & \cdots & \Delta v_{qq}
\end{bmatrix},
\]

(20)

From this approximation, the gradient of the utility function with respect to the objective vector, \( v \), is given through

\[
\nabla_v u = \Delta u (\Delta V)^{-1}.
\]

(21)

Since the changes in the values of the individual objective functions are given by

\[
\Delta v_i = e_i^T (x_0 + \rho \alpha_j \text{d}x^i) - e_i^T x_0 = \rho \alpha_j e_i^T \text{d}x^i,
\]

(22)

we can now express the \( q \times q \) matrix \( \Delta V \) directly as

\[
\Delta V = C^T \left[ \rho \alpha_1 \text{d}x^1, \rho \alpha_2 \text{d}x^2, \ldots, \rho \alpha_q \text{d}x^q \right],
\]

(23)

where \( \text{d}x^i \) is the set of \( q \) individual step direction vectors derived through (6) and where \( \{\rho \alpha_j\} \) is the set of step sizes used in deriving the set of \( q \) interior new iterates through

\[
x^i = x_0 + \rho \alpha_j \text{d}x^i, \quad 1 \leq i \leq q.
\]

With the value of the gradient of the utility function in objective space given through (21), we now return to the expression of the gradient in the solution space. Doing so results in

\[
\nabla_x u = (\nabla_v u) C = \Delta u (\Delta V)^{-1} C^T.
\]

(24)

### 4. The proposed MOLP algorithm

In this section we consider the question of generating interior search directions and combining them into a single step direction vector along which we step from the current iterate to the next interior iterate. The combined step is derived by using the results established in the previous section for the approximated gradient.

The Taylor’s series approximation of (17) for the implicitly-known utility function involves the value of the utility function at the current iterate \( x_0 \) as well as the values at the set of \( q \) new interior iterates \( \{x^i\} \). In the absence of the true utility function, these values are unavailable and have to be expressed through some proxy measures. There is no single best way for doing it and we choose here to use the Analytic Hierarchy Process (AHP) to assess relative preference for the value vectors \( \{v_i\} \) associated with the initial solution point \( x_0 \) as well as the \( q \) new iterates \( \{x^i\} \). For details associated with priority derivation using the AHP, the reader is referred to [9].

To obtain an approximate measure for the utility function at the \( q \) points of interest, we proceed as follows. While the value of the utility function at the points of interest is unknown, we can still evaluate the complete \( q \)-dimensional vector of objective function values, \( v(x^i) = C^T x^i \), at each of the \( q \) points \( \{x^i\} \). Together with the value vector at the current iterate, \( x_0 \), we have information about \( q + 1 \) points in objective space which we present to the DM in order to obtain a measure of relative preference for these end points. This step can be done by using the AHP and a \((q + 1) \times (q + 1)\) comparison matrix that provides the relative preference of these points through its principal eigenvector. This relative preference is used to provide an approximate measure for the vector \( \Delta u \) used in (18). The relation between the priority vector, \( p \), and the utility values at the \( q + 1 \) interior iterates \( \{v_0, v_1, \ldots, v_q\} \) is through the component-wise relation given by

\[
\frac{u_i}{p_i} = \frac{u_j}{p_j}
\]

for all \( 0 \leq i \leq q + 1 \) and \( 0 \leq j \leq q + 1 \).

This results in the row vector, \( \Delta u \), being approximated through

\[
\Delta u = \beta \Delta p = \beta [p_1 - p_0, \ldots, p_q - p_0]
\]

(26)

for some (yet unknown) scalar \( \beta \), where \( p_i \) \((i = 0, 1, \ldots, q)\) are the priorities of the \( i \)-th iterate as derived by using the AHP.

Using the expression for \( \Delta u \) as shown in (26) we proceed next to derive the approximated gra-
Gradient $\nabla u$ as shown in the previous section. Once the approximate gradient is available, we have to generate the next iterate and move from our current position given by $x_0$ to a new iterate by stepping along a single direction. We next discuss how to use our knowledge of the approximated gradient to derive this step direction vector.

Using the approximation for the utility function through the approximate gradient, we have

$$u(x) = u(x_0) + (\nabla_x u) dx,$$

where $dx$ is the required combined step direction vector to be used in making the actual next step. The approximate problem for determining $dx$ is then given through

$$\max \; u(x_0) + (\nabla_x u) dx$$

subject to $A(x_0 + dx) = b$, $x > 0$, and since the current iterate, $x_0$, is feasible, this is equivalent to

$$\max \; (\nabla_x u) dx$$

subject to $A dx = 0$, $x_0 + dx > 0$.

We can use the idea employed in deriving the affine-scaling primal algorithm to find the required step direction vector $dx$. The difference, however, is that while in the primal algorithm one projects the scaled cost vector, $Dc$, on the null space of the scaled constraints matrix, $AD$, we now project that approximated gradient, $\nabla u$, instead. Therefore, the required step direction vector, $dx$, is now obtained from

$$dx = D^2 \left[ (\nabla_x u)^T - A^T (AD^2 A^T)^{-1} AD^2 (\nabla_x u)^T \right].$$

Once the combined step vector, $dx$, is available, we perform the ratio test to determine the step size, $\alpha$, and take the next interior step according to the updating formula given by

$$x = x_0 + \rho \alpha \; dx, \quad 0 < \rho < 1.$$  

Note that by determining the step size, $\alpha$, directly through the ratio test that takes us all the way to the boundary, we eliminate the need for the unknown constant $\beta$ in (26).

**Summary:** An interior path-following primal–dual multiobjective linear programming algorithm

**Step 1.** Set the iteration counter, $k$, at $k = 0$. Initialize the solution vectors through a set of feasible solution to the primal and dual problem $x(k) = x_0$, $y(k) = y_0^i$, and $z(k) = z_0^i$, where $x_0 > 0$, $Ax_0 = b$, and, in addition,

$$A^T y_0^i + z_0^i = c_0^i \quad 0 \leq i \leq q.$$

**Step 2.** Define the scaling matrices $X$ and $Z_i$ $(0 \leq i \leq q)$ through

$$X = \text{diag}\left[x_1(k), x_2(k), \ldots, x_q(k)\right], \quad Z_i = \text{diag}\left[z_1^i(k), z_2^i(k), \ldots, z_n^i(k)\right],$$

where $x_i(k)$ and $z_i^j(k)$ are the $j$-th components of the current solution vectors $x(k)$ and $z(k)$ respectively.

**Step 3.** Solve for the set of individual step direction vectors $(dx^i)$, $(dy^i)$, and $(dz^i)$ $(0 \leq i \leq q)$ from

$$dy^i = -\left(AZ_i^{-1}X^T\right)^{-1}AZ_i^{-1}w^i(\bar{\mu}_i),$$

$$dz^i = -A^T dy^i,$$

$$dx^i = Z_i^{-1}w^i(\bar{\mu}_i) - Z_i^{-1}X dz^i,$$

where the vector $w^i(\bar{\mu}_i) \in \mathbb{R}^n$ and the barrier parameter, $\bar{\mu}_i$, are given through

$$w^i(\bar{\mu}_i) = \frac{\sigma e^T XZ_i e}{n}, \quad 0 < \sigma < 1.$$

**Step 4.** Derive the set of new iterates $(x^i)$, $(y^i)$, and $(z^i)$ $(0 \leq i \leq q)$ from

$$x^{i}(k + 1) = x(k) + \rho \alpha_p^i dx^i, \quad 0 < \rho < 1,$$

$$y^{i}(k + 1) = y^{i}(k) + \rho \alpha_y^i dy^i,$$

$$z^{i}(k + 1) = z^{i}(k) + \rho \alpha_z^i dz^i,$$

where the step sizes are found from the ratio tests administered through

$$\alpha_p^i = \min \left\{ \frac{-x_j^i(k + 1)}{dx_j^i(k + 1)} : \forall \; dx_j^i(k + 1) < 0, \quad 1 \leq j \leq n \right\}, \quad 0 \leq i \leq q,$$

$$\alpha_d^i = \min \left\{ \frac{-z_j^i(k + 1)}{dz_j^i(k + 1)} : \forall \; dz_j^i(k + 1) < 0, \quad 1 \leq j \leq n \right\}.$$
Step 5. Construct a \((q + 1) \times (q + 1)\) comparison matrix for deriving relative preference, \(p\), for the current iterate, \(x(k)\) and the \(q\) new iterates \(\{x'(k+1)\}\). Use this vector of priorities \(p\), to find the \((q+1)\)-dimensional vector \(\Delta u\) given through
\[
\Delta u = [p_1 - p_0, p_2 - p_0, \ldots, p_q - p_0],
\]
where \(p_i\) is the \(i\)-th component of the \((q+1)\)-dimensional vector \(p\).

Step 6. Evaluate the value matrix, \(\Delta V\) through
\[
\Delta V = C[p_1 \alpha_1, p_2 \alpha_2, \ldots, p_q \alpha_q, d x^1, \ldots, d x^q].
\]
Next, find the approximate gradient \(\nabla_u\) according to
\[
\nabla_u = (\nabla_u) C^T = \Delta p (\Delta V)^T \Delta V^{-1}.
\]

Step 7. Find the combined step direction vector, \(d x\), and the new iterate, \(x(k+1)\), from
\[
d x = D^2 [(\nabla_u)^T - A^T (AD^2 A^T)^{-1} AD^2 (\nabla_u)^T],
\]
where the scaling matrix is defined through
\[
D = \text{diag}[x_1(k), x_2(k), \ldots, x_q(k)].
\]
Find the new interior iterate \(x(k+1)\) from
\[
x = x + \rho \alpha d x, \quad 0 < \rho < 1,
\]
where \(\alpha\) is determined through a ratio test.

Step 8. Find the candidate for a new boundary point, \(\hat{x}\), by taking a full step (\(\rho = 1\)) along \(d x\).
\[
\hat{x} = x(k) + \alpha d x.
\]
In subsequent iterations, if the new value vector at this point \(\hat{v} = C \hat{x}\) is preferred to that identified for any previous boundary point, replace the old (stored) boundary point with the new point.

Step 9. If at Step 5, all the components of the vector \(\Delta u\) are negative, no new direction is preferred to the current one. In this case terminate the iterative process at the boundary point. Otherwise, increment the iteration counter, \(k := k + 1\). GoTo Step 2.

Note that the use of Taylor’s series approximation for the value of the utility function should restrict the algorithm to use relatively small step size factors that will not invalidate the approximation.

5. An illustrative example

We demonstrate the algorithm proposed in this paper through a numerical example. Consider the MOLP problem given by
\[
\begin{align*}
\text{max } & x_1
\end{align*}
\begin{align*}
\text{max } & x_2
\end{align*}
\begin{align*}
\text{s.t. } & x_1 + 5x_2 \leq 41,
& 2x_1 + 3x_2 \leq 33,
& 4x_1 + x_2 \leq 41,
& x_1 - 2x_2 \leq 8,
& x_1 + x_2 \geq 2,
& -4x_1 + x_2 \leq 4,
& x_1, x_2 \geq 0
\end{align*}
\]
For this example, an initial set of primal and dual solution vectors is available through
\[
\begin{align*}
x_0 &= [2, 1, 34, 26, 32, 8, 11]^T,
y_0 &= [-1, -1, -1, -1, -1]^T,
z_A &= [2, 3, 1, 1, 1, 1, 1]^T,
z_B &= [3, 2, 1, 1, 1, 1, 1]^T,
\end{align*}
\]
where
\[
y_i^0 = y_0, z_i^0 = c_i - A^T y_0, \quad i = 1, 2,
\]
and where the cost vectors are negated to reflect a minimization problem. Assuming next that the DM’s utility function is given through
\[
u(x) = (x_1 + 4)(x_2 + 1),
\]
this vector optimization problem has a unique solution given through
\[
x^* = [7.0, 6.3, 2.3, 0.0, 6.7, 13.7, 11.3, 25.7]^T,
u^* = 80.6.
\]

In order to test our proposed algorithm we use the utility function to provide the relative preference for the interior step direction vectors. Following the steps outlined above we generate a sequence of steps summarized in Table 1. For
Table 1
Solution results ($\rho = 0.4$)

<table>
<thead>
<tr>
<th>$k$</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$u$</th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$u$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2.0000</td>
<td>1.0000</td>
<td>12.0000</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>1.4337</td>
<td>3.1349</td>
<td>22.4677</td>
<td>0.5843</td>
<td>6.3372</td>
<td>33.6357</td>
</tr>
<tr>
<td>2</td>
<td>1.7365</td>
<td>4.9857</td>
<td>34.3369</td>
<td>2.1908</td>
<td>7.7618</td>
<td>54.3425</td>
</tr>
<tr>
<td>3</td>
<td>2.0579</td>
<td>6.0682</td>
<td>42.8182</td>
<td>2.5399</td>
<td>7.6920</td>
<td>56.8446</td>
</tr>
<tr>
<td>4</td>
<td>2.4388</td>
<td>6.6801</td>
<td>49.4506</td>
<td>3.0102</td>
<td>7.5980</td>
<td>60.2733</td>
</tr>
<tr>
<td>5</td>
<td>3.1130</td>
<td>6.9581</td>
<td>56.6061</td>
<td>4.1243</td>
<td>7.3751</td>
<td>68.0421</td>
</tr>
<tr>
<td>6</td>
<td>4.4440</td>
<td>6.8574</td>
<td>66.3479</td>
<td>6.4405</td>
<td>6.7063</td>
<td>80.4580</td>
</tr>
<tr>
<td>7</td>
<td>5.4883</td>
<td>6.6332</td>
<td>72.4258</td>
<td>7.0546</td>
<td>6.2969</td>
<td>80.6647</td>
</tr>
<tr>
<td>8</td>
<td>5.8972</td>
<td>6.6437</td>
<td>75.6518</td>
<td>7.0546</td>
<td>6.2969</td>
<td>80.6647</td>
</tr>
<tr>
<td>9</td>
<td>6.2179</td>
<td>6.5999</td>
<td>77.6547</td>
<td>7.0546</td>
<td>6.2969</td>
<td>80.6647</td>
</tr>
<tr>
<td>10</td>
<td>6.4144</td>
<td>6.5708</td>
<td>78.8455</td>
<td>7.0546</td>
<td>6.2969</td>
<td>80.6647</td>
</tr>
</tbody>
</table>

For brevity we show only the first two components of the decision vector and the best boundary point identified during the iterative process. In addition, we also show the value of the utility function at these two points. The sequence of iterates shown in Table 1 was generated using a step size factor of 40% ($\rho = 0.4$). The utility value at the current iterate as well as the values at the currently best boundary point are shown in Fig. 1.

Note that since a new boundary point becomes available at each iteration, we have a candidate for updating the current set of anchor points with a new point if it is more preferred to any of the current anchors. This relative preference of the new boundary point relative to the existing anchors is done in this example by directly using the utility function. When the utility value is unavailable, the DM is asked to provide for this relative preference.

6. Summary and conclusions

A new algorithm for linear programming problems with multiple objectives has been presented. The algorithm is based on modifying the interior path-following primal–dual algorithm to MOLP problems. The modification is based on finding an approximation to the gradient of the implicitly-known utility function. By interacting with the DM, locally-relevant preference concerning the different interior directions can be obtained and used in deriving the approximated gradient. Projecting the approximated gradient provides an interior direction along which to take a step from the current iterate toward the next one. Worthy subjects for future work in this area should consider the choice of interior step size as well as the incorporation of information available on boundary points into the process.

Appendix. The path-following primal–dual algorithm

We provide a brief review of the so-called path-following primal–dual algorithm (see, e.g., [5], [6], [7]) that forms the basis for our proposed
MOLP algorithm. In developing the algorithm we consider a primal problem \((P)\) in standard form given through

\[
\begin{align*}
\text{(P)} & \\
\text{min } & c^T x \\
\text{subject to } & Ax = b, \\
& x > 0,
\end{align*}
\]

where \(x \in \mathbb{R}^n\) and \(b \in \mathbb{R}^m\). By adding slack variables, its dual problem \((D)\) is defined by

\[
\begin{align*}
\text{(D)} & \\
\text{max } & b^T y \\
\text{subject to } & A^T y + z = c, \\
& z > 0,
\end{align*}
\]

where \(y \in \mathbb{R}^m\) and \(z \in \mathbb{R}^n\).

The non-negativity constraints on the primal vector can be eliminated by adding a barrier term to the objectives of the problems in \((A.1)-(A.2)\). This results in

\[
\begin{align*}
\text{(P)} & \\
\text{min } & c^T x - \mu \sum_{i=1}^{n} \ln(x_i) \\
\text{subject to } & Ax = b, \\
& x > 0. \\
\text{(D)} & \\
\text{max } & b^T y + \mu \sum_{i=1}^{n} \ln(z_i) \\
\text{subject to } & A^T y + z = c. \\
\end{align*}
\]

Letting \(e\) be the vector of all 1's and \(X\) and \(Z\) be \(n \times n\) diagonal matrices where

\[
X = \text{diag}(x_i), \quad Z = \text{diag}(z_i),
\]

and using the Lagrangian for the dual problem (identical results are obtained when using the Lagrangian for the primal problem) leads to a set of first order necessary conditions for the optimal solution given by

\[
\begin{align*}
\frac{\partial L_d}{\partial x} &= 0 = A^T y + z - c, \\
\frac{\partial L_d}{\partial y} &= 0 = b - A x, \\
\frac{\partial L_d}{\partial z} &= 0 = \mu Z^{-1} e - x.
\end{align*}
\]

After some manipulations, the conditions for both the primal and dual problems become

\[
\begin{align*}
X z e &= \mu e, \\
A x &= b, \\
A^T y + z &= c.
\end{align*}
\]

Note that condition \((A.12)\) maintains primal feasibility and \((A.13)\) maintains dual feasibility. In addition, from condition \((A.11)\) we have \(x_i z_i = \mu \) for all \(1 \leq i \leq n\).

By fixing \(\mu\) we can solve \((A.11)-(A.13)\) for \(x\), \(y\) and \(z\). Since these vectors are dependent on the choice of the barrier parameter, \(\mu\), we get a family of solutions. The central trajectory, or the central path, is defined as the set of all vectors \(x(\mu), y(\mu)\) and \(z(\mu)\), satisfying \((A.13)-(A.15)\), and the resulting primal–dual algorithm summarized here belongs, therefore, to the class of algorithms known as path-following algorithms.

Assume now that the \(m \times n\) matrix \(A\) is of full row rank \(m\) and that starting feasible and interior vectors \(x > 0, y \) and \(z > 0\) for the problems \((P)\) and \((D)\) in \((1)\) and \((2)\) are available. Then, for a given \(\mu > 0\) we have to find step direction vectors \(dx, dy\) and \(dz\) that take us from the current feasible solution and move us to a new iterate...
while satisfying the necessary conditions shown in (A.11)–(A.13). Using Newton’s method we have
\[
\begin{pmatrix}
Z & O & X \\
A & O & O \\
O & A^T & I_n
\end{pmatrix}
\begin{pmatrix}
dx \\
dy \\
dz
\end{pmatrix}
= 
\begin{pmatrix}
\mu e - XZe \\
0 \\
0
\end{pmatrix}.
\] (A.15)

Defining an auxiliary vector \( w(\bar{\mu}) \) through
\[
w(\bar{\mu}) = \bar{\mu} e - XZe, \quad w(\bar{\mu}) \in \mathbb{R}^n,
\] (A.16)
and fixing the barrier parameter \( \mu \) through
\[
\bar{\mu} = \sigma \frac{e^T XZe}{n} = \sigma \frac{\sum_{i=1}^n x_i z_i}{n},
\] (A.17)
where \( 0 < \sigma < 1 \), the equations in (A.15) lead to the following relations for the required step direction vectors:
\[
dy = - (AZ^{-1}X^T)^{-1}AZ^{-1}w(\bar{\mu}),
\] (A.18)
\[
dz = -A^T dy,
\] (A.19)
\[
dx = Z^{-1}w(\bar{\mu}) - Z^{-1}X dz.
\] (A.20)

With these results for the step direction vectors, the new iterates are given as
\[
x = x_0 + \rho \alpha_p \ dx,
\] (A.21)
\[
y = y_0 + \alpha_d \ dy,
\] (A.22)
\[
z = z_0 + \alpha_d \ dz,
\] (A.23)
where the step sizes \( \alpha_p \) and \( \alpha \) are found from the ratio tests given through
\[
\alpha_p = \min \left\{ - \frac{x_i(k)}{dx_i(k)} : \forall \ dx_i(k) < 0, 1 \leq i \leq n \right\},
\] (A.24)
\[
\alpha_d = \min \left\{ - \frac{z_i(k)}{dz_i(k)} : \forall \ dz_i(k) < 0, 1 \leq i \leq n \right\},
\] (A.25)
and where \( 0 < \rho < 1 \) is a step size factor that keep the new iterates interior.

References