Convex Separable Optimization Is Not Much Harder than Linear Optimization

DORIT S. HOCHBAUM AND J. GEORGE SHANTHIKUMAR

University of California, Berkeley, Berkeley, California

Abstract. The polynomiality of nonlinear separable convex (concave) optimization problems, on linear constraints with a matrix with "small" subdeterminants, and the polynomiality of such integer problems, provided the integer linear version of such problems is polynomial, is proven. This paper presents a general-purpose algorithm for converting procedures that solves linear programming problems with or without integer variables, to procedures for solving the respective nonlinear separable problems. The conversion is polynomial for constraint matrices with polynomially bounded subdeterminants. Among the important corollaries of the algorithm is the extension of the polynomial solvability of integer linear programming problems with totally unimodular constraint matrix, to integer-separable convex programming problems. An algorithm for finding an ϵ-accurate optimal continuous solution to the nonlinear problem that is polynomial in $\log(1/\epsilon)$ and the input size and the largest subdeterminant of the constraint matrix is also presented. These developments are based on proximity results between the continuous and integral optimal solutions for problems with any nonlinear separable convex objective function. The practical feature of our algorithm is that is does not demand an explicit representation of the nonlinear function, only a polynomial number of function evaluations on a prespecified grid.

Categories and Subject Descriptors: F.2.0 **[Analysis of Algorithms and Problem Complexity]**: General; G.2.2 **[Discrete Mathematics]**: Graph Theory—*network problems*

General Terms: Algorithms, Decision, Theory

Additional Key Words and Phrases: Nonlinear optimization, proximity results, scaling algorithms

1. *Introduction*

We consider in this paper the nonlinear minimization (maximization) problem in either integer variables or in continuous variables: $\text{Min}\{\sum_{i=1}^{n} f_i(x_i) \mid A\mathbf{x} \geq \mathbf{b}\}$. The f_i's are convex (concave) functions (with no additional properties assumed). The polyhedron $\{\mathbf{x} \mid A\mathbf{x} \geq \mathbf{b}\}$ is bounded, or alternatively, if unbounded, we assume that there is a known bound on the optimal solution, which, when incorporated into the constraint set, will convert it into a (bounded) polytope. A is an $m \times n$ integer matrix, and $\mathbf{b}$ is an integer vector of dimension m. We denote the maximum absolute value of the subdeterminants of A by Δ.

D. S. Hochbaum was supported in part by the Office of Naval Research under grant N00014-88-K-0377 and by the National Science Foundation under grant ECS-85-01988. J. G. Shanthikumar was supported in part by Air Force grant AFOSR-84-0205.

Authors' address: School of Business Administration and IEOR Department, University of California, Berkeley, 350 Barrows Hall, Berkeley, CA 94720.

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.
© 1990 ACM 0004-5411/90/1000-0843 $01.50

The purpose of this paper is to develop efficient solution methods for this class of problems. In this regard, we present:

(1) A polynomial algorithm for the continuous problem. The polynomiality is in the input size, the logarithm of the accuracy required in the solution and Δ.
(2) A polynomial algorithm for the integer nonlinear problem over a polytope with its Δ polynomially bounded. In particular,
   (a) A polynomial algorithm for integer nonlinear optimization over polytopes defined by totally unimodular matrices.
   (b) A polynomial (and very simple) algorithm for nonlinear integer optimization in fixed number of variables. This is an extension to the nonlinear case and a simplification of the linear case, of Lenstra's result [12], when Δ is bounded.

For many nonlinear optimization problems, our algorithms consist of the first polynomial algorithms devised. Since it is perceived that solving nonlinear programs is much harder than solving linear programs, most continuous problems are formulated as linear programs, while a nonlinear formulation would have been more appropriate. Having shown that nonlinear separable problems can be solved without much additional effort compared to the linear programs, we can expect the use of nonlinear formulations to become more prevalent.

Our analysis constitutes a novel venture in constrained optimization of arbitrary functions. The length of the input involving such functions is not well defined, as a complete representation might require infinite number of bits. In our algorithm, there is no need to provide any representation of the functions in the objective. It suffices to have an oracle that will compute the function values at points on a polynomial grid, and this oracle will be called only polynomially many times.

To date, there are no finite algorithms known to produce the optimal continuous solutions to such problems. Indeed, no such algorithms can exist as the description of the output alone could be infinite (consider minimizing $x^3 - 6x$ subject to $x \geq 0$). For any desired finite precision of the solution, the algorithm described in this paper delivers such solution in time polynomial in the data and in the required precision.

The issue of boundedness of the solution vector is critical in the analysis of nonlinear optimization. Once the polyhedron is unbounded, the optimal solution vector may still be bounded (i.e., there is a cube of finite size that contains the origin and the optimal solution), but there is no known function of the input that constitutes a bound on the solution vector. This is in contrast to the linear case where either the polyhedron and the solution vector are unbounded or there is a polynomial length bound on the finite optimal solution vector (see, e.g., [19, p. 30 and p. 320]). Such a bound plays a critical role in polynomial algorithms for linear programming. The condition that the polyhedron is bounded can be easily verified using linear programming.

Our method of analysis relies on sensitivity results on the proximity between integer and continuous solutions for separable nonlinear programming. These results can be viewed as extensions of the proximity results derived for linear programming [1], and for separable quadratic programming [5]. We also make use of proximity results between the continuous solutions for two different piecewise linear approximations of the nonlinear objective. Finally, we make use of the strongly polynomial algorithm for linear programming with Δ of polynomial length, by Tardos [21].

Since the 0–1 version of the nonlinear problem is always linear, it seems that one would want to convert the nonlinear integer problem to a 0–1 problem. A straightforward replacement of each integer variable by a sum of binary variables results in an exponential representation, since the number of variables ought to be equal to the length of the cube bounding the value of the solution, B; thus resulting in a formulation with a large number of variables. It would then appear that an alternative would be to replace each variable by the binary representation, that is, only a logarithmic number of variables, $x_i = L_i + \sum_{j=1}^{\log_2 B} 2^j x_{ij}$, where L_i is a known lower bound on the ith component of the solution $\mathbf{x}^* = (\mathbf{x}_1^*, \ldots, \mathbf{x}_n^*)$. This approach however, will change critically the constraint matrix in a way that may convert a polynomially solvable (linear) integer programming problem into an NP-complete one.

For the continuous problem, it seems that one could use piecewise linear approximation of the objective function and formulate it as a linear programming problem. If the break intervals (in which the function is linear), used for the piecewise linear approximation, are small, one would end up with a large number of such intervals and hence a large number of variables. On the other hand, if those intervals are large, the accuracy of the linear programming solution may not conform to the required accuracy.

Our algorithm can in fact be viewed as an efficient way of using the idea of piecewise linear approximation of the objective, while controlling the number of variables to be polynomially bounded, and simultaneously guaranteeing the accuracy of the final solution.

1.1 Literature Review. There is an extensive literature on nonlinear programming problems of the type considered here. Such nonlinear programming problems appear in the design and control of stochastic systems, in image processing and elsewhere. Hoffman and Wolfe [9] proposed an algorithm for unimodal nonlinear problems on integers with two variables. Their algorithm is not guaranteed to run within certain bounded complexity. It applies, however, to unimodal functions that are a generalization of convex functions. McCallum [14] presented a heuristic for a special class of quadratic separable problems with a single constraint in nonnegative integer variables. This particular class of problems is actually solvable in polynomial time using our procedure. Minoux [15, 16] presents an algorithm for solving capacitated quadratic separable network flow problems in polynomial time. His procedure can be viewed as a special case of ours, though his presentation relies heavily on the Edmonds and Karp algorithm [3]. Edmonds and Karp scaling algorithm could be thought of as an application of a proximity result between the scaled and the original network flow problem, in which case the general purpose algorithm presented here applies immediately. The general nonlinear proximity result (in Section 3) is sufficient though to obtain the polynomial algorithm for any convex and separable objective, and in particular of course, to the quadratic case.

Laughhunn [11] uses an explicit enumeration approach for solving a binary integer programming problem with a quadratic convex objective. Note that, since the variables in that problem are binary, this problem is immediately reducible to a linear binary integer programming problem with a quadratic increase in the number of variables (cross products are also represented by binary variables, and one constraint is added for each). It is not clear whether Laughhunn's approach offers any computational advantage compared to solving the linear reduced version. A recent paper [8] considers a *nonseparable* quadratic integer problem with

transportation constraints. They solve a related quadratic continuous problem and derive the integer solution using a certain rounding property. The rounding property can be viewed as a tight proximity result for that particular nonseparable problem.

As for the literature on continuous solutions to nonlinear objective and linear constraints problems, there has been a large body of papers on the strictly convex and separable problem with a single constraint $\text{Min}\{\sum_{i=1}^{n} f_i(x_i) \mid \sum_{i=1}^{n} x_i = C, x_i \geq 0, i = 1, \ldots, n\}$. This type of problem is immediately solvable in polynomial time (the matrix is totally unimodular) for the integer case, and in $\log_2(1/\epsilon)$ for the continuous case for an ϵ-accurate optimal solution. This is a substantial improvement compared to the algorithms in Luss and Gupta [13], Yao and Shanthikumar [22], and Zipkin [23]. It should be noted that they require the functions to be strictly convex and continuously differentiable (a condition not required for the procedure in this paper). Helgason et al. [6] describe an $O(n \log n)$ algorithm for the problem, when the objective is separable quadratic, which derives directly the optimal solution (and in strongly polynomial time). Such exact optimal solution rather than an ϵ-accurate optimal solution can be derived in the quadratic case since the optimality conditions are linear and the length of the output is hence polynomial in the input.

Finally, Monteiro and Adler [17], derived an adaptation of linear programming interior point methods restricted to a class of nonlinear convex separable objective functions. The complexity of the algorithm depends on the objective function. It is not polynomial for finding ϵ-accurate optimal solutions. Therefore, that algorithm is not comparable to our algorithm for that class of problems.

1.2 A Complexity Model. There is no complexity model available for the description of general functions. The only attempt to define such a model appears in Nemirovsky and Yudin's book [18]. Problems of the type we are considering pose certain difficulty as far as their complexity is concerned. This is due to the fact that the length of the output—the description of the solution—may be infinite, and the length of the input is not bounded either. This happens when we have nonanalytical functions, or even functions without explicit regular presentations. Such a function could be considered as an infinitely (noncountable) long table that, with our algorithm, never needs even to be looked at more than for a polynomial number of entries, each with polynomially long input and output. Since nonlinear functions cannot be treated with the absolute accuracy of linear functions, the notion of approximate solutions is particularly important. We prefer to approximate the solution vector, since in our approach a specification of the accuracy of the solution vector implies directly the complexity of the algorithm and the arithmetic accuracy with which it is to work. Nemirovsky and Yudin choose to approximate the objective value. If there is certain information about the behavior of the objective at the optimum, that can always be translated to a level of accuracy of the solution vector itself (and vice versa).

For the nonlinear optimization problems that are considered here, this translation process will work as follows. First, derive an upper bound on the absolute value of the variation, d_i of the function f_i over an interval of unit length. This can be done by taking the maximum of the absolute values of the first order difference to the left of the left endpoint and to the right of the right endpoint of the interval bounding the value of x_i in the optimal solution vector. For a given required approximation of the objective function, δ, one determines the accuracy of the solution vector as $\epsilon = \delta / \sum_{i=1}^{n} d_i$. The value of ϵ specified will determine the

maximum distance between the optimal solution and the derived solution components. It will also imply the grid and the arithmetic accuracy with which that approximate solution is to be found.

1.3 OVERVIEW OF ALGORITHM. The algorithm maintains a box that contains an optimal solution to the problem. At each iteration, the size of this box is reduced by a factor of 2^n. This is achieved by reducing each dimension of the box by a factor of 2. A direct implementation will therefore require a total of $\log_2 B$ iterations, where B is the length of the initial box for the integer problem. Certain modifications (to be discussed in Section 4) will reduce this complexity. For the continuous problem solved to ϵ-accuracy, there are $\log_2(B/2\epsilon)$ such iterations.

At a given iteration, the interval for each variable x_i is divided into a grid of $O(n\Delta)$ points, where Δ is the bound on the absolute value of a subdeterminant of the matrix A. The nonlinear function is approximated by a piecewise linear function on that particular grid. Due to the convexity, this piecewise linear approximation is solved as an ordinary linear program (e.g., [2, pp. 482–486]).

We prove a proximity result between the optimal solution to the nonlinear problem (integer or continuous) and the optimal solution derived for the piecewise linear approximation. This proximity is a function of n, Δ, and the size of the grid (i.e., the scaling constant). We choose the grid size such that the optimal solution to the piecewise linear approximation is at most one-fourth the length of the box away from the optimal solution. This allows us to update the box in which the optimal solution is to be found, and reduce its size by a factor of 2^n.

When the grid corresponds to the integer grid, the final iteration for the integer problem consists of solving the integer problem on that particular grid. That problem looks precisely like a linearized version of the original problem except that each variable has $O(n\Delta)$ copies. It is important to notice that the duplication of the variables does not change the complexity of solving the linear problem, except for an adjustment for the number of variables. The size of the largest subdeterminant of the constraint matrix does not change, since we only added copies of the columns of the matrix (that are obviously linearly dependent). Since the number of variables though grows by a factor of $O(n\Delta)$, the requirement that Δ is polynomially bounded is essential for our algorithm.

As for the continuous problem, the iterations continue, until the optimal solution interval is reduced to a size at most 2ϵ. We summarize the properties of the algorithm in the following theorems.

THEOREM 1.1. *Let the complexity of Linear Programming* $Min\{\mathbf{cx} \mid A\mathbf{x} \geq \mathbf{b}, \mathbf{0} \leq \mathbf{x} \leq \mathbf{1}\}$ *be* $T(n, m, \Delta)$, *then the complexity of solving for an* ϵ*-accurate optimal solution to a nonlinear separable and convex (concave) minimization (maximization) problem on* $\{\mathbf{x} \mid A\mathbf{x} \geq \mathbf{b}\}$ *is* $\log_2(B/2\epsilon)T(8n^2\Delta, m, \Delta)$.

In the theorem for the integer problem, the notation A^k is used for a matrix A in which each column appears k times.

THEOREM 1.2. *Let the complexity of solving an Integer Linear Programming problem* $Min\{\mathbf{cx} \mid A\mathbf{x} \geq \mathbf{b}, \mathbf{0} \geq \mathbf{x} \geq \mathbf{1}, \mathbf{x}$ *integer*$\}$ *be* $TI(n, m, A)$, *then the complexity of solving a nonlinear separable convex integer optimization problem on* $\{\mathbf{x} \mid A\mathbf{x} \geq \mathbf{b}\}$ *is*

$$\log_2 \frac{B}{2n\Delta}\, T(8n^2\Delta, m, \Delta) + TI(4n^2\Delta, m, A^{4n\Delta}).$$

Remark 1.3. We can write the complexity of solving the binary linear program as TI(n, m, A), that is, independent of **b** and **c**. The independence of **c** follows from [4]. Since the variables are all 0–1, the right-hand side, **b**, can be replaced by a vector of entries that are functions of A and n only. This is because the left-hand side of all inequalities can add up to integers in the interval $[\sum_{a_{ij}<0} a_{ij}, \sum_{a_{ij}>0} a_{ij}]$ only.

In the following section, we provide a formal problem statement and discuss the method of solution of the piecewise linear optimization problems. Section 3 includes all the proofs of the proximity theorems. Section 4 describes the algorithms used and their complexity in the unit cost model, that is, in terms of the number of arithmetic operations. Finally, Section 5 studies the accuracy of the oracle computing the objective function value in terms of the number of digits required to derive the optimal solutions.

2. *Problem Statement*

Let $f_i: R \to R$, $i = 1, \ldots, n$ be n convex functions and define

$$F(\mathbf{x}) := \sum_{i=1}^{n} f_i(x_i), \qquad \mathbf{x} := (x_1, \ldots, x_n) \in R^n. \tag{2.1}$$

We are interested in the solutions to the nonlinear integer programming problem

(IP) Min $F(\mathbf{x})$
such that $A\mathbf{x} \geq \mathbf{b}$
$\mathbf{x}$ integer

and to its continuous relaxation

(RP) Min $F(\mathbf{x})$
such that $A\mathbf{x} \geq \mathbf{b}$.

Here A is an integral $m \times n$ matrix and **b** is an m-vector. The solutions to (IP) and (RP) will be obtained by solving a sequence of scaled and linearized versions of problems (IP) and (RP). For this, we introduce the following classes of problems. For any scaling constant $s \in R_+$ let the scaled problem (IP $- s$) be defined by

(IP $- s$) Min $F(s\mathbf{y})$
$A\mathbf{y} \geq \mathbf{b}/s$
$\mathbf{y}$ integer.

By setting $\mathbf{x} = s\mathbf{y}$ in (IP $- s$), it can be observed that the only difference between (IP) and (IP $- s$) is that in (IP) we require the solution to be an integer while in (IP $- s$) we require it to be an integer multiple of s. Hence, for $s = 1$, both (IP) and (IP $- s$) are the same problems. The continuous relaxation of (IP $- s$) with $\mathbf{x} := s\mathbf{y}$ is easily seen to be the same as (RP). We do not need the scaled version of (RP). Rather, we use a linearized version of (RP) as defined below. For any $s > 0$ let $f_i^{L:s}: R \to R$ be the linearized version of f_i such that $f_i^{L:s}$ takes the same value as f_i at all integer multiples of s: that is, $f_i^{L:s}(sy_i) = f_i(sy_i)$, for y_i integer and

$$f_i^{L:s}(x_i) = \left(\left\lfloor \frac{x_i}{s} + 1 \right\rfloor - \frac{x_i}{s}\right) f_i\left(s\left\lfloor \frac{x_i}{s} \right\rfloor\right) + \left(\frac{x_i}{s} - \left\lfloor \frac{x_i}{s} \right\rfloor\right) f_i\left(s\left\lfloor \frac{x_i}{s} + 1 \right\rfloor\right), \quad x_i \in R, \tag{2.2}$$

where $\lfloor x_i/s \rfloor$ is the largest integer value smaller than or equal to x_i/s. Clearly $f_i^{L:s}$ is a piecewise linear function which is convex if f_i is convex. Now define

$$\begin{aligned}(\text{LP} - s)\quad &\text{Min } F^{L:s}(\mathbf{x})\\ &A\mathbf{x} \geq \mathbf{b}\end{aligned}$$

where

$$F^{L:s}(\mathbf{x}) := \sum_{i=1}^{n} f_i^{L:s}(x_i), \qquad \mathbf{x} \in R^n. \tag{2.3}$$

Note that the optimal solution to the integer program

$$\begin{aligned}(\text{IP}' - s)\quad &\text{Min } F^{L:s}(s\mathbf{y})\\ &A\mathbf{y} \geq \mathbf{b}/s\\ &\mathbf{y} \text{ integer}\end{aligned}$$

is also an optimal solution to (IP $- s$) because $F^{L:s}$ and F take the same value at integer multiples of s, that is, $F^{L:s}(s\mathbf{y}) = F(s\mathbf{y})$ for all integer vectors $\mathbf{y}$. Hence, to solve (IP $- s$) we may solve (IP$' - s$) and use its optimal solution.

(LP $- s$) or (IP $- s$) is solved by using a linear programming formulation. Since the optimal solution is enclosed in a bounded box, we incorporate those bounds into the feasible solution set. That is, we add the constraints to the (RP) or (IP) problems:

$$L_i \leq x_i \leq U_i, \qquad i = 1, \ldots, n.$$

These constraints will be scaled as well. In our procedure, at each iteration we shall work with the upper and lower bounds, U_i and L_i, and a scaling constant s, such that the length $(U_i - L_i)/s$, is an integer constant independent of i. We denote $N = (U_i - L_i)/s$. Each variable y_i, for the integer case, is then substituted by a sum of N 0–1 variables:

$$y_i = \left\lfloor \frac{L_i}{s} \right\rfloor + \sum_{j=1}^{N} z_{ij}$$

$$\text{for} \quad i = 1, \ldots, n \qquad z_{ij} \text{ is 0 or 1} \qquad \text{for} \quad \text{all } i \text{ and } j. \tag{2.4}$$

For the continuous case the substitution for x_i is:

$$x_i = s\left\{\left\lfloor \frac{L_i}{s} \right\rfloor + \sum_{j=1}^{N} z_{ij}\right\}$$

$$\text{for } i = 1, \ldots, n \qquad 0 \leq z_{ij} \leq 1 \qquad \text{for} \quad \text{all } i \text{ and } j. \tag{2.5}$$

So now (LP $- s$) and (IP$' - s$) are piecewise linear convex (concave) minimization (maximization) problems on the variables z_{ij}.

The modified objective function for the linear programming formulation (e.g., [2, pp. 482–486]) of both (LP $- s$) and (IP$' - s$) is

$$\text{Min} \sum_{i=1}^{n} f_i^{L:s}\left(s\left\lfloor \frac{L_i}{s} \right\rfloor\right) + \sum_{i=1}^{n}\sum_{j=1}^{N}\left[f_i^{L:s}\left(s\left(\left\lfloor \frac{L_i}{s} \right\rfloor + j\right)\right) - f_i^{L:s}\left(s\left(\left\lfloor \frac{L_i}{s} \right\rfloor + j - 1\right)\right)\right] z_{ij}.$$

We denote the columns of A by $a_1, \ldots, a_n$. Then the constraint set

$$\sum_{i=1}^{n} a_i x_i \geq b \qquad \left[\sum_{i=1}^{n} a_i y_i \geq \frac{b}{s}\right]$$

of (LP $-$ s) [respectively, (IP$'$ $-$ s)] is converted using the substitution (2.5) [respectively, (2.4)] into the constraint set

$$\sum_{i=1}^{n}\sum_{j=1}^{N} a_i z_{ij} \geq b',$$

for both (LP $-$ s) and (IP$'$ $-$ s), where $b' = b/s - \sum_{i=1}^{n} a_i \lfloor L_i/s \rfloor$. So the linear programming version of the (LP $-$ s) problem (omitting the constant from the objective function) is

$$\text{(LP}' - s\text{)} \quad \text{Min} \sum_{i=1}^{n}\sum_{j=1}^{N}\left[f_i^{L:s}\left(s\left(\left\lfloor\frac{L_i}{s}\right\rfloor + j\right)\right) - f_i^{L:s}\left(s\left(\left\lfloor\frac{L_i}{s}\right\rfloor + j - 1\right)\right)\right] z_{ij}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{N} a_i z_{ij} \geq b', \qquad 0 \leq z_{ij} \leq 1, \quad j = 1, \ldots, N, \quad i = 1, 2, \ldots, n.$$

One then has from well-known results (e.g., [2]):

LEMMA 2.1. *Let $\hat{\mathbf{z}}$ be an optimal solution to (LP$'$ $-$ s). If f_i is convex for each $i = 1, \ldots, n$, then $\hat{\mathbf{x}}$ defined by $\hat{x}_i = s(\lfloor L_i/s \rfloor + \sum_{j=1}^{N} \hat{z}_{ij})$, $i = 1, \ldots, n$, is an optimal solution to (LP $-$ s).*

The treatment of the integer problem (IP$'$ $-$ s) is similar. For this we have

LEMMA 2.2. *Let $\hat{\mathbf{z}}$ be an optimal solution to (LP$'$ $-$ s) with the added constraint that $\mathbf{z}$ is a 0–1 vector. If f_i is convex for each $i = 1, \ldots, n$, then $\hat{\mathbf{y}}$ defined by*

$$\hat{y}_i = \left\lfloor\frac{L_i}{s}\right\rfloor + \sum_{j=1}^{N} \hat{z}_{ij}, \qquad i = 1, \ldots, n$$

is an optimal solution to (IP$'$ $-$ s).

Note that in both (LP$'$ $-$ s) and (IP$'$ $-$ s), the constraint matrix is A^N with each column of A appearing precisely N times, $A^N = [a_1, \ldots, a_1; a_2, \ldots, a_2; \cdots; a_n, \ldots, a_n]$.

Polynomial linear programming algorithms such as the ellipsoid method or Karmarkar's method, can be adapted (see [21]) to run in time that is polynomial in the number of variables, n, the number of constraints, m, and the length of the largest subdeterminant of the constraint matrix, A. Denote the running time of a selected linear programming algorithm, solving $\min\{\mathbf{cx} \mid A\mathbf{x} \geq \mathbf{b}, \mathbf{0} \leq \mathbf{x} \leq \mathbf{1}\}$, by $T(n, m, \Delta)$. It is important to notice that the size of the biggest subdeterminant of A^N is exactly the same as that of A's, Δ. Consequently, the complexity of solving (LP$'$ $-$ s) is $T(Nn, m, \Delta)$. For (IP$'$ $-$ s), the complexity is $\text{TI}(Nn, m, A^N)$, where $\text{TI}(n, m, A)$ is the running time required to solve $\min\{\mathbf{cx} \mid A\mathbf{x} \geq \mathbf{b}, \mathbf{0} \leq \mathbf{x} \leq \mathbf{1}$, and $\mathbf{x}$ integer$\}$. Recall that $\text{TI}(n, m, A)$ is independent of $\mathbf{b}$ and $\mathbf{c}$, see Remark 1.3.

3. *Proximity Results*

In this section, proximity results for the optimal solutions of (IP), (RP), (IP $-$ s) and (LP $-$ s) will be derived. These results will be used to develop the algorithms to solve (IP) and (RP). The following preliminaries are needed.

3.1 PRELIMINARIES. Let $g_i: R \to R$, $i = 1, \ldots, n$ and

$$G(\mathbf{x}) := \sum_{i=1}^{n} g_i(x_i), \qquad \mathbf{x} \in R^n. \tag{3.1}$$

We shall use the notation $\mathbf{y} \wedge \mathbf{z}$ and $\mathbf{y} \vee \mathbf{z}$ to denote

$$\mathbf{y} \wedge \mathbf{z} := (\min\{y_1, z_1\}, \ldots, \min\{y_n, z_n\})$$

and

$$\mathbf{y} \vee \mathbf{z} := (\max\{y_1, z_1\}, \ldots, \max\{y_n, z_n\}).$$

In order to establish the proximity result, we shall need a property stronger than convexity for the objective function. This property, called *directional convexity*, is shown for separable convex functions in the following lemma:

LEMMA 3.1. *Suppose G is convex (i.e., g_i is convex for each i, $i = 1, \ldots, n$). Then for any quadruple of vectors $\mathbf{x}^{(j)} \in R^n$, $j = 1, \ldots, 4$ that satisfies*

(i) $\mathbf{x}^{(1)} \wedge \mathbf{x}^{(4)} \leq \mathbf{x}^{(2)} \leq \mathbf{x}^{(1)} \vee \mathbf{x}^{(4)}$
(ii) $\mathbf{x}^{(1)} \wedge \mathbf{x}^{(4)} \leq \mathbf{x}^{(3)} \leq \mathbf{x}^{(1)} \vee \mathbf{x}^{(4)}$, *and*
(iii) $\mathbf{x}^{(1)} + \mathbf{x}^{(4)} = \mathbf{x}^{(2)} + \mathbf{x}^{(3)}$,

we have

$$G(\mathbf{x}^{(1)}) + G(\mathbf{x}^{(4)}) \geq G(\mathbf{x}^{(2)}) + G(\mathbf{x}^{(3)}).$$

PROOF. Let $\mathbf{y}^{(1)} = \mathbf{x}^{(1)} \wedge \mathbf{x}^{(4)}$ and $\mathbf{y}^{(4)} = \mathbf{x}^{(1)} \vee \mathbf{x}^{(4)}$.

Then, from (i), (ii), and (iii), one has for any i, $i = 1, \ldots, n$,

$$y_i^{(1)} \leq x_i^{(2)} \leq y_i^{(4)},$$
$$y_i^{(1)} \leq x_i^{(3)} \leq y_i^{(4)},$$

and

$$y_i^{(1)} + y_i^{(4)} = x_i^{(2)} + x_i^{(3)}.$$

Then, from the convexity of g_i, it is easily seen that

$$g_i(y_i^{(1)}) + g_i(y_i^{(4)}) \geq g_i(x_i^{(2)}) + g_i(x_i^{(3)}). \tag{3.2}$$

Therefore, summing (3.2) over all i, $i = 1, \ldots, n$, one gets

$$G(\mathbf{y}^{(1)}) + G(\mathbf{y}^{(4)}) \geq G(\mathbf{x}^{(2)}) + G(\mathbf{x}^{(3)}). \tag{3.3}$$

Due to the separability of G, $G(\mathbf{y}^{(1)}) + G(\mathbf{y}^{(4)}) = G(\mathbf{x}^{(1)}) + G(\mathbf{x}^{(4)})$. The desired result then follows from (3.3). □

Remark 3.2. Consider a fixed partition $\langle S_1, S_2 \rangle$ of $\{1, \ldots, n\}$. Consider $\mathbf{x}^{(j)} \in R^n$, $j = 1, \ldots, 4$ that satisfy conditions (i), (ii), and (iii) of Lemma 3.2 *and* $x_i^{(1)} \leq x_i^{(4)}$, $i \in S_1$, and $x_i^{(1)} \geq x_i^{(4)}$, $i \in S_2$. Let $G: R^n \to R$ be a function (not necessarily separable), that satisfies for such $\mathbf{x}^{(j)}$,

$$G(\mathbf{x}^{(1)}) + G(\mathbf{x}^{(4)}) \geq G(\mathbf{x}^{(2)}) + G(\mathbf{x}^{(3)}).$$

Then, Shaked and Shanthikumar [20] call G directionally convex (in the direction specified by $\langle S_1, S_2 \rangle$). A necessary and sufficient condition for this directional convexity (when these derivatives exist) is:

$$\frac{\partial^2}{\partial x_i^2} G(\mathbf{x}) \geq 0, \quad i = 1, \ldots, n$$

$$\frac{\partial^2}{\partial x_i \partial x_j} G(\mathbf{x}) \geq 0, \quad i, j \in S_1 \quad \text{or} \quad i, j \in S_2$$

and

$$\frac{\partial^2}{\partial x_i \partial x_j} G(\mathbf{x}) \leq 0, \qquad i \in S_1, \quad j \in S_2.$$

(See Proposition 2.3 and remarks in [20].) If G is twice differentiable, then Lemma 3.1 follows from Shaken and Shanthikumar [20]. Notice that a separable convex function is directionally convex in all directions. If G is directionally convex in all directions, first choosing the direction $\langle S_1, S_2 \rangle$ such that $i, j \in S_1$, and then choosing $\langle S_1', S_2' \rangle$ such that $i \in S_1', j \in S_2'$, it is seen that all cross derivatives $(\partial^2/\partial x_i \partial x_j)G(\mathbf{x})$, $i \neq j$ vanish. Hence, G should be separable. Indeed, even if the function is not differentiable, it can be verified that a convex function is directionally convex in all directions iff it is a separable convex function. Therefore, since we use the directional convexity in developing the proximity results and the algorithms, it appears that our approach may not easily extend to nonseparable objective functions.

Let Δ be the maximum of the absolute values of the determinants of the square submatrices of A and $\| \mathbf{x} \|_\infty = \max\{| x_i |, i = 1, \ldots, n\}$ be the l_∞-norm of $\mathbf{x} \in R^n$. We next give a proximity result between the optimal solution of (IP) and (RP).

THEOREM 3.3

(*i*) *For each optimal solution* $\hat{\mathbf{x}}$ *for* (*RP*), *there exists an optimal solution* $\mathbf{z}^*$ *for* (*IP*) *such that* $\| \hat{\mathbf{x}} - \mathbf{z}^* \|_\infty \leq n\Delta$.

(*ii*) *For each optimal solution* $\hat{\mathbf{z}}$ *for* (*IP*), *there exists an optimal solution* $\mathbf{x}^*$ *for* (*RP*) *such that* $\| \mathbf{x}^* - \hat{\mathbf{z}} \|_\infty \leq n\Delta$.

PROOF. Let $\hat{\mathbf{z}}$ and $\hat{\mathbf{x}}$ be two optimal solutions for (IP) and (RP), respectively. We first construct a cone with respect to $\hat{\mathbf{z}}$ and $\hat{\mathbf{x}}$ along the same line as in the proof of Theorem 1 of Granot and Skorin-Kapov [5]. Let $\{S_1, S_2\}$ be a partition of $\{1, \ldots, n\}$ such that for any $i \in S_1$, $\hat{z}_i \geq \hat{x}_i$ and for any $i \in S_2$, $\hat{z}_i < \hat{x}_i$. Also partition the matrix A into submatrices A_1, and A_2 such that $A_1\hat{\mathbf{z}} < A_1\hat{\mathbf{x}}$ and $A_2\hat{\mathbf{z}} \geq A_2\hat{\mathbf{x}}$. Define the cone $C = \{\mathbf{y}: A_1\mathbf{y} \leq \mathbf{0}, A_2\mathbf{y} \geq \mathbf{0}, y_i \geq 0, i \in S_1, y_i \leq 0, i \in S_2, \mathbf{y} \in R^n\}$. Note that $\hat{\mathbf{z}} - \hat{\mathbf{x}} \in C$. Let $U \subset C$ be a finite set of integral vectors that generates C. Then, because of the integrality of the elements of A, for each $\mathbf{u} \in U$, $\| \mathbf{u} \|_\infty \leq \Delta$, and since $\hat{\mathbf{z}} - \hat{\mathbf{x}} \in C$, there exist t ($t \leq n$) vectors $\mathbf{u}^{(j)} \in U$, $j = 1, \ldots, t$ and $\alpha_j > 0$, $j = 1, \ldots, t$ such that $\hat{\mathbf{z}} - \hat{\mathbf{x}} = \sum_{j=1}^{t} \alpha_j \mathbf{u}^{(j)}$ (e.g., [1]). That is,

$$\hat{\mathbf{z}} = \hat{\mathbf{x}} + \sum_{j=1}^{t} \alpha_j \mathbf{u}^{(j)}. \tag{3.4}$$

Let

$$\beta_j = \alpha_j - \lfloor \alpha_j \rfloor, \qquad j = 1, \ldots, t$$

and define

$$\mathbf{z}^* = \hat{\mathbf{x}} + \sum_{j=1}^{t} \beta_j \mathbf{u}^{(j)}, \tag{3.5}$$

and

$$\mathbf{x}^* = \hat{\mathbf{z}} - \sum_{j=1}^{t} \beta_j \mathbf{u}^{(j)}. \tag{3.6}$$

From (3.4) and (3.5), one also has

$$\mathbf{z}^* = \hat{\mathbf{z}} - \sum_{j=1}^{t} (\alpha_j - \beta_j)\mathbf{u}^{(j)}. \tag{3.7}$$

Similarly, from (3.4) and (3.6), one has

$$\mathbf{x}^* = \hat{\mathbf{x}} + \sum_{j=1}^{t} (\alpha_j - \beta_j)\mathbf{u}^{(j)}. \tag{3.8}$$

Next, we show that $\mathbf{z}^*$ and $\mathbf{x}^*$ are feasible for (IP) and (RP), respectively. Observe that

$$A_1\mathbf{z}^* \underset{(3.7)}{=} A_1\hat{\mathbf{z}} - \sum_{j=1}^{t} (\alpha_j - \beta_j)A_1\mathbf{u}^{(j)} \geq A_1\hat{\mathbf{z}},$$

since $\alpha_j - \beta_j \geq 0$ and $\mathbf{u}^{(j)} \in C$ (i.e., $A_1\mathbf{u}^{(j)} \leq \mathbf{0}$, $j = 1, \ldots, t$). Similarly

$$A_2\mathbf{z}^* \underset{(3.5)}{=} A_2\hat{\mathbf{x}} + \sum_{j=1}^{t} \beta_j A_2\mathbf{u}^{(j)} \geq A_2\hat{\mathbf{x}}.$$

Hence, $A\mathbf{z}^* \geq \mathbf{b}$. Since $\alpha_j - \beta_j = \lfloor \alpha_j \rfloor$ and $\mathbf{u}^{(j)}$ are integers, the integrality of $\mathbf{z}^*$ is immediate from (3.7). Likewise,

$$A_1\mathbf{x}^* \underset{(3.6)}{=} A_1\hat{\mathbf{z}} - \sum_{j=1}^{t} \beta_j A_1\mathbf{u}^{(j)} \geq A_1\hat{\mathbf{z}}$$

and

$$A_2\mathbf{x}^* \underset{(3.8)}{=} A_2\hat{\mathbf{x}} + \sum_{j=1}^{t} (\alpha_j - \beta_j)A_2\mathbf{u}^{(j)} \geq A_2\hat{\mathbf{x}}.$$

Hence, $A\mathbf{x}^* \geq \mathbf{b}$. Since $\mathbf{u}^{(j)} \in C$ one has $u_i^{(j)} \geq 0$, $i \in S_1$, and $u_i^{(j)} \leq 0$, $i \in S_2$. Combining this with $\hat{z}_i \geq \hat{x}_i$, $i \in S_1$ and $\hat{z}_i < \hat{x}_i$, $i \in S_2$ one has from (3.5) and (3.7)

$$\begin{aligned} \hat{z}_i \geq z_i^* \geq \hat{x}_i, \quad & i \in S_1 \\ \hat{z}_i \leq z_i^* \leq \hat{x}_i, \quad & i \in S_2. \end{aligned} \tag{3.9}$$

Similarly, from (3.6) and (3.8), one has

$$\begin{aligned} \hat{z}_i \geq x_i^* \geq \hat{x}_i, \quad & i \in S_1 \\ \hat{z}_i \leq x_i^* \leq \hat{x}_i, \quad & i \in S_2. \end{aligned} \tag{3.10}$$

From (3.9) and (3.10), one sees that

$$\hat{\mathbf{z}} \wedge \hat{\mathbf{x}} \leq \mathbf{z}^* \leq \hat{\mathbf{z}} \vee \hat{\mathbf{x}} \tag{3.11}$$

and

$$\hat{\mathbf{z}} \wedge \hat{\mathbf{x}} \leq \mathbf{x}^* \leq \hat{\mathbf{z}} \vee \hat{\mathbf{x}}. \tag{3.12}$$

Furthermore, from (3.5) and (3.6), one has

$$\hat{\mathbf{z}} + \hat{\mathbf{x}} = \mathbf{z}^* + \mathbf{x}^*. \tag{3.13}$$

Since F is separable, from (3.11), (3.12), and (3.13) and from Lemma 3.1, one gets

$$F(\hat{\mathbf{z}}) + F(\hat{\mathbf{x}}) \geq F(\mathbf{z}^*) + F(\mathbf{x}^*). \tag{3.14}$$

Therefore

$$F(\mathbf{z}^*) \leq F(\hat{\mathbf{z}}) + F(\hat{\mathbf{x}}) - F(\mathbf{x}^*) \leq F(\hat{\mathbf{z}}),$$

since $\hat{\mathbf{x}}$ is an optimal solution to (RP) and $\mathbf{x}^*$ is a feasible solution to (RP) (i.e., $A\mathbf{x}^* \geq \mathbf{b}$). Therefore, $\mathbf{z}^*$ is an alternate optimal solution to (IP) (feasibility is already established) and $F(\mathbf{z}^*) = F(\hat{\mathbf{z}})$. Consequently, $F(\hat{\mathbf{x}}) = F(\mathbf{x}^*)$ and $\mathbf{x}^*$ is an optimal solution to (RP). From (3.5), one has

$$\| \hat{\mathbf{x}} - \mathbf{z}^* \|_\infty = \left\| \sum_{j=1}^{t} \beta_j \mathbf{u}^{(j)} \right\|_\infty \leq n\Delta.$$

Similarly from (3.6)

$$\| \mathbf{x}^* - \hat{\mathbf{z}} \|_\infty = \left\| \sum_{j=1}^{t} \beta_j \mathbf{u}^{(j)} \right\|_\infty \leq n\Delta. \qquad \square$$

Along the same line suppose $\mathbf{z}$ is an integral solution of $A\mathbf{x} \geq \mathbf{b}$ that is not an optimal solution to (IP). Let $\mathbf{z}^*$ be an optimal solution to (IP). Then, as in the proof of Theorem 3.3, it can be shown that there exist two integral solutions $\hat{\mathbf{z}}$ and $\mathbf{z}'$ for $A\mathbf{x} \geq \mathbf{b}$ such that $\| \mathbf{z} - \hat{\mathbf{z}} \|_\infty = \| \mathbf{z}' - \mathbf{z}^* \|_\infty \leq n\Delta$ and $F(\mathbf{z}) + F(\mathbf{z}^*) \geq F(\mathbf{z}') + F(\hat{\mathbf{z}})$. Since, $F(\mathbf{z}^*) < F(\mathbf{z})$, it follows that $F(\hat{\mathbf{z}}) < F(\mathbf{z})$. Therefore, one has for (IP) that has an optimal solution:

THEOREM 3.4. *For each integral solution* $\mathbf{z}$ *of* $A\mathbf{x} \geq \mathbf{b}$ *either* $\mathbf{z}$ *is an optimal solution to (IP) or there exists an integral solution* $\hat{\mathbf{z}}$ *of* $A\mathbf{x} \geq \mathbf{b}$ *with* $\| \mathbf{z} - \hat{\mathbf{z}} \|_\infty \leq n\Delta$ *and* $F(\hat{\mathbf{z}}) < F(\mathbf{z})$.

Using a proof very similar to that of Theorem 3.3, we can obtain the following proximity between the optimal solutions to (IP) and (IP $- s$).

THEOREM 3.5. *Let* s *be a positive integer.*

(*i*) *For each optimal solution* $\hat{\mathbf{y}}$ *for (IP* $- s$*), there exists an optimal solution* $\mathbf{z}^*$ *for (IP) such that*

$$\| s\hat{\mathbf{y}} - \mathbf{z}^* \|_\infty \leq ns\Delta.$$

(*ii*) *For each optimal solution* $\hat{\mathbf{z}}$ *for (IP), there exists an optimal solution* $\mathbf{y}^*$ *for (IP* $- s$*) such that*

$$\| s\mathbf{y}^* - \hat{\mathbf{z}} \|_\infty \leq ns\Delta.$$

Remark 3.6. Since (RP) is a continuous relaxation of (IP $- s$) with $\mathbf{x} := s\mathbf{y}$, from Theorem 3.3, one sees that for any optimal solution $\hat{\mathbf{y}}$ of (IP $- s$), there exists an optimal solution $\mathbf{x}^*$ to (RP) such that $\| \mathbf{x}^*/s - \hat{\mathbf{y}} \|_\infty \leq n\Delta$. Also from Theorem 3.3, we know that there exists an optimal solution $\mathbf{z}^*$ to (IP) such that $\| \mathbf{x}^* - \mathbf{z}^* \|_\infty \leq n\Delta$. Combining these two proximity results using the triangular inequality of the l_∞ norm, one has $\| s\hat{\mathbf{y}} - \mathbf{z}^* \|_\infty \leq n(1 + s)\Delta$. Similarly, it can be shown that for any optimal solution $\hat{\mathbf{y}}$ of (IP $- s$), there exists an optimal solution $\mathbf{y}^*$ for (IP $- s$) such that $\| s\mathbf{y}^* - \hat{\mathbf{z}} \|_\infty \leq n(1 + s)\Delta$. These bounds, however, are slightly improved in Theorem 3.5.

The following proximity result between the optimal solutions of (IP) and (LP $- s$) will be used to develop an algorithm for (IP).

THEOREM 3.7. *Let s be a fixed positive integer. For every optimal solution $\hat{\mathbf{x}}$ for (LP − s), there exists an optimal solution $\mathbf{z}^*$ for (IP) such that*

$$\| \hat{\mathbf{x}} - \mathbf{z}^* \|_\infty \leq 2ns\Delta.$$

PROOF. Let $\hat{\mathbf{x}}$ be a given optimal solution to (LP − s). Since (LP − s) is a continuous relaxation of (IP′ − s) with $\mathbf{x} := s\mathbf{y}$, from Theorem 3.3, one knows that there exists an optimal solution $\mathbf{y}^*$ to (IP′ − s) (and hence (IP − s)) such that $\| \hat{\mathbf{x}}/s - \mathbf{y}^* \|_\infty \leq n\Delta$. From Theorem 3.5, it is clear that there exists an optimal solution $\mathbf{z}^*$ to (IP) such that $\| s\mathbf{y}^* - \mathbf{z}^* \|_\infty \leq ns\Delta$. Combining the above two proximity results, one gets the desired conclusion. □

To develop an algorithm for (RP), we use the following proximity result between the optimal solutions to (RP) and (LP − s).

THEOREM 3.8. *Let $s > 0$ be fixed. For every optimal solution $\hat{\mathbf{x}}$ for (LP − s), there exists an optimal solution $\mathbf{x}^*$ for (RP) such that*

$$\| \hat{\mathbf{x}} - \mathbf{x}^* \|_\infty \leq 2ns\Delta.$$

PROOF. Let $\hat{\mathbf{x}}$ be a given optimal solution to (LP − s). Then, as in the proof of Theorem 3.3, there exists an optimal solution $\mathbf{y}^*$ for (IP − s) such that $\| \hat{\mathbf{x}} - s\mathbf{y}^* \|_\infty \leq ns\Delta$. Observe that the continuous relaxation of (IP − s) with $\mathbf{x} := s\mathbf{y}$ is the same problem as (RP). Hence, from Theorem 3.3, it follows that there exists an optimal solution $\mathbf{x}^*$ for (RP) such that $\| s\mathbf{y}^* - \mathbf{x}^* \|_\infty \leq ns\Delta$. Combining the above two proximity results, one gets the desired conclusion. □

4. *Algorithms and Complexity*

In this section, we present algorithms for (IP) and (RP) and discuss their complexity. First consider

$$\begin{aligned} \text{(IP)}\quad & \text{Min } F(\mathbf{x}) \\ & A\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \text{ integer} \end{aligned}$$

As pointed out earlier, we can assume that the polyhedron $\{\mathbf{x}: A\mathbf{x} \geq \mathbf{b}, \mathbf{x} \in R^n\}$ is bounded (since if it is not, there is a known bound on an optimal solution so that when it is incorporated into the constraint set it will form a polytope). There exists an integer $\gamma \geq 1$, such that the optimal solution to,

$$\begin{aligned} \text{(IP}^{(1)}\text{)}\quad & \text{Min } F(\mathbf{x}) \\ & A\mathbf{x} \geq \mathbf{b} \\ & -2^{\gamma+1}n\Delta\mathbf{e} \leq \mathbf{x} \leq 2^{\gamma+1}n\Delta\mathbf{e} \\ & \mathbf{x} \text{ integer} \end{aligned}$$

where $\mathbf{e} := (1, \ldots, 1)^T$, is also an optimal solution to (IP). The existence of such γ follows from the fact (e.g., [19, p. 30]) that the box $\{\mathbf{x}: \| \mathbf{x} \|_\infty \leq \| \mathbf{b} \|_\infty m \cdot \Delta, \mathbf{x} \in R^n\}$ of length, $B = 2\| \mathbf{b} \|_\infty \cdot m \cdot \Delta$, contains $\{\mathbf{x}: A\mathbf{x} \geq \mathbf{b}, \mathbf{x} \in R^n\}$. Hence, it is sufficient to choose γ such that $2^{\gamma+1}n\Delta \geq \| \mathbf{b} \|_\infty \cdot m \cdot \Delta$. In particular, $\gamma = \lceil \log_2 ((m/n)\| \mathbf{b} \|_\infty) \rceil - 1$ satisfies this inequality. We remark here that adding these upper and lower bounds will not change Δ and hence the proximity results given in Section 3.

Next we present an algorithm for (IP).

Algorithm 4.1

1. Set $s_k = 2^{\gamma-k}$, $k = 0, \ldots, \gamma$; $\hat{\mathbf{x}}^{(0)} = \mathbf{0}$.
2. For $k = 1, \ldots, \gamma$ using the substitution (2.5) and solving the resulting linear program, obtain an optimal solution $\hat{\mathbf{x}}^{(k)}$ to

$$(\text{LP}^* - s_k) \quad \text{Min } F^{L:s_k}(\mathbf{x})$$
$$A\mathbf{x} \geq \mathbf{b}$$
$$\hat{\mathbf{x}}^{(k-1)} - 2ns_{k-1}\Delta\mathbf{e} \leq \mathbf{x} \leq \hat{\mathbf{x}}^{(k-1)} + 2ns_{k-1}\Delta\mathbf{e}$$

3. Obtain an optimal solution $\hat{\mathbf{z}}$ using the substitution (2.4) with $L_i = \mathbf{x}_i^{(\gamma)} - 2n\Delta$ and $s = 1$, and solving the resulting linear integer program,

$$(\text{IP}^*) \quad \text{Min } F(\mathbf{x})$$
$$A\mathbf{x} \geq \mathbf{b}$$
$$\hat{\mathbf{x}}^{(\gamma)} - 2n\Delta\mathbf{e} \leq \mathbf{x} \leq \hat{\mathbf{x}}^{(\gamma)} + 2n\Delta\mathbf{e}$$
$$\mathbf{x} \text{ integer.}$$

Let $T(n, m, \Delta)$ denote the running time of a linear programming algorithm solving $\text{Min}\{\mathbf{cx} \mid A\mathbf{x} \geq \mathbf{b}, \mathbf{0} \leq \mathbf{x} \leq \mathbf{1}\}$. Let $TI(n, m, A)$, denote the running time of an algorithm for solving the integer problem $\text{Min}\{\mathbf{cx} \mid A\mathbf{x} \geq \mathbf{b}, \mathbf{0} \leq \mathbf{x} \leq \mathbf{1}, \mathbf{x} \text{ integer}\}$. A^k denotes the matrix A in which each column is duplicated k times.

THEOREM 4.1. *$\hat{\mathbf{z}}$ obtained by Algorithm* 4.1 *is an optimal solution to (IP). The complexity of Algorithm* 4.1 *is* $\log_2((m/n)\|\mathbf{b}\|_\infty) \cdot T(8n^2\Delta, m, \Delta) + TI(4n^2\Delta, m, A^{4n\Delta})$.

PROOF. Observe that for $k = 1$, $s_1 = 2^{\gamma-1}$ and

$$(\text{LP}^* - s_1) \quad \text{Min } F^{L:s_1}(\mathbf{x})$$
$$A\mathbf{x} \geq \mathbf{b}$$
$$-2^{\gamma+1}n\Delta\mathbf{e} \leq \mathbf{x} \leq 2^{\gamma+1}n\Delta\mathbf{e}.$$

Then $(\text{LP}^* - s_1)$ is a linearized relaxation of $(\text{IP}^{(1)})$. From Theorem 3.7, one sees that there exists an optimal solution $\mathbf{z}^{(1)}$ for $(\text{IP}^{(1)})$ such that $\|\hat{\mathbf{x}}^{(1)} - \mathbf{z}^{(1)}\|_\infty \leq 2ns_1\Delta = 2^\gamma \cdot n \cdot \Delta$. Therefore, the optimal solution to

$$(\text{IP}^{(2)}) \quad \text{Min } F(\mathbf{x})$$
$$A\mathbf{x} \geq \mathbf{b}$$
$$\hat{\mathbf{x}}^{(1)} - 2^\gamma \cdot n\Delta\mathbf{e} \leq \mathbf{x} \leq \hat{\mathbf{x}}^{(1)} + 2^\gamma \cdot n \cdot \Delta\mathbf{e}$$
$$\mathbf{x} \text{ integer}$$

is also an optimal solution to $(\text{IP}^{(1)})$ (and hence to (IP)). $(\text{LP}^* - s_2)$ with $s_2 = 2^{\gamma-2}$ is indeed the linearized relaxation of $(\text{IP}^{(2)})$. Hence, as before from Theorem 3.7, one sees that there exists an optimal solution $\mathbf{z}^{(2)}$ for $(\text{IP}^{(2)})$ and (hence to (IP)) such that $\|\hat{\mathbf{x}}^{(2)} - \mathbf{z}^{(2)}\|_\infty \leq 2ns_2\Delta = 2^{\gamma-1} \cdot n \cdot \Delta$. Continuing this way one finds that there exists an optimal solution $\mathbf{z}^{(k)}$ for (IP) such that $\|\hat{\mathbf{x}}^{(k)} - \mathbf{z}^{(k)}\| \leq 2^{\gamma+1-k} \cdot n \cdot \Delta$, $k = 1, \ldots, \gamma$. Therefore, for any $k = 1, \ldots, \gamma$, an optimal solution for

$$(\text{IP}^{(k)}) \quad \text{Min } F(\mathbf{x})$$
$$A\mathbf{x} \geq \mathbf{b}$$
$$\hat{\mathbf{x}}^{(k-1)} - 2^{\gamma+2-k} \cdot n \cdot \Delta\mathbf{e} \leq \mathbf{x} \leq \hat{\mathbf{x}}^{(k-1)} + 2^{\gamma+2-k} \cdot n \cdot \Delta\mathbf{e}$$
$$\mathbf{x} \text{ integer}$$

is also an optimal solution to (IP). The desired conclusion is reached by observing that there exists an optimal solution $\hat{\mathbf{z}}$ to (IP) such that, $\|\mathbf{x}^{(\gamma)} - \hat{\mathbf{z}}\|_\infty \leq 2n\Delta$ (see Theorem 3.7).

As for the complexity, Step 2 is repeated γ times with $\gamma = \lceil \log_2((m/n)\|\mathbf{b}\|_\infty) \rceil - 1 < \log_2((m/n)\|\mathbf{b}\|_\infty)$. At every iteration, each variable x_i is replaced by $8n\Delta$ z_{ij}

variables (see substitution (2.5)). Thus, the complexity of solving the corresponding linear program is $T(8n^2\Delta, m, \Delta)$. Step 3 in which a linear integer program is solved is executed exactly once. Since x_i for all $i = 1, \ldots, n$ is an interval of length $4n\Delta$, in substitution (2.4), we introduce only $4n\Delta z_{ij}$ variables for each y_i. So the total number of variables in the linearized version is $4n^2\Delta$ and the matrix A has each column duplicated $4n\Delta$ times. Hence, $TI(4n^2\Delta, m, A^{4n\Delta})$ is the complexity of solving the integer problem in Step 3. □

Remark 4.2. When the matrix A is totally unimodular, the optimal solution to (IP $-$ s) can be obtained using Linear Programming. In such a case, a more efficient algorithm can be used. The efficiency in this algorithm derives from the above fact and also from the tighter proximity between (IP) and (IP $-$ s) as proved in Theorem 3.5 (compared to (IP) and (LP $-$ s) as proved in Theorem 3.7). Note that Δ is equal to 1 for totally unimodular matrices.

In the following algorithm, which deals with the case $\Delta = 1$, we need to solve a sequence of integer programs:

$$
\begin{aligned}
(\mathrm{IP}'' - s_k)\ \ \mathrm{Min}\ & F(s_k \mathbf{y}) \\
& A\mathbf{y} \geq \left\lceil \frac{\mathbf{b}}{s_k} \right\rceil \\
& 2\hat{\mathbf{y}}^{(k-1)} - 2n\mathbf{e} \leq \mathbf{y} \leq 2\hat{\mathbf{y}}^{(k-1)} + 2n\mathbf{e} \\
& \mathbf{y} \text{ integer}
\end{aligned}
$$

for $k = 1, \ldots, \gamma$ where $s_k = 2^{\gamma-k}$, $k = 0, \ldots, \gamma$. In order to guarantee that each one of these integer programs has a feasible solution, it is sufficient to have $\hat{\mathbf{y}} = \mathbf{0}$ be feasible for $A\mathbf{x} \geq \mathbf{b}$. For this observe that $\mathbf{0}$ is also a feasible solution to (IP$''$ $-$ s_1) and that $2\hat{\mathbf{y}}^{(k-1)}$ is a feasible solution to (IP$''$ $-$ s_k), $k = 2, \ldots, \gamma$. In case $\mathbf{0}$ is not a feasible solution to $A\mathbf{x} \geq \mathbf{b}$, we may simply obtain a feasible (integer) solution $\mathbf{x}_0$ to $A\mathbf{x} \geq \mathbf{b}$, and replace the constraint set by $A(\mathbf{x} - \mathbf{x}_0) \geq \mathbf{b} - A\mathbf{x}_0$ and substitute $\mathbf{x} - \mathbf{x}_0$ by $\mathbf{x}$ and $\mathbf{b} - A\mathbf{x}_0$ by $\mathbf{b}$. Since (IP$''$ $-$ s_k) has a feasible integer solution and A is a totally unimodular matrix, the optimal extreme point solution to

$$
\begin{aligned}
(\mathrm{RP}' - s_k)\ \ \mathrm{Min}\ & F(s_k \mathbf{y}) \\
& A\mathbf{y} \geq \left\lceil \frac{\mathbf{b}}{s_k} \right\rceil \\
& 2\hat{\mathbf{y}}^{(k-1)} - 2n\mathbf{e} \leq \mathbf{y} \leq 2\hat{\mathbf{y}}^{(k-1)} + 2n\mathbf{e}
\end{aligned}
$$

is also an optimal solution to (IP$''$ $-$ s_k). Note that an optimal solution to (RP$'$ $-$ s_k) can be obtained using the substitution (2.4) and then applying a linear programming algorithm that can identify an optimal extreme point solution (if one exists), to the resulting linearized problem. The value of γ used in the algorithm is one unit larger than in the general case. That is $\gamma = \lceil \log_2 ((m/n) \| \mathbf{b} \|_\infty) \rceil$. This additional unit is required to bring down the size of the interval to the form $2ns\Delta$, as opposed to $4ns\Delta$ as in the general case. This can be done since the proximity theorem used here is a factor of 2 better than the general proximity theorem.

Algorithm 4.2

1. Set $s_k = 2^{\gamma-k}$, $k = 0, \ldots, \gamma$; $\hat{\mathbf{y}}^{(0)} = 0$.
2. For $k = 1, \ldots, \gamma$ obtain an optimal solution $\hat{\mathbf{y}}^{(k)}$ to (IP$''$ $-$ s_k) using substitution (2.4) and solving the resulting linear program.

In the next theorem, $T(4n^2, m, 1)$ is used to denote the running time of the linear programming algorithm as before, except that this linear programming algorithm has to identify an extreme point solution. Otherwise, the solution identified is not necessarily integer, as required for the following theorem:

THEOREM 4.3. *Algorithm* 4.2 *delivers* $\hat{\mathbf{y}}^{(\gamma)}$, *which is an optimal solution to* (*IP*) *with a totally unimodular constraint matrix A. The running time of Algorithm* 4.2 *is*

$$\left\lceil \log_2 \left(\frac{m}{n} \| \mathbf{b} \|_\infty \right) \right\rceil \cdot T(4n^2, m, 1).$$

PROOF. Using Theorem 3.5, it can be shown (as in the proof of Theorem 4.1) that an optimal solution to

$$\begin{aligned} (\mathrm{IP}^{(k)}) \quad & \text{Min } F(\mathbf{x}) \\ & A\mathbf{x} \geq \mathbf{b} \\ & s_{k-1}\hat{\mathbf{y}}^{(k-1)} - ns_{k-1}\mathbf{e} \leq \mathbf{x} \leq s_{k-1}\hat{\mathbf{y}}^{(k-1)} + ns_{k-1}\mathbf{e} \\ & \mathbf{x} \text{ integer} \end{aligned}$$

is also an optimal solution to (IP). The required conclusion is obtained by the observation that $(\mathrm{IP}^{(\gamma)})$ and $(\mathrm{IP}'' - s_\gamma)$ are the same problems, for totally unimodular constraint matrix A.

As for the running time of Algorithm 4.2, it suffices to notice that the linearized version of the integer linear program in Step 2 can be solved by applying a linear programming algorithm. The number of variables is multiplied by a factor of $4n$ and $\Delta = 1$. Since Step 2 is applied γ times with $\gamma \leq \lceil \log_2 ((m/n)\| \mathbf{b} \|_\infty) \rceil$, it follows that the complexity of Algorithm 4.2 is $\lceil \log_2 ((m/n)\| \mathbf{b} \|_\infty) \rceil \cdot T(4n^2, m, 1)$. □

Next we will look at the problem (RP). Note than an optimal solution to

$$\begin{aligned} (\mathrm{RP}') \quad & \text{Min } F(\mathbf{x}) \\ & A\mathbf{x} \geq \mathbf{b} \\ & -2^{\gamma+1}n\Delta\mathbf{e} \leq \mathbf{x} \leq 2^{\gamma+1}n\Delta\mathbf{e} \end{aligned}$$

is also an optimal solution to (RP). We are interested only in an ϵ-accurate optimal solution to (RP). A feasible solution $\hat{\mathbf{x}}$ for (RP) is said to be ϵ-accurate optimal if there exists an optimal solution $\mathbf{x}^*$ to (RP) such that $\| \hat{\mathbf{x}} - \mathbf{x}^* \|_\infty \leq \epsilon$. The following algorithm gives an ϵ-accurate optimal solution for (RP).

Algorithm 4.3

1. Let $K = \lceil \gamma + \log_2 n + \log_2 \Delta - \log_2 \epsilon \rceil + 1$, $s_k = 2^{\gamma-k}$, $k = 0, \ldots, K$, $\hat{\mathbf{x}}^{(0)} = \mathbf{0}$.
2. For $k = 1, \ldots, K$ using the substitution (2.5) and solving the resulting linear program, obtain an optimal solution $\hat{\mathbf{x}}^{(k)}$ to

$$\begin{aligned} (\mathrm{LP}^* - s_k) \quad & \text{Min } F^{L:s_k}(\mathbf{x}) \\ & A\mathbf{x} \geq \mathbf{b} \\ & \hat{\mathbf{x}}^{(k-1)} - 2ns_{k-1}\Delta\mathbf{e} \leq \mathbf{x} \leq \hat{\mathbf{x}}^{(k-1)} + 2ns_{k-1}\Delta\mathbf{e} \end{aligned}$$

THEOREM 4.4. *Algorithm* 4.3 *produces* $\hat{\mathbf{x}}^{(K)}$ *that is an* ϵ-*accurate optimal solution for* (*RP*). *The running time of Algorithm* 4.3 *is*

$$\left\lceil \log_2 \frac{\| \mathbf{b} \|_\infty m\Delta}{\epsilon} \right\rceil \cdot T(8n^2\Delta, m, \Delta).$$

PROOF. Using Theorem 3.8, it can be shown (see the proof of Theorem 4.1) that for each k ($k = 1, \ldots, K$), there exists an optimal solution $\mathbf{x}^{*(k)}$ for (RP) such that

$$\| \hat{\mathbf{x}}^{(k)} - \mathbf{x}^{*(k)} \|_\infty \leq 2ns_k\Delta. \tag{4.1}$$

Since $s_K = 2^{\gamma - K}$, one sees that

$$2ns_K\Delta = 2^{\gamma+1-K}n\Delta \leq \frac{2^{\gamma+1}n\Delta\epsilon}{2^{\gamma+1}n\Delta} = \epsilon. \tag{4.2}$$

Combining (4.1) and (4.2), one sees that

$$\| \hat{\mathbf{x}}^{(K)} - \mathbf{x}^{*(K)} \|_\infty \leq \epsilon.$$

At each application of Step 2, the interval of length $4ns_{k-1}\Delta$ containing each optimal value of x_i, $i = 1, \ldots, n$ is subdivided into a grid of granularity s_k that produces an increase in the number of variables of factor $8n\Delta$ in the linearized version, resulting in a complexity $T(8n^2\Delta, m, \Delta)$. Since Step 2 is applied K times ($K \leq \lceil \log_2 (\| \mathbf{b} \|_\infty \cdot m \cdot \Delta/\epsilon) \rceil$), the complexity of the algorithm is as specified. □

5. *Nonlinear Integer Programming with Fixed Number of Variables*

First, consider the linear integer programming problem with fixed number of variables,

$$\begin{aligned} \text{Min } & \mathbf{cx} \\ & A\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \text{ integer.} \end{aligned}$$

Lenstra has shown that linear integer programming with a fixed number of variables is solvable in polynomial time [12]. The problem solved in [12] is a feasibility problem rather than an optimality problem. In order to find the optimal integer solution, one could use the feasibility procedure as a subroutine, and carry out a binary search on the value of the objective. The number of applications of the subroutine will depend on the logarithm of the bounds on the objective value, especially on $\log_2 \| \mathbf{c} \|_\infty$, and $\log_2 \| \mathbf{b} \|_\infty$.

The same idea as used in the algorithms we present in this paper, that is, relying on the use of proximity theorem to reduce the size of the cube in which the optimal solution is to be found, can be used to provide an alternative approach to solving linear integer programming with fixed number of variables. Moreover, this reduction can also be applied to the convex separable integer programming problem. For the linear case, the procedure simply amounts to solving the LP relaxation,

$$\begin{aligned} \mathbf{cx}^* = \text{Min } & \mathbf{cx} \\ & A\mathbf{x} \geq \mathbf{b}, \end{aligned}$$

and carrying out a complete enumeration on the integer points in the cube $\{\mathbf{x} \mid \mathbf{x}^* - n\Delta\mathbf{e} \leq \mathbf{x} \leq \mathbf{x}^* + n\Delta\mathbf{e}\}$. The number of such points is $(2n\Delta)^n$. The complexity of this procedure is then $T(n, m, \Delta) + n(2n\Delta)^n$. Note that since in this case the number of columns of the matrix A is fixed, Δ is bounded by a polynomial in $\| A \|_\infty$, the largest absolute value of an entry in the matrix. Hence the complexity is polynomial in m and $\| A \|_\infty$, but is independent of $\mathbf{b}$ and $\mathbf{c}$. So if either $\log_2 \| \mathbf{b} \|_\infty$ or $\log_2 \| \mathbf{c} \|_\infty$ is very large compared to $\| A \|_\infty$, then our procedure might yield faster running time.

For the problem with the nonlinear objective we apply Steps 1 and 2 of Algorithm (IP). In Step 3, the integer problem (IP*) is solved by enumerating all $(4n^2\Delta)^n$ points.

6. *Concluding Remarks and Possible Extensions*

The algorithms presented in this paper determine the complexity of nonlinear separable optimization problems. Yet, there are numerous issues involving nonlinear optimization that remain unresolved, and many others that require further study and improvement.

As pointed out earlier, the question of boundedness of the solution vector is critical. It would be important to determine the conditions under which bounds on a solution vector can be derived for nonlinear optimization. This issue is yet unresolved even for unconstrained optimization. Indeed, in order to obtain polynomial algorithms of the type described in this paper for separable convex minimization, one needs only the convexity property, a proximity result, and boundedness of the solution vector in an interval whose length's logarithm is polynomial in the data. Therefore, our algorithms are also applicable to unconstrained separable optimization problems in the presence of polynomial bounds. Consequently, the derivation of a polynomial bound to the optimal solution of such problems is tantamount to solving it in polynomial time.

Another closely related question is that of strong polynomiality when the polyhedron is bounded. We bound the solution by a value that essentially depends on the log of the right-hand side. Tardos' algorithm for linear programming [21] works independently of this value. If we were to apply Tardos' procedure in the nonlinear context, it would amount to discarding one constraint at a time, until a face is identified where the optimal solution lies. On this face, the problem can be solved as unconstrained optimization. But here again, we come up against the difficulty of identifying an optimal solution for an unconstrained optimization problem. A strongly polynomial procedure will have to resolve the question of unconstrained optimization first.

One obvious extension is for the proximity theorem to apply also for nonseparable functions. We use the fact that separable convex functions are directionally convex in the proof. It is possible that such a strong property may not be essential for the proof, and it may suffice for the function to be supermodular. Note though, that even if a proximity theorem is proved, the linearization of nonseparable functions may increase, by at least an exponential factor, the number of variables. Perhaps another approach altogether is needed for nonseparable objective functions.

The optimization problems discussed here are defined over a set of linear inequalities. A closely related problem is the separable optimization over a set of nonlinear inequalities defining a convex set. Note that our proximity theorem's proof does not directly extend to this case.

The algorithms described in the paper are polynomially dependent on Δ in the continuous case and, when the corresponding integer linear program is polynomial, in the integer case also. To be truly polynomial, rather than pseudopolynomial, the running time should depend polynomially on the logarithm of Δ. The value of Δ is important only as a factor in the number of variables in the 0–1 linearized version of the piecewise linear convex functions. Such problems, however, possess a specialized structure where large sets of columns of the constraint matrix are identical. The simplex method, for instance, is adaptable for such setups; at each

iteration, the search for the entering and leaving basic variable can be implemented in time logarithmic in the size of those sets (notice that the reduced-costs coefficients are arranged in each set of identical columns in descending order). There are, however, no known polynomial algorithms that can work in such setups in time logarithmic in the multiplicities of the set of columns. Both the ellipsoid and Karmarkar's algorithms work in a space, the dimension of which is determined by the number of variables.

Even if Δ is polynomial, the running time of the algorithms presented here is polynomial but not strongly polynomial in the right-hand side. This is in contrast to linear programming that is solvable in time that is independent of the magnitude of both the right-hand side and the objective function [21]. In a recent work [7], we show that it is impossible to solve the nonlinear separable convex problem independently of both the right-hand side and the objective, even over totally unimodular constraint matrices and even for the simple allocation problem (which consists of one constraint on the sum of the variables). This "impossibility" is established in the algebraic tree model allowing for the four arithmetic operations, comparisons, and even the floor and ceiling operations.

This study is only a beginning of introducing nonlinear and nonanalytical functions into complexity theory. This topic needs addressing with tools that go beyond those of numerical analysis, in order to obtain truly efficient algorithms that run on digital computers, or delineate the limits of such algorithms.

ACKNOWLEDGMENT. The authors are grateful to Ilan Adler for several stimulating discussions that motivated the work done in this paper.

REFERENCES

1. COOK, W., GERARDS, A. M. H., SCHRIJVER, A., AND TARDOS, E. Sensitivity results in integer linear programming. *Math. Prog. 34* (1986), 251–264.
2. DANTZIG, G. B. *Linear Programming and Extensions.* Princeton University Press, Princeton, N.J., 1963.
3. EDMONDS, J., AND KARP, R. M. Theoretical improvements in the algorithmic efficiency for network flow problems. *J. ACM 19* (1972), 248–264.
4. FRANK, A., AND TARDOS, E. An application of simultaneous approximation in combinatorial optimization. *Combinatorica 7* (1987), 49–66.
5. GRANOT, F., AND SKORIN-KAPOV, J. Some proximity and sensitivity results in quadratic integer programming. Working Paper No. 1207. Univ. British Columbia, Vancouver, British Columbia, Canada, 1986.
6. HELGASON, R., KENINGTON, J., AND LALL, H. A polynomially bounded algorithm for a singly constrained quadratic program. *Math. Prog. 18* (1980), 338–343.
7. HOCHBAUM, D. S. Optimal algorithms for the allocation problem and its extensions. Manuscript, Univ. California, Berkeley, Berkeley, Calif., 1989.
8. HOCHBAUM, D. S., SHAMIR, R., AND SHANTHIKUMAR, J. G. A polynomial algorithm for an integer quadratic nonseparable transportation problem. *Math. Prog.* to appear.
9. HOFFMAN, A., AND WOLFE, P. Minimizing a unimodal function of two integer variables. *Math. Prog. Studies 25* (1985), 76–87.
10. JEROSLOW, R. G. There cannot be any algorithm for integer programming with quadratic constraints. *Op. Res. 21* (1973), 221–224.
11. LAUGHHUNN, D. J. Quadratic binary programming with applications to capital budgeting problems. *Op. Res. 10* (1970), 454–467.
12. LENSTRA, JR., H. W. Integer programming with a fixed number of variables. *Math. Op. Res. 8*, 4 (1983), 508–548.
13. LUSS, H., AND GUPTA, S. Allocation of effort resources among competing activities. *Op. Res. 23* (1975), 360–366.
14. MCCALLUM, JR., C. J. An algorithm for certain quadratic integer programs. Bell Labs Tech. Rep., Bell Laboratories, Holmdel, N.J. (undated, referred to in [6]).

15. MINOUX, M. A polynomial algorithm for minimum quadratic cost flow problems. *Europ. J. Op. Res. 18* (1984), 377–387.
16. MINOUX, M. Solving integer minimum cost flows with separable convex cost objective polynomially. *Math. Prog. Study 26* (1986), 237–239.
17. MONTEIRO, R. C., AND ADLER, I. An extension of Karmarkar-type algorithm to a class of convex separable programming problems with global linear rate of convergence. Manuscript, Univ. California, Berkeley, Berkeley, Calif., ESRC 87-4, Oct. 1987.
18. NEMIROVSKY, A. S., AND YUDIN, D. D. *Problem Complexity and Method Efficiency in Optimization.* Wiley, New York, 1983.
19. PAPADIMITRIOU, C. H., AND STEIGLITZ, K. *Combinatorial Optimization: Algorithms and Complexity.* Prentice-Hall, Englewood Cliffs, N.J., 1982.
20. SHAKED, M., AND SHANTHIKUMAR, J. G. Parametric stochastic convexity and concavity of stochastic processes. *Ann. Inst. Stat. Math.*, to appear.
21. TARDOS, E. A strongly polynomial algorithm to solve combinatorial linear programs. *Op. Res. 34* (1986), 250–256.
22. YAO, D. D., AND SHANTHIKUMAR, J. G. The optimal input rates to a system of manufacturing cells. *INFOR 25* (1987), 57–65.
23. ZIPKIN, P. Simple ranking methods for allocation of one resource. *Manage. Sci. 26* (1980), 34–43.

RECEIVED AUGUST 1988; REVISED JANUARY, JULY, AND NOVEMBER 1989; ACCEPTED DECEMBER 1989