JOURNAL OF OPTIMIZATION THEORY AND APPLICATIONS: Vol. 114, No. 3, pp. 671–694, September 2002 (©2002)

Global Optimization Approach to Unequal Sphere Packing Problems in 3D[1]

A. SUTOU[2] AND Y. DAI[3]

Communicated by P. M. Pardalos

Abstract. The problem of the unequal sphere packing in a 3-dimensional polytope is analyzed. Given a set of unequal spheres and a polytope, the double goal is to assemble the spheres in such a way that (i) they do not overlap with each other and (ii) the sum of the volumes of the spheres packed in the polytope is maximized. This optimization has an application in automated radiosurgical treatment planning and can be formulated as a nonconvex optimization problem with quadratic constraints and a linear objective function. On the basis of the special structures associated with this problem, we propose a variety of algorithms which improve markedly the existing simplicial branch-and-bound algorithm for the general nonconvex quadratic program. Further, heuristic algorithms are incorporated to strengthen the efficiency of the algorithm. The computational study demonstrates that the proposed algorithm can obtain successfully the optimization up to a limiting size.

Key Words. Nonconvex quadratic programming, unequal sphere packing problem, simplicial branch-and-bound algorithm, LP relaxation, heuristic algorithms.

1. Introduction

The optimization of the packing of unequal spheres in a 3-dimensional polytope is analyzed. Given a set of unequal spheres and a polytope, the

[1]The authors thank two anonymous referees for their valuable comments. The work of the second author was partially supported by Grant-in-Aid C-13650444 from the Ministry of Education, Science, Sports, and Culture of Japan.

[2]Member of the Research Staff, Enterprise Server Systems Research Department, Central Research Laboratory, Hitachi, Tokyo, Japan.

[3]Assistant Professor, Department of Bioengineering, University of Illinois at Chicago, Chicago, Illinois.

0022-3239/02/0900-0671/0

objective is to assemble them in such a way that (i) the spheres do not overlap with each other and (ii) the sum of the volumes of the packed spheres is maximized. We note that the conventional 2-dimensional and 3-dimensional packing problems (also called bin-packing problems), which have been extensively studied (Refs. 1–3), are fundamentally different from the problem considered in the present work.

The unequal sphere packing problem has important applications in automated radiosurgical treatment planning (Refs. 4–5). Stereotactic radiosurgery is an advanced medical technology for treating brain and sinus tumors. It uses the Gamma knife to deliver a set of extremely high dose ionizing radiations, called "shots", to the target tumor area (Ref. 6). In good approximation, these shots can be considered as solid spheres. For large or irregular target regions, multiple shots are used to cover different parts of the tumor. However, this procedure results usually in large dose inhomogeneities, due to the overlap of the different shots, and the delivery of large amount of dose to normal tissue arising from the enlargement of the treated region when two or more shots overlap.

Optimizing the number, position, and individual sizes of the shots can reduce significantly both the inhomogeneities and the dose to normal tissue, while simultaneously achieving the required coverage. Unfortunately, since the treatment planning process is tedious, the protocol quality depends heavily on the user experience. Therefore, an automated planning process is desired. To achieve this goal, Wang and Wu et al. (Refs. 4–5) formulated mathematically this planning problem as the packing of spheres into a 3D region with a packing density greater than a certain given level. This packing problem was proved to be NP-complete and an approximate algorithm was proposed (Ref. 4). In this work, we formulate the question as a nonconvex quadratic optimization and present solution methods based on the branch-and-bound technique.

Let K be the number of different radii of the spheres in the given set, and let r_k, $k = 1, \ldots, K$, be the corresponding radii. There are L available spheres for each radius. Therefore, the total number of the spheres in the set is KL. Here, we use a single value of L for simplicity of the presentation. However, this model can be modified easily for the case where different numbers L_k of spheres are available for different radii r_k and the total number of spheres in the given set is $\sum_{i=1}^{K} L_k$.

Let the polytope be given by

$$P = \{(x, y, z) \in R^3 : a_m x + b_m y + c_m z \geq d_m, m = 1, \ldots, M\},$$

for some $M > 0$.

Let L be the maximum number of spheres to be packed. We designate the variables (x_i, y_i, z_i), $i = 1, \ldots, L$, as the location of sphere i in a packing. For

each sphere in the packing, a radius has to be assigned. The variables t_{ik}, $i = 1, \ldots, L$ and $k = 1, \ldots, K$, are used to handle this task,

$$t_{ik} = 1, \qquad \text{if sphere } i \text{ has radius } r_k,$$
$$t_{ik} = 0, \qquad \text{otherwise.}$$

With these preliminaries, the optimization problem can be formulated as follows:

$$\text{(P1)} \quad \max \quad (4/3)\pi \sum_{i=1}^{L} \sum_{k=1}^{K} r_k^3 t_{ik},$$

$$\text{s.t.} \quad (x_i - x_l)^2 + (y_i - y_l)^2 + (z_i - z_l)^2 \geq \left(\sum_{k=1}^{K} r_k t_{ik} + \sum_{k=1}^{K} r_k t_{lk} \right)^2,$$
$$i = 1, \ldots, L-1 \quad \text{and} \quad l = i+1, \ldots, L, \tag{1}$$

$$|a_m x_i + b_m y_i + c_m z_i - d_m| / \sqrt{a_m^2 + b_m^2 + c_m^2}$$
$$\geq \sum_{k=1}^{K} r_k t_{ik}, \qquad i = 1, \ldots, L \text{ and } m = 1, \ldots, M, \tag{2}$$

$$a_m x_i + b_m y_i + c_m z_i - d_m \geq 0,$$
$$i = 1, \ldots, L \text{ and } m = 1, \ldots, M, \tag{3}$$

$$\sum_{k=1}^{K} t_{ik} \leq 1, \qquad i = 1, \ldots, L, \tag{4}$$

$$t_{ik} \in \{0, 1\}, \qquad i = 1, \ldots, L \text{ and } k = 1, \ldots, K. \tag{5}$$

Constraints (1) and (3) respectively ensure that no two spheres overlap with each other and that each sphere is centered within the polytope. Constraints (4) and (5) guarantee that at most one radius is chosen for each sphere; i.e., if $t_{ik} = 1$, then the sphere i with radius r_k, is packed, and if $t_{ik} = 0$, then the sphere i with radius r_k is not packed. Together with (3)–(5), the constraints (2) state that the distance between the center of a sphere and the boundary of the polytope is at least as large as the radius of that sphere. Hence, (2) and (3) force all the spheres to be packed inside the polytope.

Let

$$e_m = \sqrt{a_m^2 + b_m^2 + c_m^2}.$$

By (3), the constraint (2) can be rewritten as

$$a_m x_i + b_m y_i + c_m z_i - d_m \geq e_m \sum_{k=1}^{K} r_k t_{ik}, \qquad \text{for each } i, m. \tag{6}$$

Since the right-hand side of (6) is nonnegative, (6) implies (3). Moreover, the binary 0–1 variables t_{ik} in (5) can be replaced by the inequalities

$$t_{ik}(t_{ik}-1)\geq 0 \quad \text{and} \quad 0\leq t_{ik}\leq 1.$$

Note that $t_{ik}\leq 1$ is implied by the constraint (4). These steps allow the restatement of Problem (P1) as follows:

$$\text{(P2)} \quad \max \quad \sum_{i=1}^{L}\sum_{k=1}^{K} r_k^3 t_{ik},$$

$$\text{s.t.} \quad -(x_i-x_l)^2-(y_i-y_l)^2-(z_i-z_l)^2 + \left(\sum_{k=1}^{K} r_k t_{ik}+\sum_{k=1}^{K} r_k t_{lk}\right)^2 \leq 0,$$

$$i=1,\ldots,L-1 \text{ and } l=i+1,\ldots,L, \tag{7}$$

$$-t_{ik}^2+t_{ik}\leq 0, \quad i=1,\ldots,L \text{ and } k=1,\ldots,K, \tag{8}$$

$$a_m x_i+b_m y_i+c_m z_i-d_m \geq e_m \sum_{k=1}^{K} r_k t_{ik},$$

$$i=1,\ldots,L \text{ and } m=1,\ldots,M, \tag{9}$$

$$\sum_{k=1}^{K} t_{ik}\leq 1, \qquad i=1,\ldots,L, \tag{10}$$

$$t_{ik}\geq 0, \qquad i=1,\ldots,L \text{ and } k=1,\ldots,K. \tag{11}$$

Note that the constant $4/3\pi$ in the original objective function is omitted here. The numbers of variables and quadratic constraints are $(3+K)L$ and $L(L-1)/2+LK$, respectively. The quadratic function in each constraint (7) is neither convex nor concave.

Commonly, methods of solution for the NQP class are designed through linear programming (LP) relaxation, an approach known as the reformulation–linearization technique (Ref. 9). Based on this technique of linearization, Al-Khayyal et al. (Ref. 7) proposed a rectangular branch-and-bound algorithm to solve a class of quadratically constrained programs. Raber (Ref. 8) proposed another branch-and-bound algorithm for the same problem based on the use of simplices as the partition elements and the use of an underestimate affine function, whose value at each vertex of the simplex agrees with that of the corresponding nonconvex quadratic function. This work (Ref. 8) demonstrated that often the simplicial algorithm has a better performance over the rectangular algorithm with respect to the computational time. Interestingly, Raber (Ref. 8) mentioned that both the

simplicial algorithm and the rectangular algorithm exhibit poor performance for the packing problem

$$\max\{t: |t - \|x_i - x_j\|_2^2 \leq 0,\ 1 \leq i < j \leq n,\ x_i, x_j \in [0, 1]^2\},$$

without giving the experimental details. It is obvious that their packing problem is similar to ours, but has much simpler structures for the constraints.

In this paper, we examine a customization of the Raber algorithm tailored to our problem. The investigation of the structure of our problem suggests (i) an efficient simplicial subdivision and (ii) different underestimations of the nonconvex functions. Based on these observations, two variants of the algorithm have been constructed. The discrete nature of the packing enables a heuristic design for obtaining good feasible solutions, an outcome which leads to savings on both computational time and memory size.

The remainder of this paper is organized as follows. In Section 2, we derive the LP relaxation of the problem with respect to the simplicial subdivision. The simplicial branch-and-bound algorithm is presented in Section 3. Section 4 gives the heuristic algorithm, and Section 5 presents two variations of the previous algorithm based on the use of special structures. Section 6 reports the computational results of the proposed branch-and-bound algorithm. The conclusions of the work are presented in Section 7.

2. Linear Programming Relaxation

The construction of the LP relaxation of Problem (P2) is the same as the one developed in Ref. 8. For the sake of a complete description, we outline this procedure below.

Let

$$n = 3L + KL,$$
$$v^T = (x_1, y_1, z_1, \ldots, x_L, y_L, z_L, t_{11}, \ldots, t_{LK}) \in R^n.$$

Here, a^T denotes the transpose of a vector a. First, we write Problem (P2) in the following form:

$$\begin{aligned}
(\text{P2}') \quad \max \quad & c^T v, \\
\text{s.t.} \quad & v^T Q_{il} v \leq 0, \qquad i = 1, \ldots, L-1 \text{ and } l = i+1, \ldots, L, \\
& v^T \hat{Q}_{ik} v + \hat{d}_{ik}^T v \leq 0, \qquad i = 1, \ldots, L \text{ and } k = 1, \ldots, K, \\
& Av \leq b,
\end{aligned}$$

where $Q_{il}, \hat{Q}_{ik} \in R^{n\times n}$, $\hat{d}_{ik} \in R^n$, and where $c \in R^n$ is the coefficient vector of the objective function. The inequalities

$$v^T Q_{il} v \le 0 \quad \text{and} \quad v^T \hat{Q}_{ik} v + \hat{d}_{ik}^T v \le 0$$

correspond to the constraints (7) and (8), respectively. The inequality

$$Av \le b$$

represents all the linear constraints of (9), (10), (11). Furthermore, the matrix Q_{il} can be specified as follows:

$$Q_{il} = \begin{bmatrix} Q_{il}^{xyz} & O \\ O & Q_{il}^{t} \end{bmatrix},$$

where O is a matrix having zero for all entries with appropriate size,

$$Q_{il}^{xyz} = \begin{bmatrix}
\ddots & \vdots & & \vdots & & \vdots & & \vdots & \\
\cdots & -1 & & & \cdots & 1 & & & \cdots \\
 & & -1 & & & & 1 & & \\
\cdots & & & -1 & \cdots & & & 1 & \cdots \\
 & \vdots & & \vdots & \ddots & \vdots & & \vdots & \\
\cdots & 1 & & & \cdots & -1 & & & \\
 & & 1 & & & & -1 & & \\
\cdots & & & 1 & \cdots & & & -1 & \cdots \\
 & \vdots & & \vdots & & \vdots & & \vdots & \ddots
\end{bmatrix} \in R^{3L\times 3L}$$

corresponds to the coefficients of $(x_1, y_1, z_1, \ldots, x_L, y_L, z_L)^T$ in (7) for i and l, and

$$Q_{il}^{t} = \begin{bmatrix}
\ddots & \vdots & & & \vdots & & \vdots & & & \vdots & \\
\cdots & r_1^2 & r_1 r_2 & \cdots & r_1 r_K & \cdots & r_1^2 & r_1 r_2 & \cdots & r_1 r_K & \cdots \\
 & r_1 r_2 & r_2^2 & \cdots & r_2 r_K & & r_1 r_2 & r_2^2 & \cdots & r_2 r_K & \\
 & \vdots & \vdots & \ddots & \vdots & & \vdots & \vdots & \ddots & \vdots & \\
\cdots & r_1 r_K & r_2 r_K & \cdots & r_K^2 & \cdots & r_1 r_K & r_2 r_K & \cdots & r_K^2 & \\
 & \vdots & & & \vdots & \ddots & \vdots & & & \vdots & \cdots \\
\cdots & r_1^2 & r_1 r_2 & \cdots & r_1 r_K & \cdots & r_1^2 & r_1 r_2 & \cdots & r_1 r_K & \cdots \\
 & r_1 r_2 & r_2^2 & \cdots & r_2 r_K & & r_1 r_2 & r_2^2 & \cdots & r_2 r_K & \\
 & \vdots & \vdots & \ddots & \vdots & & \vdots & \vdots & \ddots & \vdots & \\
\cdots & r_1 r_K & r_2 r_K & \cdots & r_K^2 & \cdots & r_1 r_K & r_2 r_K & \cdots & r_K^2 & \cdots \\
 & \vdots & & & \vdots & & \vdots & & & \vdots & \ddots
\end{bmatrix} \in R^{KL\times KL}$$

corresponds to the coefficients of $(t_{11}, \ldots, t_{1K}, \ldots, t_{L1}, \ldots, t_{LK})^T$ in (7) for i and k. Similarly, $\hat{Q}_{ik} \in R^{3L \times 3L}$ and $\hat{d}_{ik} \in R^{KL \times KL}$ can be written as follows:

$$\hat{Q}_{ik} = \begin{bmatrix} O & & \vdots & & O \\ & & 0 & & \\ \cdots & 0 & -1 & 0 & \cdots \\ & & 0 & & \\ O & & \vdots & & O \end{bmatrix},$$

$$\hat{d}_{ik}^T = (0, \ldots, 0, 1, 0, \ldots, 0).$$

Let $\tilde{m}$ be the number of linear constraints in Problem (P2). Denote the polytope defined by these linear constraints as

$$U = \{v \in R^n : Av \leq b\},$$

where

$$A \in R^{\tilde{m} \times n} \quad \text{and} \quad b \in R^{\tilde{m}}.$$

To construct the LP relaxation problem, we need to represent the matrix Q_{il} [resp. $\hat{Q}_{ik}$] by the sum of a positive-semidefinite matrix C_{il} [resp. $\hat{C}_{ik}$] and a negative-semidefinite matrix D_{il} [resp. $\hat{D}_{ik}$]. Usually, a spectrum decomposition achieves this goal. However, we do not need to perform such a task, since the matrices Q_{il} and $\hat{Q}_{ik}$ possess special structures that give the decomposition immediately. It is seen readily that the decompositions

$$Q_{il} = C_{il} + D_{il} = \begin{bmatrix} O & O \\ O & Q_{il}^t \end{bmatrix} + \begin{bmatrix} Q_{il}^{xyz} & O \\ O & O \end{bmatrix}, \tag{12}$$

$$\hat{Q}_{ik} = \hat{C}_{ik} + \hat{D}_{ik} = O + \hat{Q}_{ik} \tag{13}$$

satisfy the desired property.

Now, we consider how to construct our linear programming relaxation problem. Let $S = \{v_0, \ldots, v_n\}$ be an n-simplex with $U \cap S \neq \emptyset$, where v_i, $i = 0, \ldots, n$, are its vertices. Then,

$$S = \left\{ v \in R^n : \sum_{j=0}^{n} \lambda_j v_j, \ \sum_{j=0}^{n} \lambda_j = 1, \lambda_j \geq 0, j = 0, \ldots, n \right\}.$$

Let $W_S \in R^{n \times n}$ be a matrix which consists of the columns $v_j - v_0$, $j = 1, \ldots, n$. Then, each point v in S can be represented as

$$v = v_0 + W_S \lambda, \qquad \text{for some } \lambda \in B = \left\{ \lambda \in R^n : \sum_{j=1}^{n} \lambda_j \leq 1, \lambda_j \geq 0, j = 1, \ldots, n \right\}.$$

Through substitution of v in the constraints of Problem (P2′), we obtain the following two equivalent constraints:

$$(W_S\lambda)^T Q_{il} W_S\lambda + 2v_0^T Q_{il} W_S\lambda + v_0^T Q_{il} v_0 \le 0,$$
$$i = 1, \ldots, L-1 \text{ and } l = i+1, \ldots, L, \tag{14}$$

$$(W_S\lambda)^T \hat{Q}_{ik} W_S\lambda + (2\hat{Q}_{ik} v_0 + \hat{d}_{ik})^T W_S\lambda + v_0^T \hat{Q}_{ik} v_0 + \hat{d}_{ik}^T v_0 \le 0,$$
$$i = 1, \ldots, L \text{ and } k = 1, \ldots, K. \tag{15}$$

By replacing Q_{il} and $\hat{Q}_{ik}$ with (12) and (13), the quadratic term of the left-hand side of each constraint of (14) and (15) is divided into a convex function and a concave function by replacing Q_{il} and $\hat{Q}_{ik}$ with (12) and (13), respectively. The relaxation of Problem (P2′) is constructed by ignoring the convex part and replacing the concave part with a linear underestimate function in (14) and (15), respectively. For such an underestimation, we use the convex envelope of a concave function f with respect to the simplex S, which is an affine function whose value at each vertex of S coincides with that of f. More precisely, for the quadratic constraints (14), we have

$$\begin{aligned}
&(W_S\lambda)^T Q_{il} W_S\lambda + 2v_0^T Q_{il} W_S\lambda + v_0^T Q_{il} v_0 \\
&= (W_S\lambda)^T C_{il} W_S\lambda + (W_S\lambda)^T D_{il} W_S\lambda + 2v_0^T Q_{il} W_S\lambda + v_0^T Q_{il} v_0 \\
&\ge \phi_{Sil}(\lambda) + 2v_0^T Q_{il} W_S\lambda + v_0^T Q_{il} v_0,
\end{aligned} \tag{16}$$

where

$$\phi_{Sil}(\lambda) = \sum_{j=1}^{n} (v_j - v_0)^T D_{il} (v_j - v_0)\lambda_j$$

is the convex envelope of $(W_S\lambda)^T D_{il} W_S\lambda$ with respect to S. In a similar fashion, for the quadratic constraints (15), we have

$$\begin{aligned}
&(W_S\lambda)^T \hat{Q}_{ik} W_S\lambda + (2\hat{Q}_{ik} v_0 + \hat{d}_{ik})^T W_S\lambda + v_0^T \hat{Q}_{ik} v_0 + \hat{d}_{ik}^T v_0 \\
&\ge \psi_{Sik}(\lambda) + (2\hat{Q}_{ik} v_0 + \hat{d}_{ik})^T W_S\lambda + v_0^T \hat{Q}_{ik} v_0 + \hat{d}_{ik}^T v_0,
\end{aligned} \tag{17}$$

where

$$\psi_{Sik}(\lambda) = \sum_{j=1}^{n} (v_j - v_0)^T \hat{D}_{ik} (v_j - v_0)\lambda_j$$

is the convex envelope of $(W_S\lambda)^T \hat{D}_{ik} W_S\lambda$ with respect to S.

Obviously, an upper bound of the objective function of Problem (P3) can be obtained by solving the following LP relaxation problem:

$$\begin{aligned}
(\mathrm{LPR})_S \quad \max \quad & c^T W_S\lambda + c^T v_0, \\
\text{s.t.} \quad & \phi_{Sil}(\lambda) + 2v_0^T Q_{il} W_S\lambda + v_0^T Q_{il} v_0 \le 0, \\
& i = 1, \ldots, L-1 \text{ and } l = i+1, \ldots, L,
\end{aligned} \tag{18}$$

$$\psi_{Sik}(\lambda) + (2\hat{Q}_{ik}v_0 + \hat{d}_{ik})^T W_S\lambda + v_0^T\hat{Q}_{ik}v_0 + \hat{d}_{ik}^T v_0 \leq 0,$$
$$i = 1, \ldots, L \text{ and } k = 1, \ldots, K,$$
$$AW_S\lambda \leq b - Av_0, \qquad \lambda \in B.$$

3. Simplicial Branch-and-Bound Algorithm

The simplicial branch-and-bound algorithm is presented in this section. As mentioned above, we use the algorithm in Ref. 8 as our prototype. However, two heuristics designed for obtaining feasible solutions are embedded.

The branching operation is carried out by dividing the current simplex S into two simplices. Let v_{i^*} and v_{j^*} be two vertices of S satisfying

$$\|v_{i^*} - v_{j^*}\|_2 = \max\{\|v_k - v_{k'}\|_2 \,|\, v_k, v_{k'} \in S\}, \tag{21}$$

where $\|a\|_2$ denotes the 2-norm of a vector a. Define

$$v_M = (1/2)v_{i^*} + (1/2)v_{j^*}.$$

The simplex S is split into two simplices,

$$S^1 = [v_0, v_1, \ldots, v_{i^*-1}, v_M, v_{i^*+1}, \ldots, v_n], \tag{22}$$

$$S^2 = [v_0, v_1, \ldots, v_{j^*-1}, v_M, v_{j^*+1}, \ldots, v_n]. \tag{23}$$

The splitting of the simplices has the property that, for each nested sequence $\{S_q\}$ of simplices,

$$\delta^2(S_q) \to 0, \qquad q \to \infty,$$

where

$$\delta^2(S_q) = \max\{\|v_i - v_j\|_2^2 \,|\, v_i, v_j \in S_q\}.$$

For further details, see Horst (Refs. 10–11). The resulting algorithm is presented below.

Branch-and-Bound Algorithm.

Step 1. Start Heuristic Algorithm 1 to calculate a possible feasible solution v_f and the objective function value $f(v_f)$. If successful, set $LB = f(v_f)$.

Step 1.1. Let $SC = \varnothing$. Construct a simplex S_0 which contains the polytope U. Set

$$SC = \{S_0\}.$$

Solve the problem $(\mathrm{LPR})_{S_0}$. If the problem is infeasible, then the original problem has no solution; stop. Otherwise, let the optimal solution be v_0 and the optimal value be $\mu(S_0)$. Set

$$UB = \mu(S_0).$$

If v_0 is a feasible solution of (P2), stop. Otherwise, start Heuristic Algorithm 2 with v_0 to calculate a feasible solution v_0^* and the value $f(v_0^*)$. If

$$LB < f(v_0^*),$$

then set

$$LB = f(v_0^*), \qquad v_f = v_0^*.$$

Set $k = 0$.

Step 1.2. If $(UB - LB)/UB \leq \epsilon$, then stop.

Step 2. Branching. Split S_k into S_k^1 and S_k^2 according to (21) to (23). Set

$$SC = SC \setminus \{S_k\} \cup \{S_k^1\} \cup \{S_k^2\}.$$

For $j = 1, 2$, run Step 2.1 to Step 2.3.

Step 2.1. Solve Problem $(\mathrm{LPR})_{S_k^j}$. If it is infeasible, set

$$SC = SC \setminus \{S_k^j\}.$$

Otherwise, let the optimal solution be v_k^j and the optimal value be $\mu(S_k^j)$.

Step 2.2. If v_k^j is a feasible solution of (P2), set

$$SC = SC \setminus \{S_k^j\}.$$

Furthermore, if

$$LB < \mu(S_k^j),$$

then set

$$v_f = v_k^j, \qquad LB = \mu(S_k^j).$$

Step 2.3. If v_k^j is not feasible, then run Heuristic Algorithm 2 with v_k^j to calculate a feasible solution $(v_k^j)^*$ and the value $f((v_k^j)^*)$. If

$$LB < f((v_k^j)^*),$$

then set

$$v_f = (v_k^j)^*, \qquad LB = f((v_k^j)^*).$$

Step 3. Bounding. Set

$$SC = SC \backslash \{S \in SC: \mu(S) \leq LB + \epsilon\mu(S)\}.$$

If

$$SC = \emptyset,$$

then stop. Otherwise, select a simplex $\bar{S} \in SC$ such that

$$\mu(\bar{S}) = \max_{S \in SC} \mu(S).$$

Set

$$S_{k+1} = \bar{S}, k = k + 1;$$

go to Step 2.

The details of Heuristic Algorithm 1 and Heuristic Algorithm 2 will be given in Section 4. The parameter ϵ gives the tolerance of the solution obtained by the algorithm. We call the solution obtained from the above algorithm an ϵ-optimal solution. The convergence of the algorithm is guaranteed as follows.

Theorem 3.1. See Ref. 8. If the algorithm generates an infinite sequence $\{v_k\}$, then every accumulation point v^* of this sequence is an ϵ-optimal solution of problem (P1).

4. Heuristic Algorithms

First, we give the details of Heuristic Algorithm 1. The basic idea is to place as many spheres as possible having relatively large radii in the polytope. Let C be a list of the given set of KL candidate spheres, which are ordered such that $r_1 \geq \cdots \geq r_{KL}$. Consider a $3D$-triangle defined by four constraints arbitrarily chosen from those of the polytope P. Designate $P_i, i = 1, \ldots, N$ and $N = \binom{m}{4}$, as these triangles. Fixing a P_i, the algorithm starts by picking a sphere from the top of C. Then, it checks whether the sphere can be located at one of the four corners of the triangle P_i. This procedure is continued for the rest of the spheres on the list until four spheres are placed or the search of the spheres in the list is exhausted. All the spheres packed have to satisfy these conditions:

(i) they touch exactly three sides of the triangle;
(ii) no mutual intersection occurs between each pair of the spheres [see Fig. 1 (a)];
(iii) they satisfy all other constraints on P.

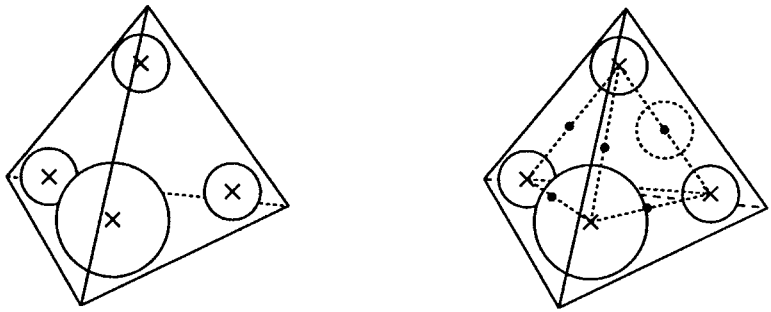

Fig. 1. Initial packing found by Phase I (a); dotted sphere added by Phase II (b).

After obtaining the initial packing, the algorithm attempts to insert more spheres between each pair of the spheres in P_i without violating the packing condition [see Fig. 1(b)].

Heuristic Algorithm 1.

Phase I

Step 0. Set $l = 1, s = 1, k = 0$.

Step 1. Determine the center of the sphere l so that it is tangent to the three planes corresponding to the three constraints of P_s. If the sphere l satisfies the constraints of the polytope P and does not overlap with the spheres $1, \ldots, k-1$, then pack the sphere l and set $k = k + 1$.

Step 2. If (i) $k = L$ or (ii) $l = KL$ and $k > 0$, go to Step 5.

Step 3. If $k = 0$ and $l = KL$, then set $s = s + 1$. If $s = N$, stop. Otherwise, set $l = 1$ and go to Step 1.

Step 4. Set $l = l + 1$, go to Step 1.

Phase II

Step 5. If $k = 1$, stop. Otherwise, for each pair of packed spheres i, j, set $l = 1$; repeat Steps 6 and 7.

Step 6. If $l > KL$ stop. Otherwise, set

$$M = ((x_i + x_j)/2, (y_i + y_j)/2, (z_i + z_j)/2).$$

Step 7. Locate the center of the sphere l at M. If it satisfies the constraints P and does not overlap with the spheres $1, \ldots, k$, then pack the sphere l and set $k = k + 1$. If $k = L$, stop. Otherwise, set $l = l + 1$ and go to Step 6.

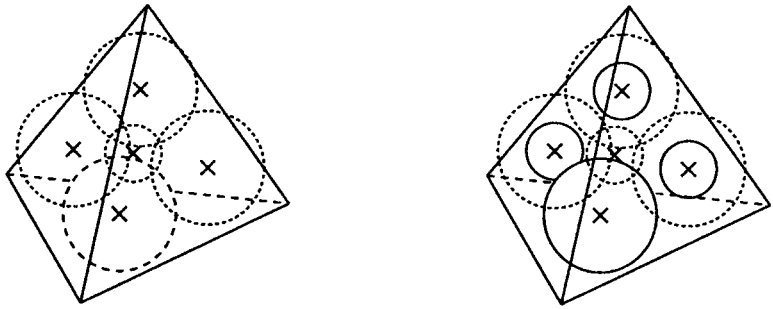

Fig. 2. Infeasible packing given by the relaxation problem (a); Packing of solid spheres found by Heuristic Algorithm 2 (b).

By the termination of the algorithm, if $k > 0$, a feasible packing is obtained.

Next, we describe Heuristic Algorithm 2. Let $(x_1^*, y_1^*, z_1^*, \ldots, x_L^*, y_L^*, z_L^*, t_{11}^*, \ldots, t_{1K}^*)$ be the solution of a relaxation subproblem. Suppose that it is not feasible. Then, either t_{ik}^* is not integral or some spheres overlap with each other if the radius is decided by $t_{ik}^* = 1$ for each i; i.e., use r_k as the radius of the sphere i [see Fig. 2 (a)]. In the following, we fix the centers of these spheres and determine their radii so that they do not mutually overlap [see Fig. 2 (b)]. Note that, for a triplet (x_i^*, y_i^*, z_i^*), if

$$t_{ik}^* = 0, \qquad \text{for all } k = 1, \ldots, K,$$

this means that no sphere is placed there. Let

$$r_1 > \cdots > r_K.$$

Heuristic Algorithm 2.

Step 1. Set $l = 1, k = 1$.

Step 2. If the sphere l centered at (x_l^*, y_l^*, z_l^*) with radius r_k satisfies the polytope constraints and does not overlap with spheres $1, \ldots, l-1$, then set $t_{lk}^* = 1$, $t_{lk'}^* = 0$ for $k' \neq k$ [the sphere l centered at (x_l^*, y_l^*, z_l^*) has radius r_k]; go to Step 4.

Step 3. If $k < K$, set $k = k + 1$; go to Step 2. Otherwise, set $t_{lk'}^* = 0$ for $k' = 1, \ldots, K$ [no sphere is centered at (x_l^*, y_l^*, z_l^*)].

Step 4. If $l < L$, set $l = l + 1$; go to Step 2. Otherwise, stop.

5. Improvement of the Algorithm

In this section, we discuss the special structures of the optimization and present the improvements on the previous algorithm based on these structures. Here, we describe how to construct the initial simplex S_0, which contains the polytope U.

First, choose a nondegenerate vertex v_0 of the polytope U. Then, construct the matrix

$$A' \in R^{n \times n}, \qquad A' = [A'_1, A'_2, \ldots, A'_n],$$

from the n binding constraints at v_0,

$$(A'_i)^T v_0 = b'_i.$$

Set

$$S_0 = \left\{ v: (A')^T v \leq b', \left(-\sum_{i=1}^{n} A'_i \right)^T v \leq \gamma \right\},$$

where

$$\gamma = \max\left\{ \left(-\sum_{i=1}^{n} A'_i \right)^T v: v \in U \right\}.$$

Remark 5.1. The n constraints which decide the nondegenerate vertex v_0 can be selected as follows. For each i, we choose three constraints from (9) and all constraints of (11). It is not difficult to see that $n, n = 3L + KL$, coefficient vectors from these constraints are linearly independent. Set these linear inequality constraints as linear equations, i.e.,

$$a_m x_i + b_m y_i + c_m z_i + d_m = e_m \sum_{k=1}^{K} r_k t_{ik},$$

$$i = 1, \ldots, L \text{ and } m = 1, \ldots, M, \tag{24}$$

$$t_{ik} = 0, \qquad i = 1, \ldots, L \text{ and } k = 1, \ldots, K. \tag{25}$$

The above system of linear equations determines a unique solution which can be considered as v_0.

Remark 5.2. The vertex v_0 has KL zero entries, which are determined by (25). Furthermore, if an equation in (24) is replaced by

$$\left(-\sum_{i=1}^{n} A'_i \right)^T v = \gamma,$$

then the solution obtained maintains

$$t_{ik} = 0, \quad \text{for all } i \text{ and } k.$$

Repeating this process for all the equations in (24) yields $3L$ solutions, which we index as $v_1, \ldots, v_{3L}$. All the other solutions generated by replacing $t_{ik} = 0$, $i = 1, \ldots, L$ and $k = 1, \ldots, K$, are denoted by $v_{3L+1}, \ldots, v_n$. More precisely, we have

$$v_i = (v_{i,1}, \ldots, v_{i,3L}, 0, \ldots, 0)^T, \qquad i = 0, \ldots, 3L, \tag{26}$$

$$\mathrm{v}_i = (\mathrm{v}_{i,1}, \ldots, \mathrm{v}_{i,3L}, 0, \ldots, 0, \mathrm{v}_{i,i}, 0, \ldots, 0), \quad i = 3L+1, \ldots, n. \tag{27}$$

5.1. Splitting a Simplex in a Different Way. It is well known that, in the simplicial branch-and-bound paradigm, both the computational time and the memory usage grow extremely fast as the dimension of the polytope U increases.

When a simplex S is split into two simplices according to (21)–(23), the length of $v_{i^*} - v_{j^*}$ is the longest among all other pairs of vertices of the simplex S. If the vertex v_0 is not replaced by the vertex v_M, then the matrix W_{S^1} is the same as W_S, except the column $v_{i^*} - v_0$, which is replaced by the column $v_M - v_0$. Similarly, all columns in the matrix W_{S^2} are the same as W_S, except the column $v_{j^*} - v_0$, which is replaced by $v_M - v_0$. Recall that the entries (i,j), with $i, j = 3L+1, \ldots, n$, are zeros in the matrices D_{il}, $i = 1, \ldots, L-1$ and $l = i+1, \ldots, L$, in (12). Hence, the coefficients of λ_j in $\phi_{Sil}(\lambda)$ of (18) depend only on the first $3L$ coordinates of $v_0, v_1, \ldots, v_n$. If v_M, v_{i^*}, v_{j^*} have the same values for the first $3L$ coordinates, then $\phi_{Sil}(\lambda)$ remains unchanged in the constraints (18) of the subproblem with respect to the simplices S^j, $j = 1, 2$, and the relaxation quality would not be improved significantly. Such a splitting leads to the computation of subproblems which provide neither good lower bounds nor useful upper bounds. To avoid this situation, we choose v_{i^*}, v_{j^*} such that $\sum_{k=1}^{3L} (v_{i^*,k} - v_{j^*,k})^2$ is the maximum among all other $i-j$ pairs. Consequently, the coordinates corresponding to t_{ik}, $i = 1, \ldots, L$ and $k = 1, \ldots, K$, will not be considered.

One potential difficulty with the convergence of the algorithm must be addressed. Let

$$\tilde{\delta}^2(S_q) = \max\left\{ \sum_{k=1}^{3L} (v_{i,k} - v_{j,k})^2 \;\middle|\; v^i, v^j \text{ are vertices of } S_q \right\}.$$

Since we divide the simplex so as to minimize the largest value $\sum_{k=1}^{3L} (v_{i^*,k} - v_{j^*,k})^2$ over all vertices, it is possible that a nested sequence

$\{S_q\}$ of simplices with $\tilde{\delta}^2(S_q) \to 0$, $q \to \infty$, but $\delta^2(S_q)$ does not satisfy. In this case, we are not guaranteed that the accumulation point of an infinite sequence obtained by the algorithm will be an optimal solution.

5.2. Another Form of the Relaxation. In this section, we focus on a different form of relaxation of Problem (P2). Let us omit the quadratic constraints (8), which are corresponding to the $0-1$ condition of t_{ik}. Furthermore, we allow the overlapping of any two spheres; i.e., we replace the constraints (7) by

$$(x_i - x_l)^2 + (y_i - y_l)^2 + (z_i - z_l)^2 \geq (2\tilde{\epsilon})^2,$$
$$i = 1, \ldots, L-1 \text{ and } l = i+1, \ldots, L. \tag{28}$$

Note that one can take the smallest value among all radii given in the set as the magnitude of $\tilde{\epsilon}$.

The convex envelope of the concave function

$$-(x_i - x_l)^2 - (y_i - y_l)^2 - (z_i - z_l)^2$$

can be determined by the first $3L$ coordinates of the vertices of the corresponding simplex as given above. This implies that it is sufficient to construct simplices in the $3L$-dimensional space. Let

$$S_0' = [v_0', \ldots, v_{3L}']$$

be the initial simplex in the $3L$-dimensional space that contains the polytope P. The ith, $i = 1, \ldots, 3L$, entry of v'_j is identical to that of v_j for $j = 0, \ldots, 3L$. Let

$$B = \left\{ \lambda \in R^{3L} : \sum_{j=1}^{3L} \lambda_j \leq 1, \lambda_j \geq 0, j = 1, \ldots, 3L \right\}.$$

Let Q_{il}'' and A'' be defined similarly as Q_{il} and A, respectively. Let A_x'' and A_t'' be the submatrices of A'' which are corresponding to

$$x = (x_1, y_1, z_1, \ldots, x_L, y_L, z_L)^T \quad \text{and} \quad t = (t_{11}, \ldots, t_{LK})^T,$$

respectively. Then, the following inequality is an equivalent representation of the constraint (28):

$$(W_S\lambda)^T Q_{il}'' W_S\lambda + 2v_0^T Q_{il}'' W_S\lambda + v_0^T Q_{il}'' v_0 \leq -4\tilde{\epsilon}^2,$$
$$i = 1, \ldots, L-1 \text{ and } l = i+1, \ldots, L. \tag{29}$$

Since Q_{il}'' is a negative-semidefinite matrix, the convex envelope of the quadratic term $(W_S\lambda)^T Q_{il}'' W_S\lambda$ is

$$\phi_{Sil}''(\lambda) = \sum_{j=1}^{3L} (v_j - v_0)^T Q_{il}'' (v_j - v_0)\lambda_j.$$

Therefore, we obtain the relaxation Problem (P4) as follows:

$$\max \quad \sum_{i=1}^{L} \sum_{k=1}^{K} r_k^3 t_{ik}$$

$$\text{s.t.} \quad \phi''_{Sil}(\lambda) + 2v_0^T Q''_{il} W_S \lambda + v_0^T Q''_{il} v_0 \leq -4\tilde{\epsilon}^2,$$

$$i = 1, \ldots, L-1 \text{ and } l = i+1, \ldots, L, \tag{30}$$

$$\sum_{k=1}^{K} t_{ik} = 1, \qquad i = 1, \ldots, L, \tag{31}$$

$$0 \leq t_{ik} \leq 1, \qquad i = 1, \ldots, L \text{ and } k = 1, \ldots, K, \tag{32}$$

$$A''_x W_S \lambda + A''_t t \leq b'' - A''_x v_0, \qquad \lambda \in B. \tag{33}$$

Since the constraints (8) are ignored and the constraints (7) are relaxed as (30) in the above problem, the relaxation quality may be inferior to that of the previous methods. However, the dimension of the simplices kept in the memory is only $3L$. Therefore, the total memory used may be far smaller than that required in the other methods.

6. Computational Study

In this section, we discuss details of the implementation of the algorithm and report the experimental results.

6.1. Test Problems. The test problems are generated as follows. First, construct a simplex with vertices

$$(0, 0, 10), (10, 0, 0), (0, 10, 0), (10, 10, 10).$$

Calculate the maximum inscribed sphere of the simplex (the radius is 2.88675). Then, randomly generate m points on the surface of the sphere. Construct the tangent planes,

$$\{(x, y, z) \mid a_i x + b_i y + c_i z = d_i\}, \qquad i = 1, \ldots, m,$$

which pass through each of those m points respectively. Let

$$H_i = \{(x, y, z) \mid a_i x + b_i y + c_i z \geq d_i\}$$

be the halfspace containing the inscribed sphere. The intersection of H_i with the simplex is the 3-dimensional polytope in which the sphere packing problem is considered. Hence, the total number of linear constraints of the polytope is

Table 1. Combinations of radii for $K = 2$.

C_r	radius	
1	1.75	0.75
2	1.50	1.25
3	1.25	1.00
4	1.25	0.50
5	1.00	0.75
6	1.00	0.50
7	1.00	0.25

$$M = m + 4.$$

In our test, m was set at 4. Different pairs of radii of spheres that we used are shown in Tables 1–2.

6.2. Results. The computational experiments were conducted on a DEC Alpha 21164 Workstation (600MHz). We used CPLEX 6.5.1 as an LP solver for the relaxation problems. The limits of memory and computational time were set at 512MB and 3600 seconds, respectively. Besides the prototype algorithm given in Section 3, two variations were also implemented. They were:

(i) Algorithm NSS, which uses the simplex splitting given in Section 5.1;
(ii) Algorithm XYZ, which uses the simplex in the XYZ space given in Section 5.2.

For each algorithm, we tested five instances for each $L = 2, 3, 4$ and each pair C_r of radii from Table 1. Since the other two algorithms reached either the limiting memory or the limiting computational time for most of the instances for $L = 5$, we present only the results of Algorithm XYZ. The

Table 2. Combinations of radii for $K = 3$.

C_r	radius		
1	2.00	1.50	1.00
2	1.75	1.00	0.25
3	1.50	1.00	0.50
4	1.20	0.90	0.60
5	1.00	0.75	0.25

Table 3. Legends used in the tables.

Legend	Meaning
L	Maximum number of the spheres packed
C_r	Combination of the radii
# LP	Number of linear programs solved
# T	Number of cases terminated, caused by the CPU time limit (3600 sec)
# E	Number of cases terminated, when $\epsilon = 10^{-6}$
# M	Number of cases terminated, caused by the memory limit (512 MB)
Time	CPU time (sec)
$(UB - LB)/UB$	Ratio of the values $UB - LB$ and UB, where UB and LB are the upper and lower bounds of the objective function value
Algorithm ORG	Prototype algorithm given in Section 3
Algorithm NSS	Algorithm using the simplex splitting given in Section 5.1
Algorithm XYZ	Algorithm using the simplex in the XYZ space in Section 5.2

legends used in the tables are given in Table 3. The data shown in Tables 4–6 and Table 9 represent the average of the results.

From Tables 4–6, we observe that the computational time grows drastically as the number of spheres packed is increased, since the dimension of the problem is $(3 + K)L$. This outcome is consistent with the observation in Refs. 7–8 that the computational time increases exponentially as the dimension increases. Algorithm NSS solves fewer LPs than Algorithm ORG does; therefore, it needs less time. This leads to the solution of a greater number of instances without violating the limits on either memory or time. Since the simplex splitting is based on the length of the first $3L$ coordinates, it avoids solving unnecessary subproblems.

Among all the algorithms, Algorithm XYZ shows the best performance. All instances for $L = 2, 3, 4$ are solved, except the case $C_r = 1$, $L = 4$. It even obtained successfully the solutions for about half of the instances for $L = 5$. Two reasons for this behavior are considered. First, since the dimension of the simplices is only $3L$, much less memory is needed to keep the information on the simplices. Second, since the simplices involve only the coordinates of the variables (x, y, z), the simplex splitting has the favorable characteristic of Algorithm NSS that fewer LPs are needed to be solved.

It should be noted that:

(i) the dimension of the instances solved in the previous papers (Refs. 7–8) range up to 16;

(ii) usually, the computational time and memory demand of the simplicial branch-and-bound algorithm increases exponentially with the dimension of the problem.

Therefore, a modest increase in the dimension could put the solution out of range. In contrast, for the sphere packing problem discussed above, the

Table 4. Results of Algorithm ORG.

L	C_r	# LP	$(UB-LB)/UB$	Time	# E	# T	# M
2	1	4809	0.00	8.32	5	0	0
	2	1	0.00	0.00	5	0	0
	3	1	0.00	0.00	5	0	0
	4	1	0.00	0.00	5	0	0
	5	2670	0.00	4.42	5	0	0
	6	3164	0.00	8.56	5	0	0
	7	2542	0.00	4.24	5	0	0
3	1	270,542	0.13	2505.31	3	2	0
	2	7617	0.0	28.27	5	0	0
	3	2530	0.00	7.63	5	0	0
	4	8511	0.00	27.51	5	0	0
	5	18,585	0.00	61.39	5	0	0
	6	13,494	0.00	44.06	5	0	0
	7	15,625	0.00	49.27	5	0	0
4	1	221,050	0.35	3019.95	1	4	0
	2	139,077	0.00	865.56	4	1	0
	3	109,981	0.07	1564.78	3	2	0
	4	148,904	0.14	1947.72	3	2	0
	5	109,476	0.08	1554.34	3	2	0
	6	149,203	0.14	2235.52	2	3	0
	7	207,166	0.15	2730.21	2	3	0

dimension of the instances solved by Algorithm XYZ extended up to $n = 25$, when $L = 5$ and $K = 2$. Hence, we conclude that Algorithm XYZ gains efficiency, since it takes advantage of the intrinsic structure of the problem.

Next, let us focus on Algorithm ORG and Algorithm XYZ and consider the influence of the polytope P on their performance. Five instances of the polytope are generated randomly and they have same number of constraints. Taking $L = 4$, $C_r = 2$, the results are shown in Tables 7–8. It is observed that, even if L and C_r are identical, the comparative behaviors of the algorithms are very sensitive to the shape of the polytope. This behavior arises since:

(i) the quality of the solutions generated by the heuristics depends on the shape of the polytope;
(ii) the initial simplex S_0 depends on the polytope.

For a skinny polytope, the volume of $S_0 \backslash P$ would be large, a fact which means that the algorithms waste effort copiously on solving LPs defined on this zone before reaching the optimal solution.

Table 5. Results of Algorithm NSS.

L	C_r	# LP	$(UB-LB)/UB$	Time	# E	# T	# M
2	1	1574	0.00	3.08	5	0	0
	2	1	0.00	0.00	5	0	0
	3	1	0.00	0.00	5	0	0
	4	1	0.00	0.00	5	0	0
	5	215	0.00	0.32	5	0	0
	6	228	0.00	0.48	5	0	0
	7	299	0.00	0.45	5	0	0
3	1	212,048	0.19	2424.45	2	3	0
	2	35	0.00	0.09	5	0	0
	3	1685	0.00	4.52	5	0	0
	4	1067	0.00	2.85	5	0	0
	5	7243	0.00	25.04	5	0	0
	6	7645	0.00	26.36	5	0	0
	7	1671	0.00	4.34	5	0	0
4	1	222,788	0.28	3019.95	1	4	0
	2	112,190	0.04	1397.28	3	1	1
	3	56,887	0.07	710.24	4	0	1
	4	111,017	0.05	1305.93	4	0	1
	5	30,829	0.08	209.47	5	0	0
	6	65,880	0.04	833.36	4	1	0
	7	63,520	0.00	520.50	5	0	0

Table 9 shows the results when three values of radii are considered, i.e., $K = 3$. Note that:

(i) The value of K affects the number of the quadratic constraints in (8).

(ii) Increasing the number of quadratic constraints enhances the difficulty of the problem. Comparing this case with the results for $K = 2$ (Table 6), we observed that there is no big change in the numbers of LPs solved. The algorithm is less sensitive to the value of K than to the size of L. This occurs, since the numbers of quadratic constraints (7) and (8) grow with increasing L.

Finally, all of the algorithms exhibited reduced performance when $C_r = 1$, namely, when the differences of the radii are large.

7. Conclusions

In this paper, we considered the optimization of unequal sphere packing and demonstrated the improvements over the existing simplicial branch-and-bound algorithm through advantageous use of the intrinsic structures

Table 6. Results of Algorithm XYZ.

L	C_r	# LP	$(UB-LB)/UB$	Time	# E	# T	# M
2	1	577	0.00	0.70	5	0	0
	2	1	0.00	0.00	5	0	0
	3	1	0.00	0.00	5	0	0
	4	1	0.00	0.00	5	0	0
	5	164	0.00	0.19	5	0	0
	6	485	0.00	0.70	5	0	0
	7	293	0.00	0.33	5	0	0
3	1	147,821	0.06	1432.87	4	1	0
	2	11	0.00	0.00	5	0	0
	3	793	0.00	1.38	5	0	0
	4	1603	0.00	2.85	5	0	0
	5	1827	0.00	3.04	5	0	0
	6	1734	0.00	2.86	5	0	0
	7	1617	0.00	2.67	5	0	0
4	1	267,940	0.32	3600.00	0	5	0
	2	49,365	0.00	448.67	5	0	0
	3	33,103	0.00	124.17	5	0	0
	4	68,588	0.00	532.67	5	0	0
	5	20,487	0.00	94.53	5	0	0
	6	22,199	0.00	108.72	5	0	0
	7	67,332	0.00	630.07	5	0	0
5	1	266,880	0.46	3600.00	0	5	0
	2	86,677	0.03	927.39	4	1	0
	3	199,838	0.06	2660.20	2	3	0
	4	178,181	0.11	2367.46	2	3	0
	5	156,760	0.05	1996.32	3	2	0
	6	164,904	0.06	2089.97	3	2	0
	7	165,105	0.11	2296.86	2	3	0

Table 7. Results of Algorithm ORG for $L = 4$, $C_r = 2$.

P	# LP	$(UB-LB)/UB$	Time
1	5981	0.00	25.10
2	28,563	0.00	134.32
3	1	0.00	0.02
4	590,915	0.52	3600.03
5	69,929	0.00	568.33

of the problem. Specially, the computational study showed that the improved Algorithm XYZ could solve instances with much larger size. We observed that optimal solutions were found for many instances when the algorithm reached the limitations of either computational time or memory.

Table 8. Results of Algorithm XYZ for $L = 4$, $C_r = 2$.

P	# LP	$(UB - LB)/UB$	Time
1	27,453	0.00	8.47
2	210,891	0.00	2578.08
3	1511	0.00	3.20
4	71,329	0.00	393.70
5	25,477	0.00	86.90

Table 9. Results of Algorithm XYZ for $K = 3$.

L	C_r	# LP	$(UB - LB)/UB$	Time	# E	# T	# M
3	1	308,570	0.37	3600.00	0	5	0
	2	123,418	0.05	1122.95	4	1	0
	3	11	0.00	0.02	5	0	0
	4	502	0.00	0.96	5	0	0
	5	1591	0.00	2.81	5	0	0
4	1	218,901	0.50	3600.00	0	5	0
	2	261,260	0.30	3600.00	0	5	0
	3	46,204	0.00	235.28	5	0	0
	4	79,014	0.02	124.17	4	1	0
	5	47,947	0.00	414.87	5	0	0

This signals that the algorithm spends a large amount of effort verifying the optimality of the solution. In turn, this behavior indicates that: (a) developing an improved method of relaxation is necessary for solving problems with a larger size; (b) when the algorithm is terminated, the solution obtained can be of high quality; and (c) a better heuristic method, which could start with a feasible solution obtained from the branch-and-bound algorithm, would be very important for obtaining the approximate solution to larger problems.

References

1. Coffman, E. G., jr., Garey, M. R., and Johnson, D. S., *Approximation Algorithms for Bin-Packing*: *An Updated Survey*, Algorithm Design for Computer System Design, Edited by G. Ausiello, M. Lucertini, and P. Serafini, Springer Verlag, New York, NY, pp. 49–106, 1984.
2. Li, K., and Cheng, K. H., *On Three-Dimensional Packing*, SIAM Journal on Computing, Vol. 19, pp. 847–867, 1990.
3. Steinberg, A., *A Strip-Packing Algorithm with Absolute Performance Bound* 2, SIAM Journal on Computing, Vol. 26, pp. 401–409, 1997.

4. WANG, J., *Packing of Unequal Spheres and Automated Radiosurgical Treatment Planning*, Journal of Combinatorial Optimization, Vol. 3, pp. 453–463, 1999.
5. WU, Q. J., and BOURLAND, J. D., *Morphology-Guided Radiosurgery Treatment Planning and Optimization for Multiple Isocenters*, Medical Physics, Vol. 26, pp. 2151–2160, 1992.
6. WU, A., *Physics and Dosimetry of the Gamma Knife*, Neurosurgery Clinics of North America, Vol. 3, pp. 35–50, 1992.
7. AL-KHAYYAL, F. A., LARSEN, C., and VAN VOORHIS, T., *A Relaxation Method for Nonconvex Quadratically-Constrained Quadratic Programs*, Journal of Global Optimization, Vol. 6, pp. 215–230, 1995.
8. RABER, U., *A Simplicial Branch-and-Bound Method for Solving Nonconvex All-Quadratic Programs*, Journal of Global Optimization, Vol. 13, pp. 417–432, 1998.
9. SHERALI, H. D., and TUNCBILEK, C. H., *A New Reformulation–Linearization Technique for Bilinear Programming Problems*, Journal of Global Optimization, Vol. 2, pp. 101–112, 1992.
10. HORST, R., *On Generalized Bisection of n-Simplices*, Mathematics of Computation, Vol. 66, pp. 691–698, 1997.
11. HORST, R., and PARDALOS, P. M., *Handbook of Global Optimization*, Kluwer Academic Publishers, Dordrecht, Holland, 1995.