International Journal of Management Science
Vol 3, No 1, May 1997

DEVELOPMENT OF A TABU SEARCH HEURISTIC FOR SOLVING MULTI-OBJECTIVE COMBINATORIAL PROBLEMS WITH APPLICATIONS TO CONSTRUCTING DISCRETE OPTIMAL DESIGNS

JOO SUNG JUNG
Department of Manpower Management,
Korea Institute for Defense Analyses

BONG JIN YUM
Department of Industrial Engineering,
Korea Advanced Institute of Science and Technology

ABSTRACT

Tabu search (TS) has been successfully applied for solving many complex combinatorial optimization problems in the areas of operations research and production control. However, TS is for single-objective problems in its present form. In this article, a TS-based heuristic is developed to determine Pareto-efficient solutions to a multi-objective combinatorial optimization problem. The developed algorithm is then applied to the discrete optimal design problem in statistics to demonstrate its usefulness.

1. INTRODUCTION

Various complex combinatorial optimization problems arise in engineering and science fields. Obtaining an exact optimal solution for such complex combinatorial optimization problems is very difficult, and therefore, a heuristic method is usually employed instead. Tabu Search (TS) has recently emerged as a promising metaheuristic for solving such complex combinatorial problems. Unlike other search algorithms (e.g., a steepest descent or ascent algorithm), TS employs a unique scheme for alleviating the risk of being trapped at a local optimum. Many successful results of TS have been reported in the areas of operations research and production control (e.g., Glover et al. [6]). However, little application has been made for solving various combinatorial optimization problems in statistics. One exception is Jung and Yum [7] in which TS was successfully applied for solving single-objective discrete optimal design problems.

An optimization problem is usually formulated based upon a single objective for mathematical tractability, although a multi-objective formulation is more realistic. The current TS heuristic is designed for single-objective combinatorial problems. In this article, we first extend the TS strategy to the case of multiple objectives, and then, develop a TS-based heuristic for solving the discrete optimal design problem with two popular criteria. To the best of our knowledge, no systematic approach has been developed for solving multi-objective discrete optimal design problems.

2. PRELIMINARIES

2.1 Multi-Objective Optimization

A combinatorial optimization problem with multiple objectives can be described as follows.

$$\textit{Minimize} \quad \{F(s) = (f_1(s), \cdots, f_w(s)) \mid s \in S\}$$

where

$f_t(s)$: objective functions, $t = 1, 2, \cdots, w$,

S : finite solution space,

s : feasible solution (a vector of discrete values).

Typical solution approaches for a multi-objective problem include the following (Chankong and Haimes [2]).

1. Reduction to a problem with a single objective obtained as a combination of the given multiple objectives.
2. Transforming to a single-objective constrained problem where all but one objective serve as the constraints.
3. Employing a direct approach which considers all the objectives simultaneously.

Among the above three, the direct approach is considered in this article.

Due to the conflicting nature of the objectives, multi-objective problems rarely have solutions that simultaneously minimize all of the objectives. Instead, they have several solutions called Pareto-efficient solutions whose property is such that no improvement in any one objective is possible without sacrificing some other objectives. A Pareto-efficient solution is also called a nondominated or noninferior solution. In this article we are concerned with finding Pareto-efficient solutions for the discrete optimal design problem with two objectives. Once Pareto-efficient solutions are found, a best compromise solution can be determined based upon the decision maker's utility or preference function [2].

2.2. Dominant and Pareto-efficient Solutions

One property that is commonly considered as necessary for any candidate solution to a multi-objective optimization problem is that the solution is not dominated. For a minimization problem, a feasible solution s_1 is dominated by another feasible solution s_2 if and only if

$$f_t(s_1) \geq f_t(s_2), \qquad t = 1, 2, \cdots, w,$$

and

$$f_t(s_1) > f_t(s_2)$$

for at least one t. A feasible solution s^* is Pareto-efficient if and only if there exists no other feasible solutions which dominate it.

Let $s_t^* \in S$ be the unique optimal solution for the objective f_t ($t = 1, 2, \cdots, w$). Then s_t^* is also a Pareto-efficient solution to the given multi-objective optimization problem. Hence, if there exists a unique optimal solution for each of the given objectives, then the minimal number of Pareto-efficient solutions to the given multi-objective optimization problem equals the number of different optimal solutions to single-objective problems. The maximum number of Pareto-efficient solutions is theoretically the number of all feasible solutions. The actual number of Pareto-efficient solutions lies between the above two extremes. For more detailed description of dominant and Pareto-efficient solutions, the reader is refered to Chankong and Haimes [2] and Ringuest [10].

2.3. Basic Features of TS

In this section, we describe the basic features of TS as proposed by Glover [3] for a single-objective combinatorial optimization problem.

TS is a metaheuristic in the sense that it iteratively explores the solution space with another local heuristic search procedure being used at each step. In the process of iterations, let s be the current solution. Then, a typical descent algorithm generates a neighborhood $U(s)$ of s, makes a search over $U(s)$ to find a best solution s', and moves from s to s' for the next iteration if

$f(s') < f(s)$ for a single-objective minimization problem and stops otherwise. One inherent problem to the above procedure is that it will in general lead to and be trapped at a local minimum of f. In TS, various features are provided to overcome such a difficulty.

To escape from a local minimum, TS accepts a move from s to s', even though $f(s') > f(s)$. However, such an acceptance of a disimproved solution may result in cycling. To avoid such cycling, TS maintains a memory structure called a tabu list T which keeps track of solution transitions over some previous iterations. Then, T is used at the current iteration to forbid certain moves that may lead to previously visited solutions. One possible way of constructing T is to record t most recent solutions. This will prevent cycles of length less than (or equal to) t from occurring in the course of iterations. However, keeping track of t previous solutions and checking whether or not a solution matches with any one of them could be very time- and space-consuming. Therefore, some modifications to the above definition of T are usually made. Glover [4, 5] suggests that the tabu conditions be based on selected attributes of moves. The tabu size t is usually fixed, although it could be dynamically varied in the course of iterations.

Although the tabu conditions based on some selected attributes of moves are efficient in terms of time and space, it could be too restrictive in the sence that it may also forbid moves to unvisited solutions of possibly better quality. It is thus desirable in certain situations to cancel the tabu status of moves. This is performed by means of aspiration level conditions. For instance, let $A(f(s))$ be an aspiration level of an objective function value next to be reached when the current value is $f(s)$, and consider a move m from the current solution s to s'. Then, an aspiration level condition may be described as $f(s') < A(f(s))$ for a minimization problem. If this condition is satisfied, m may be regarded as a legitimate move, even though it is in tabu status. As $A(f(s))$, $f(s^*)$ is frequently adopted where s^* is the best solution obtained up to the current iteration. For a more general scheme for aspiration level conditions, the reader is referred to Glover [4 ,5], Glover et al. [6], and de Werra and Hertz [11].

Another useful feature of TS is the so called long-term memory [6], which allows TS to explore new regions of the solution space. For a given optimization problem, define a 'try' as a sequence of iterations until the stopping condition is met. When an optimization problem is solved several times to obtain a better solution, the optimum could be reached faster if a purposeful alternate starting solution is employed instead of a randomly chosen one at each try. For an alternate starting solution, we may use a long-term memory function and generate a new starting solution from the region least frequently visited in the previous tries.

One try of TS may terminate based upon some combination of the following conditions [6]:

1. A solution which gives the known optimal f is found.
2. The neighborhood U is empty.
3. The total number of iterations exceeds a predetermined maximum.
4. A predetermined number of iterations is performed without improvement since the last iteration in which improvement is made.

3. TS-BASED DIRECT HEURISTIC FOR MULTI-OBJECTIVE COMBINATORIAL PROBLEMS

The proposed TS-based direct heuristic (TSDH) for multi-objective combinatorial optimization problems is based on the properties of Pareto-efficient solutions and the basic idea of TS. It also belongs to the class of direct algorithms since all the objectives are considered simultaneously.

3.1. Basic Features of the Proposed Algorithm

From the definition of a Pareto-efficient solution, we know that the optimal solution to each single-objective problem, min $f_t(s)$, $s \in S$, $t = 1, 2, \cdots, w$, is a Pareto-efficient solution. Hence, the initial set of Pareto-efficient solutions to the multi-objective problem consists of these optimal solutions, each of which can be found by TS. However, we cannot guarantee that the solution obtained by TS is always the global optimal solution, and therefore, we will call the initial set a near Pareto-efficient solution set (NPE-set).

In a try for solving the given multi-objective combinatorial problem, we iteratively search for a best possible dominant solution beginning with a starting solution. In the first try, a starting solution is obtained by random generation, and for successive tries new starting solutions are generated from the region investigated least frequently.

In a given try, let s be the current solution and $s*$ be the dominant solution obtained up to the current iteration. In the neighborhood of $U(s)$, consider a solution s'. If s' dominates s^*, we accept s' and make a move from s to s' for the next iteration whether or not s' is in tabu status. This is based upon the idea of aspiration level condition. In finding such an s' over

$U(s)$, we may take the one encountered first for computational efficiency instead of examining the entire $U(s)$. If there does not exist s' in $U(s)$ which dominates s^*, consider two cases. If all solutions in $U(s)$ are not dominated by $s*$ (i.e., if all solutions in $U(s)$ are indeterminate with respect to $s*$), then this procedure stops. Otherwise, let $E(s)$ be the set of solutions in $U(s)$ which are dominated by $s*$ and are not in tabu status. Then, we select a best s' from $E(s)$, and move from s to s' for the next iteration. Acceptance of such non-improved solutions is to overcome the possibility of being trapped at a locally efficient solution.

The size of the tabu list is fixed in the present study. Maintaining the tabu list is the same as that for a single-objective problem. Aspiration functions $A(F(s*)) = (f_1(s*), f_2(s*), \cdots, f_w(s*))$ are the values of the objectives for the dominant solution s^* obtained up to the current iteration.

If a predetermined number of iterations (NEMAX) are performed without getting another dominant solution after the last iteration in which a dominant solution is found, then the current try is terminated, and the best dominant solution s^* is taken as a candidate for a (near) Pareto-efficient solution. In accordance with the definition of a Pareto-efficient solution, if there does not exist a (near) Pareto-efficient solution which dominates s^*, then s^* is included in the NPE-set and the existing (near) Pareto-efficient solutions dominated by s^* are excluded from the NPE-set. With the above procedure, the NPE-set approaches the true set of Pareto-efficient solutions as the number of tries increases.

3.2. The TSDH Algorithm

Step 0. Determine a (near) optimal solution to each single-objective problem by TS and put them into the NPE-set.

Step 1. Input TRYMAX (the maximum number of tries for generating candidate (near) Pareto-efficient solutions for a given multi-objective problem) and NEMAX. TRY $= 1$

Step 2. Choose a starting solution s (If TRY $= 1$, choose it at random. Otherwise, generate it from the region least frequently investigated).

s^* (current near Pareto-efficient solution) $= s$.

T (tabu list) $= \{\emptyset\}$.

NE (iteration number) $= 0$.

NEBEST (the last iteration in which an s^* is obtained) $= 0$.

Initialize the aspiration function: $A(F) = (f_1(s*), f_2(s*), \cdots, f_w(s*))$.

Step 3. NE = NE+1.
Generate a neighborhood $U(s)$, the set of solutions obtained by applying modifications to s.

Step 4. If in $U(s)$ there exists s' which dominates s^*, then accept s' and go to Step 5. Otherwise, consider two cases. If all solutions in $U(s)$ are not dominated by s^*, then go to Step 8. Otherwise, let $E(s)$ ($\subset U(s)$) be the set of solutions which are dominated by $s*$ and are not in tabu status. Select a best s' from $E(s)$ and go to Step 6.

Step 5. $s^* = s'$. $f_t(s^*) = f_t(s')$, $t = 1, 2, \cdots, w$.
NEBEST = NE.
Update $A(F)$ as $(f_1(s*), f_2(s*), \cdots, f_w(s*))$.

Step 6. Update T. $s = s'$.

Step 7. If (NE -NEBEST) > NEMAX, then go to Step 8.
Otherwise, go to Step 3.

Step 8. If, in the NPE-set, there does not exist a solution which dominates s^*, then include s^* in the NPE-set and exclude solutions dominated by s^* from the NPE-set.

Step 9. TRY = TRY + 1.
If TRY > TRYMAX, then stop.
Otherwise, go to Step 2.

4. TSDH FOR THE DISCRETE OPTIMAL DESIGN PROBLEM

4.1. The Problem

Consider the regression model

$$y_i = h'(x_i)\beta + \varepsilon_i, \quad i=1,2,\cdots,n \tag{1}$$

where

$x_i' = (x_{i1}, x_{i2}, \cdots, x_{ip})$,

$y_i =$ an observation corresponding to a $(p\times 1)$ vector x_i from a finite design space χ,

$h'(x_i) = (h_1(x_i), h_2(x_i), \cdots, h_k(x_i))$, $h_1(x_i) = 1$,

$\beta' = (\beta_1, \beta_2, \cdots, \beta_k)$,

ε_i = uncorrelated random error with mean 0 and constant variance σ^2.

Let X be an $(n \times k)$ matrix of rank k with row i containing $h'(x_i)$. Then, the least squares estimator of β is given by $\hat{\beta} = (X'X)^{-1}X'y$, with the covariance matrix $Cov(\hat{\beta}) = \sigma^2(X'X)^{-1}$. The predicted response at an x is $\hat{y}(x) = h'(x)\hat{\beta}$ with $Var(\hat{y}(x)) = d(x) = \sigma^2 h'(x)(X'X)^{-1} h(x)$.

As an illustration, consider a quadratic regression model with $p = 4$, and assume that each factor x_j $(j = 1, 2, 3, 4)$ takes three values, -1, 0, and 1. Then,

$$h'(x_i) = (1, x_{i1}, \cdots, x_{i4}, x_{i1}x_{i2}, \cdots, x_{i3}x_{i4}, x_{i1}^2, \cdots, x_{i4}^2),$$
$$\chi = \{x \mid x = (x_1, x_2, \cdots, x_4), x_j = -1, 0, 1 \text{ for } j = 1, 2, \cdots, 4\},$$
$$X = \begin{bmatrix} 1 & x_{11} & \cdots & x_{14} & x_{11}x_{12} & \cdots & x_{13}x_{14} & x_{11}^2 & \cdots & x_{14}^2 \\ 1 & x_{21} & \cdots & x_{24} & x_{21}x_{22} & \cdots & x_{23}x_{24} & x_{21}^2 & \cdots & x_{24}^2 \\ & & & & & \vdots & & & & \\ 1 & x_{n1} & \cdots & x_{n4} & x_{n1}x_{n2} & \cdots & x_{n3}x_{n4} & x_{n1}^2 & \cdots & x_{n4}^2 \end{bmatrix}.$$

For model (1), the traditional problem of constructing a discrete optimal design can be described as selecting n design points $\{x_i, i = 1, 2, \cdots, n\}$, not necessarily distinct, from N candidate points in χ such that any one of the following criteria is optimized (Note that $N = 3^4 = 81$ in the above example).

1. D-optimality: Maximize $|X'X|$
2. A-optimality: Minimize $tr(X'X)^{-1}$
3. E-optimality: Minimize the maximal eigenvalue of $(X'X)^{-1}$
4. G-optimality: Minimize the maximum $d(x)$ for $x \in \chi$
5. V-optimality: Minimize $\sum_{i=1}^{N} d(x_i) / N$

The above optimality criteria have the following statistical interpretations. A D-optimal, A-optimal, and E-optimal designs minimize the generalized variance, the average variance, and the maximum variance of the estimators of the components of β, respectively. A G-optimal and V-optimal designs minimize the maximum variance and the average variance of the estimated responses, respectively. For a review of the theory of optimum design, the reader is referred to Atkinson and Donev [1], and Pukelsheim [9].

The above single-objective approaches cannot deal with various practical aspects of a design. In this article we apply the proposed TSDH for solving the discrete optimal design problem with the D-optimality and V-optimality criteria. In other words, for two objective functions $|X'X|$ and $\sum_{i=1}^{N} d(x_i)/N$ under model (1), we want to determine (near) Pareto-efficient solutions, each of which consists of n design points, not necessarily distinct, from N candidate points in χ.

4.2. Description of the Proposed Algorithm

Initial NPE-set

For a given multi-objective discrete optimal design problem, we first obtain a (near) D-optimal design and a (near) V- optimal design, each as a best design obtained from 10 tries of TS. Next, we calculate the D and V values corresponding to the (near) D-optimal and the (near) V-optimal design, respectively. Then, (Dmax, V) and (D, Vmin) are included in the initial NPE-set where Dmax and Vmin are the D and V values for the (near) D-optimal and the (near) V-optimal design, respectively. If Dmax = D and V = Vmin, then we put any one of the two designs into the initial NPE-set.

Determination of candidate (near) Pareto-efficient designs

The starting designs are generated by the following methods. The starting design for the first try is obtained by random generation, and each of the successive starting designs consists of those points that have been least frequently included in the design during the previous tries. The latter scheme is based on the concept of the long-term memory.

The following neighborhood generation and search procedure follows the DETMAX strategy (e. g., see Mitchell [8]) which is cheaper than others. Although the DETMAX strategy was originally developed for generating D-optimal designs, a wide range of empirical studies have also demonstrated its superior performance for generating V-optimal designs.

Let X_N be the design matrix composed of all N candidate points. At the ith iteration for a given try, let X_i be the current design and X^* be the current (near) Pareto-efficient design. A neighborhood $U(X_i)$ is defined as the set of all possible exchanges of a point in X_i with a point in X_N. The search over $U(X_i)$ is then initiated by first adding to the current design matrix X_i

the point (say, the cth row) of X_N at which $d(x)$ is maximum, and then subtract from the resulting (n+1)-point design the point (say, the dth row) at which $d(x)$ is minimum. If this exchange results in a design which dominates X^* whether or not the exchange is in tabu status, the ith iteration stops and X_{i+1} is set to the design matrix of the dominant design. Then, X^* is updated, and the next iteration begins. Otherwise, add again to X_i another row (say, the c'th row, $c' \neq c$) in X_N at which $d(x)$ is maximum, and then, subtract from the resulting (n+1)-point design the point at which $d(x)$ is minimum. If this exchange does not also give a design which dominates X^*, then the above process is repeated. If there does not exist any exchange that yields a design which dominates X^* until all rows of X_N is considered, then consider two cases. If all exchanges generated by the above process give designs which are not dominated by X^*, then this procedure stops. Otherwise, let $E(X_i)$ ($\subset U(X_i)$) be the set of designs which are dominated by X^* and are not in tabu status. Then, select X_{best} as X_{i+1} where X_{best} is the best design from $E(X_i)$ subject to tabu restriction, and start the next iteration.

The tabu list maintains the row numbers of X_N which correspond to the points dropped from the design, and the tabu list size t is set to 7.

Aspiration function $A(F)$ consists of the D and V Values for the dominant design X^* obtained up to the current iteration. At iteration i, suppose an exchange is in tabu status. If that exchange gives a design whose D and V value are better than those of X^* (i.e, if the design satisfies the aspiration level condition), then the tabu status is overridden. Aspiration function is updated whenever such a dominant design is found.

We stop a try as soon as NEMAX iterations are performed without getting another dominant design since the last iteration in which a dominant design is found, and take X^* as a candidate for the (near) Pareto-efficient design. For finding an appropriate NEMAX, we tried several values of NEMAX for our test problems, and chose $4n$ as a reasonable compromise between solution quality and computing time.

The NPE-set is updated in a similar way as described in Section 3.1. The total number of tries for a given problem depends on how much computing cost the decision maker can tolerate. As the number of tries increases the NPE-set would approach the true set of Pareto-efficient designs.

4.3. Examples

As an illustrative example, we consider a discrete optimal design problem for a quadratic model with $p = 4$. Factor x_j ($j=1$, 2, 3, 4) is assumed to take three values, namely, -1, 0, and 1. Hence, the total number of candidate points $N= 34 = 81$. Suppose that the number of design points $n= 23$. Then, with respect to D and V criteria, we want to determine Pareto-efficient designs, each of which consists of 23 points out of 81 candidate points.

A (near) optimal design for each objective is obtained by 10 tries of TS. The D and V values of the two (near) optimal designs are (0.3306×10^{16}, 0.70373) and (0.2820×10^{16}, 0.67302). The initial NPE-set is composed of these two designs.

For the first try, the D and V values corresponding to the random starting design was (0.1051×10^{8}, 20.84063). Fig. 1. illustrates how the D and V values of dominant designs are changed as iterations proceed. A best dominant value (0.3140×10^{16}, 0.66972) was obtained at iteration 69. The procedure was stopped at iteration 161 after $4n$ ($= 92$) iterations of no improvement since iteration

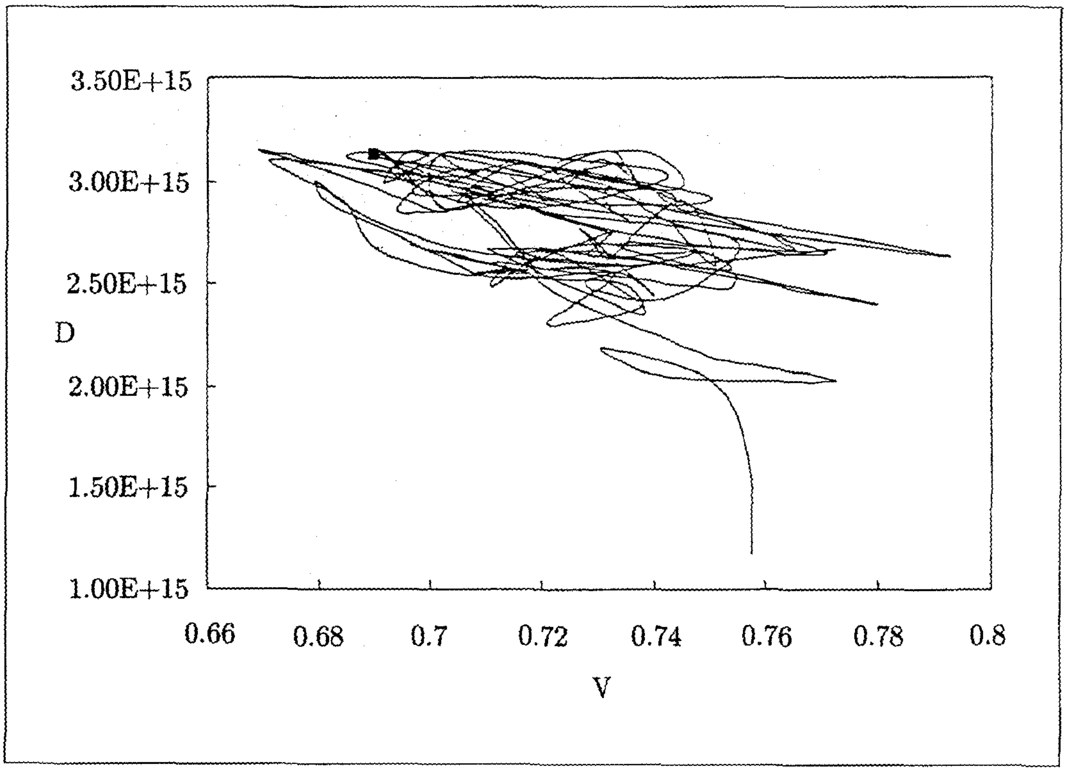

Fig. 1 Changes in the determinant and the average variance in a certain try of TSDH for the first example. '▪' represents the termination point.

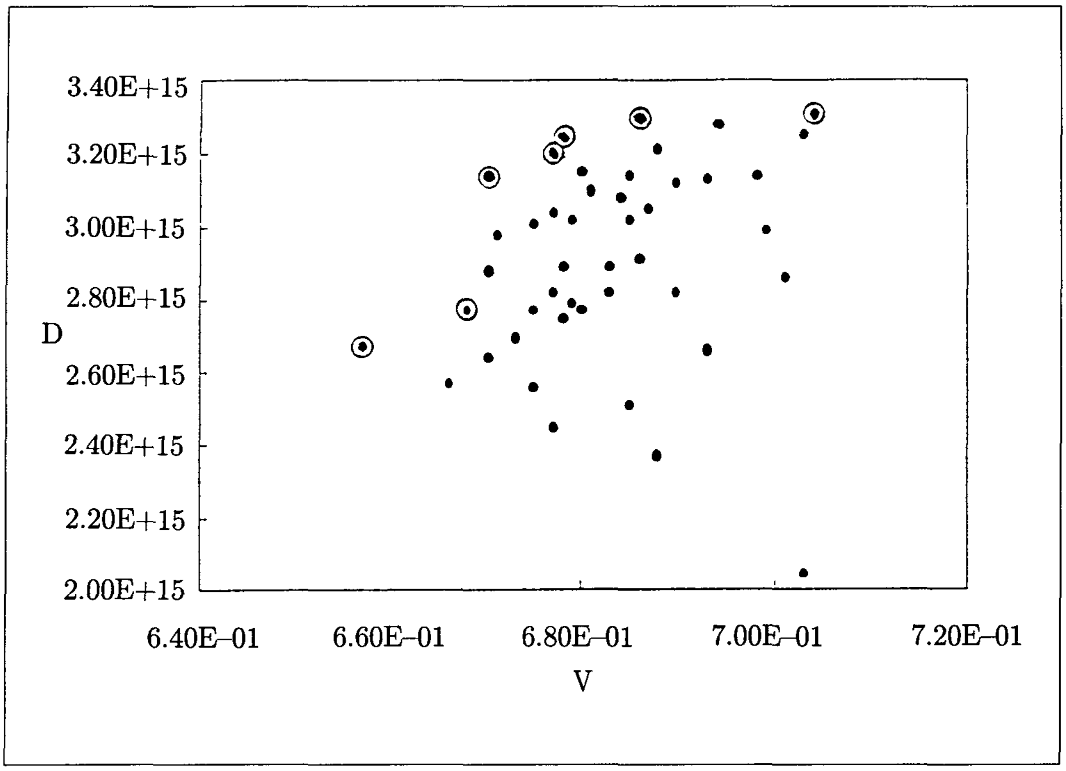

Fig.2 Near Pareto-efficient designs generated by 100 tries of TSDH for the first example. '◉' represents the final (near) Pareto-efficient designs.

69. Fig. 1 also shows that TSDH often accepts a design which is dominated by X^*. After 100 tries, the final 7 (near) Pareto-efficient designs were obtained as shown in Fig. 2. Average solution time per try without input/output time on a Vax 6430 was 6.7 seconds for this example.

For obtaining the NPE-set which is close to the true set of Pareto-efficient designs, we generated another 900 candidate Pareto-efficient designs by TSDH. The result showed that only one of the previous 7 (near) Pareto-efficient designs was dominated by one of 900 candidate Pareto-efficient designs, and only 3 new (near) Pareto-efficient designs were found.

We carried out the same experimentation for the other 10 problems of quadratic models with $p = 4$ or 5 (see Table 1), and observed similar patterns as for the first example except the case where $(p, n) = (5, 29)$. Average solution time per try without input/output time was also reasonable. That is, those were 6 $\sim$ 13 seconds for the examples with $p = 4$ and 26 $\sim$ 70 seconds for the examples with $p = 5$.

Table 1. Performances of TSDH for quadratic models ($p = 4, 5$)

(p, n)	PE1	PE2	ND	T
(4, 17)	4	4	0	3.82
(4, 19)	1	1	0	7.48
(4, 21)	6	8	0	6.16
(4, 25)	7	10	2	12.12
(4, 27)	8	10	3	13.48
(5, 23)	4	5	1	34.01
(5, 25)	5	8	1	45.22
(5, 27)	4	9	1	40.67
(5, 29)	9	16	6	44.37
(5, 31)	5	7	0	69.87

PE1: number of (near) Pareto-efficient designs out of 100 candidate Pareto-efficient designs
PE2: number of (near) Pareto-efficient designs out of 1000 candidate Pareto-efficient designs
ND: number of PE1 which are dominated by another 900 candidate Pareto-efficient designs
T: average solution time in seconds per try without input/output time

5. CONCLUSION

The TS-based heuristic, originally devised for a single-objective combinatorial optimization problem, is extended to determine (near) Pareto-efficient solutions to a multi-objective combinatorial problem. The developed algorithm is then applied to the discrete optimal design problem in statistics.

Since no systematic approach has been reported for solving multi-objective discrete optimal design problem, the relative performance of the proposed algorithm is not assessed in the present investigation.

It is recommended that future research be directed towards improving the present algorithm by considering such additional features of TS as dynamic tabu list, multiple tabu lists, multiple aspiration level conditions, etc. Comparisons of the proposed algorithm with an extended version of simulated annealing or genetic algorithm would be another fruitful area of future research.

REFERENCES

[1] Atkinson, A. C. and Donev, A. N., Optimum Experimental Design, Oxford University, Press, New York, 1992.

[2] Chankong, V. and Haimes, Y. Y., Multiobjective Decision Making: Theory and Methodology Series, 8, North-Holland, New York, 1983.

[3] Glover, F., "Future paths for integer programming and links to artificial intelligence" , *Computers and Operations Research.*, Vol. 13, (1986), pp. 533-549.

[4] Glover, F., "Tabu search-Part I" , *ORSA Journal on Computing,* Vol. 1, (1989), pp. 190-206.

[5] Glover, F., "Tabu search-Part II" , *ORSA Journal on Computing,* Vol. 1, (1990), pp. 4-32.

[6] Glover, F., Laguna, M., Taillard, E. and de Werra, D., "Tabu search" , Annals of Operations Research, Vol. 41, 1993.

[7] Jung, J. S. and Yum, B. J., "Construction of exact D-optimal designs by tabu search" , *Computational Statistics & Data Analysis,* Vol. 21, (1996), pp. 181-191.

[8] Mitchell, T. J., "An algorithm for the construction of D-optimal experimental designs" , *Technometrics,* Vol. 16, (1974), pp. 203-210.

[9] Pukelsheim, F., Optimal Design of Experiments, Wiley, New York, 1993.

[10] Ringuest, J. L., Multiobjective Optimization: Behavioral and Computational Considerations, Kluwer Academic Publishers, 1992.

[11] de Werra, D. and Hertz, A., "Tabu search techniques: A tutorial and an application to neural networks" , Working paper (Department de Mathematiques, Ecole Polytechnique Federale de Lausanne, Lausanne, 1989).