Available online at www.sciencedirect.com
SCIENCE DIRECT®
European Journal of Operational Research 169 (2006) 918–931
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

Multi-objective tabu search using a multinomial probability mass function

Sadan Kulturel-Konak [a], Alice E. Smith [b,*], Bryan A. Norman [c]

[a] *Management Information Systems, Penn State Berks-Lehigh Valley College, Reading, PA 19610, USA*
[b] *Department of Industrial and Systems Engineering, 207 Dunstan Hall, Auburn University, Auburn, AL 36849, USA*
[c] *Department of Industrial Engineering, University of Pittsburgh, Pittsburgh, PA 15261, USA*

Accepted 13 August 2004
Available online 10 December 2004

Abstract

A tabu search approach to solve multi-objective combinatorial optimization problems is developed in this paper. This procedure selects an objective to become active for a given iteration with a multinomial probability mass function. The selection step eliminates two major problems of simple multi-objective methods, a priori weighting and scaling of objectives. Comparison of results on an NP-hard combinatorial problem with a previously published multi-objective tabu search approach and with a deterministic version of this approach shows that the multinomial approach is effective, tractable and flexible.

Keywords: Multi-objective combinatorial optimization; Tabu search

1. Introduction

Most real-life problems naturally comprise more than one objective to be optimized; the task of finding one or more optimum solutions for these problems is known as multi-objective optimization. In its general form, according to Deb (2001), a multi-objective problem can be stated as follows:

* Corresponding author: Tel.: +1 334 844 1400; fax: +1 334 844 1381.
E-mail addresses: sadan@psu.edu (S. Kulturel-Konak), aesmith@eng.auburn.edu (A.E. Smith), banorman@engr.pitt.edu (B.A. Norman).

0377-2217/$ - see front matter
doi:10.1016/j.ejor.2004.08.026

$$\left.\begin{array}{lll} \text{Minimize} & f_b(\mathbf{x}), & b = 1,2,\ldots,B, \\ \text{Subject to} & g_j(\mathbf{x}) \geqslant 0, & j = 1,2,\ldots,J, \\ & h_k(\mathbf{x}) = 0, & k = 1,2,\ldots,K, \\ & x_i^{(L)} \leqslant x_i \leqslant x_i^{(U)}, & i = 1,2,\ldots,n, \end{array}\right\} \quad (1)$$

where $\mathbf{x}$ is a solution vector of n decision variables, $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. There are B objective functions (minimization is considered here but, it can also be maximization) considered in the above formulation. Since a vector of objectives rather than a single objective is optimized, sometimes multi-objective optimization is referred to as vector optimization. The bounds in the last set of constraints define the decision space. In (1), a solution, $\mathbf{x}^{(1)}$, is said to be dominated by another solution, $\mathbf{x}^{(2)}$, if $f(\mathbf{x}^{(2)})$ is at least as small as $f(\mathbf{x}^{(1)})$ (i.e., $f_b(x^{(2)}) \leqslant f_b(x^{(1)})$, $\forall b = 1, \ldots, B$, and $f_b(x^{(2)}) < f_b(x^{(1)})$ for at least one of the objectives, $b \in \{1, \ldots, B\}$). If no $\mathbf{x}^{(2)}$ exists, then $\mathbf{x}^{(1)}$ is said to be non-dominated (or Pareto optimal). A solution is said to be *Pareto optimal* if it is not dominated by another solution in the feasible solution space. The curve formed by joining all Pareto optimal solutions is called the Pareto optimal front. Therefore, the essential goal in multi-objective optimization is to find a set of solutions identical to the Pareto optimal front.

The remainder of this paper will address exploring a well-known metaheuristic technique, tabu search (TS), in the context of multi-objective combinatorial optimization. Section 2 presents a comprehensive literature review of TS and its application to multi-objective optimization. In Section 3, a new multi-objective TS using a multinomial probability function is developed. In Section 4, its effectiveness is demonstrated on two versions of an NP-hard combinatorial multi-objective optimization problem, the series parallel redundancy allocation problem. Concluding remarks are in Section 5.

2. Literature survey

TS is a local (neighborhood) search guided by a selection function which is used to evaluate candidate solutions (Glover, 1989, 1990). To avoid being trapped in local optima, a TS heuristic has two important features: the move operator and the tabu list. A new solution (candidate solution) is produced by the move operator slightly perturbing a current solution. The set of all candidate solutions produced by the move operator is called the neighborhood of the current solution. The best candidate solution is the one with the best objective function value in the neighborhood. To avoid cycling, some aspect(s) of recent moves are classified as forbidden and stored on a tabu list for a certain number of iterations. In canonical TS, in each iteration, the algorithm is forced to select the best move which is not tabu. In some cases, however, if a tabu move improves upon the best solution found so far, then that move can be accepted. This is called an aspiration criterion. After the neighborhood of the current solution is investigated and the best candidate is determined, the tabu list is updated, the best candidate is assigned to the current solution, and the entire process starts again. The search continues until a predetermined stopping condition is satisfied. Detailed material about the canonical and more complex versions of TS can be found in Glover et al. (1993) and Glover and Laguna (1997).

In an early multi-objective combinatorial optimization survey paper, Ulungu and Teghem (1994) suggested the application of metaheuristics, such as simulated annealing (SA), TS and Genetic Algorithms (GA), since they are relatively easy to implement and generate good solutions in comparatively less time than classical optimization methods such as math programming or dynamic programming. A few years later, Ehrgott and Gandibleux (2000) provided a complete survey on multi-objective combinatorial optimization by defining its main characteristics, reviewing available exact and heuristic solution techniques, and identifying potential future work areas. Whilst TS is mentioned in these survey papers, it has rarely been the primary focus of multi-objective optimization approaches. A recent overview of meta-heuristic methods for

multi-objective problems by Jones et al. (2002) pointed out that only about 6% of studies utilized TS while 70%, the majority, employed GA.

A multi-objective TS (MOTS) approach was first applied to cell formation problems in group technology with multiple objectives by Hertz et al. (1994). Their approach was based on solving a sequence of single objective, multi-constraint subproblems. In these subproblems, each objective was considered in turn and was optimized according to its relative importance. Similarly, Alves and Clímaco (2000) proposed an interactive method for 0–1 multi-objective problems using SA and TS. In each iteration, a decision-maker must first specify the regions of major interest. Then, either SA (0–1 SA) or TS (0–1 TS) is run considering only one objective at a time during the entire search. Therefore, this approach does not actually optimize multiple objectives directly.

A MOTS procedure developed by Hansen (1997) is closer to true multi-objective optimization and works with a set of solutions searching for Pareto optimal solutions in parallel. To find the best candidate in the neighborhood of each current solution, the fitness was the weighted average of the objectives. Since the weights were important, they were dynamically updated so that unexplored regions of the Pareto front become attractive. In 2000, the MOTS procedure of Hansen (1997) was applied to the resource constrained project-scheduling problem by Viana and de Dousa (2000) and compared to Pareto simulated annealing (PSA) (originated by Czyzac and Jaszkiewicz (1998)). Results indicated that the MOTS procedure obtained better solutions than PSA. In a subsequent paper, Hansen (2000) published a pseudo code for his earlier MOTS algorithm, called tabu search for multi-objective combinatorial optimization (TAMOCO). The notation and method are given below.

Notation

D set of feasible solutions
$x, y \in D$ solutions of the problem
S set of current solutions
M set of non-dominated (ND) solutions
$N(x)$ neighborhood of solution x (user defined)
TL_i tabu list associated with solution i
$A(x_i, y_i)$ attributes of the solutions stored in TL_i
B number of objectives of the problem
$\Lambda = [\lambda_1, \lambda_2, \ldots, \lambda_B]$ set of weights associated with each objective
$\Pi = [\pi_1, \pi_2, \ldots, \pi_B]$ range equalization factors
DRIFT a fixed number of increments for the count variable

Input parameters

- The size of S and TL_i
- DRIFT

Procedure TAMOCO (Hansen, 2000) (minimization of all k objectives)

Initialization

Select a starting set, S, of random feasible solutions such that $S \subset d$

For each $x_i \in S$, set $\mathrm{TL}_i = \emptyset$
$M = \emptyset$, count $= 0$, $\pi_k = 1/k$ for each objective k
While (the stopping criterion is not satisfied){
For each $x_i \in S$\{

$\Lambda = 0$
For each $x_j \in S$ where $f(x_j)$ is ND by $f(x_i)$ and $f(x_j) \neq f(x_i)$\{
$w = \frac{1}{d(f(x_i), f(x_j), \Pi)}$
where $d(f(x_i), f(x_j), \Pi) = \sum_{k=1,\ldots,B} \pi_k |f_k(x_i) - f_k(x_j)|$
For each objective k where $f_k(x_i) < f_k(x_j)$, $\lambda_k = \lambda_k + \pi_k \times w$\}
If ($\Lambda = 0$)\{
Set random weights such that $\lambda_k \geqslant 0$ for $\forall k$
Normalize Λ such that $\sum_k \lambda_k = 1$\}
Find the solution y_i which minimizes $\Lambda \times f(y_i)$
where $\{y_i \in N(x_i)$ and $(A(x_i, y_i) \notin \mathrm{TL}_i$ or $y_i \in M)\}$
Update set M with $y_i \in N(x_i)$
if (a new ND solution is found), count $= 0$
if (TL_i is full), remove the oldest element from TL_i
Add $A(y_i, x_i)$ to TL_i as the newest element $x_i = y_i$
Update Π such that $\pi_k = \frac{1}{\mathrm{Range}_k} \left[\sum_{i=1}^{B} \frac{1}{\mathrm{Range}_i}\right]^{-1}$
where Range_k is the range width of objective k in M
If (DRIFT criterion is reached)
Set one randomly selected solution from S equal to another randomly selected solution from S
count = count + 1\}\}

In order to encourage Pareto front exploration, a randomly selected solution is replaced by another randomly selected solution whenever the DRIFT criterion is satisfied. This is a diversification strategy. The procedure returns M, the set of non-dominated solutions found.

Another MOTS procedure was proposed by Baykasoglu et al. (1999). In their approach, the best candidate solution was randomly selected from a list of candidate solutions, which consists of feasible solutions non-dominated by the current neighborhood solutions and the current Pareto front. This approach did not have a tool to diversify the search over a wide Pareto front. Later, Baykasoglu (2001a,b) applied this procedure to goal programming and aggregate production planning, respectively. Both Hansen's (2000) and Baykasoglu et al.'s (1999) methods did not address handling infeasibility as they considered only feasible solutions in a neighborhood.

In the multi-objective optimization literature many methods depend on combining the set of objectives into a single objective usually using a weighting scheme (i.e., a linear combination of the objectives). Although this approach is computationally tractable and can be effective, it has several inherent difficulties. First, it is often not known how much importance should be given to each objective (i.e., weighting) in advance of conducting (or during) the optimization procedure. Results can be substantially different with different weights or weighting schemes. Second, Fonseca and Fleming (1995) pointed out that the weighted-sum approach cannot identify all points in a trade-off surface of non-convex solution spaces. This is a major handicap of the weighted-sum approach since many combinatorial problems have non-convex and discontinuous solution spaces. Third, a problem of scaling among objectives can occur. It is likely that each objective takes different orders of magnitudes which can affect the mathematical procedure. Normalization can solve the problem of scaling (Deb, 2001) but requires a priori setting of proper ranges along which to scale.

This paper circumvents these difficulties by using a multinomial probability mass function to select an objective to become active in each iteration. Although the idea is simple, it is very general and easy to implement. It can accommodate two or more objectives, there are no weights to set and no scaling adjustment to

be made. The behavior of the random variable is beneficial (as shown in the results later in the paper) to the search by enacting both short term and longer term consideration of an objective depending on the realization of the random variable.

The idea of alternating objectives in multi-objective optimization has been applied in the area of evolutionary computation where selection and recombination are cornerstones of the metaheuristic. The first multi-objective GA was proposed by Schaffer (1985) and called the Vector Evaluated Genetic Algorithm (VEGA) where each element of the vector represented each objective function. In this approach, the population was divided into equal sub-populations randomly and each sub-population used only a single objective. Solutions from the sub-populations were recombined to generate offspring. In another multi-objective GA, Fourman (1985) selected the new generation by comparing a pair of individuals using a uniformly and randomly selected single objective while ignoring other objectives. As an alternative, the selection probability of each objective could be changed. He tested this new approach on layout design problems and observed that the resulting populations show a greater variability than that found with the traditional weighted objective approach. Similarly, Kursawe (1991) employed a user-defined vector that determines the probability of each objective becoming active in the selection of the next generation. The vector can be either fixed or updated during the search.

3. The multinomial tabu search algorithm

The multinomial tabu search (MTS) algorithm is proposed and the algorithm steps are described below with some notes on the method.

Step 0. Initialization
Initialize the tabu list and the ND solutions list as empty. Select any feasible solution as the current solution and add it to the ND solutions list.

Step 1. Select the objective
Select a single objective to become active from the set of all objectives using a multinomial probability mass function.

Step 2. Search the neighborhood
Search the neighborhood of all possible defined moves. Choose the non-tabu (or if it is tabu, but dominates any solution in the ND solutions list) candidate solution with the best objective function value as the best candidate solution.

Step 3. Update the ND solutions list
Compare each feasible candidate solution with the current ND solutions list as follows. If a candidate solution dominates some current ND solutions, remove these dominated solutions from the ND solutions list and add the candidate to the ND solutions list. If a candidate solution is not dominated by any current ND solution, add it to the ND solutions list. (Computational experience herein did not result in an overly large ND set, however mechanisms such as niching might be employed to reduce the size of the ND set if necessary.)

Step 4. Update the tabu list
Add the accepted move by *Step 2* as the last tabu list entry. If the tabu list is full, the oldest tabu list entry is deleted. (A dynamic length tabu list is used, varying every 20 iterations, however from computational experience the algorithm proposed here is not sensitive to tabu list size.)

Step 5. Diversification
A diversification scheme based on restart is used. If the list of ND solutions has not been updated in the last (stopping criterion/4) moves, one of the ND solutions found during the search is uniformly randomly selected as the new current solution, the tabu list is reset to empty, and the search restarts

from this solution (i.e., return to *Step 1*). (Experiments show that this diversification scheme improves the performance of the MTS algorithm by exploring a wider Pareto front. There are probably other diversification schemes that could be implemented with good effect.)

Step 6. Check the stopping criterion

Check the stopping criterion, which is defined as the maximum number of iterations conducted without updating the ND solutions list, and if it is not satisfied, return to *Step 1*.

To accommodate constraints, a penalty function is used for infeasible solutions. The fitness function according to the minimization of the kth objective, B_k, is given by

$$F_i = B_k(i) + [B_k(\text{ND\_worst}) - B_k(\text{all})] \times \left[\frac{\max(0, \text{constraint violation})}{\text{NFT}}\right]^{\kappa}. \quad (2)$$

In (2), B_k(ND_worst) is the worst (kth objective) value in the ND solutions list, and B_k (all) is the best (kth objective) value found so far. The last part of the fitness function corresponds to a penalty for infeasibility using the notion of Near Feasibility Threshold (NFT), first defined by Smith and Tate (1993) and then improved by Coit et al. (1996). NFT is the threshold distance from the feasible region where search should be encouraged. Within the NFT region, infeasible solutions are penalized relatively lightly while beyond the NFT region, a heavy penalty is applied. NFT is updated according to the feasibility ratio of the current move and the recent moves kept in the tabu list. In particular, if most of the recent moves including the current move have been feasible, NFT is increased, thereby decreasing the penalty and encouraging more exploration of the near-feasible region. If most of the recent moves as well as the current move have been infeasible, NFT is decreased, thereby increasing the penalty and moving the search towards the feasible region. Therefore, the NFT is updated without user intervention during the search. More detailed explanation of how to determine a value of NFT is given in Kulturel-Konak et al. (2004). The penalty amplification exponent κ is set to 2 (i.e., quadratic in constraint violation); however, the MTS approach is robust to its assigned value.

4. Demonstrative applications—the redundancy allocation problem

The MTS approach is applied with success to different versions of a hard combinatorial optimization problem: the series parallel system redundancy allocation problem (RAP). In this paper, the series-parallel system RAP, illustrated in Fig. 1, is as follows: determine the optimal design configuration for given objectives and constraints when there are multiple component choices available for each of several k-out-of-n:G subsystems. Whilst the RAP has been studied as a single objective problem in the literature (see Kulturel-Konak et al. (2003) for a complete set of citations) to maintain computational tractability,

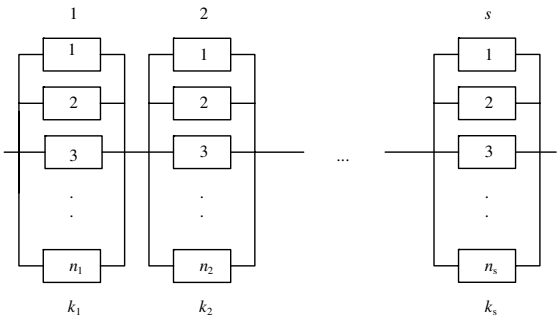

Fig. 1. Series-parallel system configuration.

the design problem is actually more realistically a multi-objective problem. In this paper, two versions of the problem are studied. The first one has two objectives along with a system constraint whilst the second one has three objectives. All algorithms are coded in C and run using an Intel Pentium IV with 2.2 GHz processor and 1 GB RAM.

Notation

R, C, W	reliability, cost, and weight of the system, respectively
s	number of subsystems
$\mathbf{x}_i$	$(x_{i1}, x_{i2}, \ldots, x_{i,m_i})$
x_{ij}	quantity of jth component in subsystem i
n_i	total number of components used in subsystem i (i.e., $\sum_{j=1}^{m_i} x_{ij}$)
$n_{\max,i}$	user assigned maximum number of components in parallel used in subsystem i
k_i	minimum number of components in parallel required for subsystem i
$R_i(\mathbf{x}_i/k_i)$	reliability of subsystem i, given k_i
$C_i(\mathbf{x}_i)$	total cost of system i
$W_i(\mathbf{x}_i)$	total weight of system i

4.1. RAP with two objectives

The bicriteria RAP considered is to determine the optimal design configuration to maximize system reliability (R) and to minimize system cost (C) when there is a maximum limit on system weight (W) and there are multiple component choices available for each of several k-out-of-n:G subsystems. This problem can be represented mathematically as:

$$\left.\begin{aligned} &\max\left(R=\prod_{i=1}^{s}R_i(x_i|k_i)\right),\ \min\left(C=\sum_{i=1}^{s}C_i(x_i)\right)\\ &\text{subject to}\\ &\sum_{i=1}^{s}W_i(x_i)<W,\\ &k_i\leqslant\sum_{j=1}^{m_i}x_{ij}\leqslant n_{\max,i}\quad\forall i=1,2,\ldots,s. \end{aligned}\right\} \tag{3}$$

The problem is to maximize overall system reliability and minimize overall system cost since developing reliable and cost-effective systems is imperative. Whilst designing systems with redundancies every unit added to the system will increase reliability but will also increase cost and weight.

4.1.1. The MTS algorithm for the RAP with two objectives

The test problem considered here uses the input parameters of the problem originally proposed by Fyffe et al. (1968). Hence, the component cost, weight, and reliability values are not reproduced here. The problem encoding is a permutation encoding of size $\sum_{i=1}^{s} n_{\max,i}$ representing a concatenation of the components in each subsystem including non-used components (i.e., defined as "blanks" when $n_i < n_{\max,i}$).

Initialization: To obtain an initial feasible solution, for each subsystem, s integers between k_i and $n_{\max,i} - 3$ (inclusive) are chosen according to a discrete uniform distribution to represent the number of components in parallel (n_i). Then, the n_i, components are randomly and uniformly assigned to the m_i different types. If feasible, it becomes the initial solution.

Objectives: Randomly choose maximizing R or minimizing C as an objective to actively evaluate candidate solutions using equal probabilities of $p_R = p_C = 0.50$ (i.e., $p_R + p_C = 1$).

Moves: Moves operate on each subsystem. The first type of move changes the number of a particular component type by adding or subtracting one. These moves are considered individually for all available component types within all subsystems. The second type of move simultaneously adds one component to a certain subsystem and deletes another component, of a different type, within the same subsystem. The two types of moves are performed independently on the current solution.

Tabu list: The structure of the subsystem that has been changed in the accepted move is stored in the tabu list. The dynamic tabu list length is changed according to a uniform integer random number between $[s, 3s]$.

Stopping criterion: The stopping criterion is defined as the maximum number of iterations conducted without updating the ND solutions list and is set to 1000.

The TAMOCO algorithm of Hansen (2000), which was defined in Section 2, is used for comparison. To equally compare the MTS algorithm with TAMOCO, all general TS parameters, such as encoding, stopping criterion, tabu list structure and length, were set to the same values in both algorithms. For TAMOCO, the set of current solutions, S, is set to 10, and the DRIFT criterion is set after 250 iterations for each current solution, similar to the diversification of *Step 5*, above.

Considering the weight infeasibility, the fitness function according to R is given by

$$F_i = R(i) - [R(\text{all}) - R(\text{ND\_worst})] \times \left[\frac{\max(0, W(i) - W)}{\text{NFT}}\right]^2 \tag{4}$$

and the fitness function according to C is given by

$$F_i = C(i) = [C(\text{ND\_worst}) - C(\text{all})] \times \left[\frac{\max(0, W(i) - W)}{\text{NFT}}\right]^2. \tag{5}$$

In (4), $R(i)$ and $W(i)$ are the system reliability and the system weight of solution i. Similarly, in (5), $C(i)$ is the system cost of solution i. The last part of each fitness function corresponds to a penalty function for weight infeasibility. The definitions of the terms in the multiplier of each penalty function are as follows:

$R_{\text{ND\_worst}}$: The worst R in the ND solutions list.
R_{all}: The best R found so far.
$R_{\text{ND\_worst}}$: The worst C in the ND solutions list.
C_{all}: The best C found so far.

This penalty approach is used for both TAMOCO and MTS.

4.1.2. Computational results

The system weight constraint varied across problems but all levels of the constraint returned similar relative results. Fig. 2 shows the Pareto fronts of TAMOCO vs. MTS for a single replication of each algorithm using $W = 190$. The number of ND solutions of TAMOCO is much less than that of MTS, and all solutions found in the former are dominated by solutions found in both settings of the latter. Two different ways of diversifying the search were tried. The search was restarted with either (i) a randomly selected ND solution (i.e., random), (ii) the furthest ND solution, in Euclidean distance, from the current solution in terms of both objectives (i.e., furthest). Both diversification methods find ND solutions spread over a wide Pareto front. Although the furthest method finds the Pareto front much faster, it is not as good in objective function quality as the Pareto front found by the random approach. Additionally, the Euclidean distance diversification requires more computation as distances must be calculated and sorted.

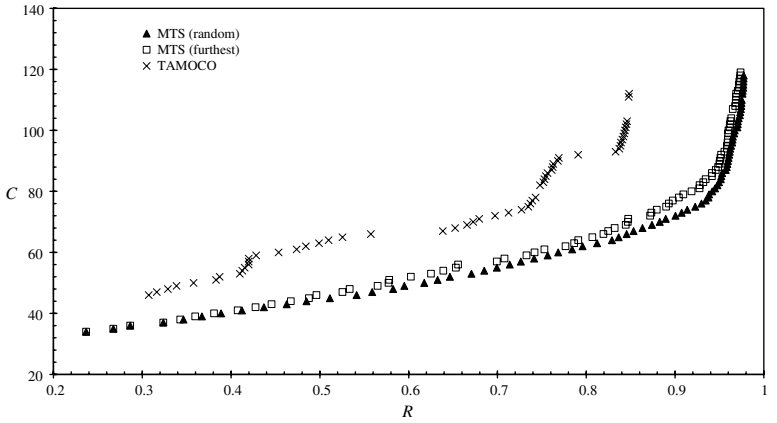

Fig. 2. Pareto fronts of MTS vs. TAMOCO for two objectives.

4.1.3. The impact of altering $\boldsymbol{p}_R$ and $\boldsymbol{p}_C$ on the algorithm performance

Instead of using $p_R = p_C = 0.50$, we analyzed altering the probabilities of selecting R or C as the active objective. A summary of the results is in Table 1. Comparisons of the Pareto fronts can be seen in Figs. 3 and 4 (p_R values of 0.75 and 0.25 are compared with a p_R value of 0.50, respectively). When higher probability is given to maximizing R more solutions with higher reliabilities are found. More solutions are found in the $p_R = 0.75$ case but nine of them are dominated by the $p_R = 0.50$ case whereas only one solution of the $p_R = 0.50$ case is dominated. On the other hand, when a higher probability is given to minimizing C, the algorithm cannot find a diverse set of solutions. Eleven solutions of the Pareto front using $p_R = 0.25$ are dominated by the $p_R = 0.50$ case whereas no solution of the $p_R = 0.50$ case is dominated. One reason for this is that the weight constraint prevents solutions with high reliabilities, but does not prevent those with low costs since minimizing the system cost and the system weight are consistent. Therefore, the area around the right tail of the Pareto optimal front is not thoroughly investigated. However, when viewing Figs. 3 and 4, it can be seen that even with these dramatic changes in probabilities, the Pareto fronts do not differ very much, and they still contain a wide range of excellent solutions.

4.1.4. Multinomial vs. deterministic selection

To test the benefit of a stochastic approach, alternating objectives deterministically, such as R, C, R, C and so on, was tested. As seen in Fig. 5, the algorithm is not able to find a Pareto front as wide as in the

Table 1
The summary of the results of the RAP with two objectives

	TAMOCO	MTS					
		Furthest	Random				
						Deterministically Stopping criterion	
		$p_R = 0.50$	$p_R = 0.50$	$p_R = 0.75$	$p_R = 0.25$	1000	10,000
Number of moves	18,240	7087	43,232	61,019	36,095	28,683	673,162
Number of diversifications	–	22	166	237	123	112	268
Number of DRIFT	70	–	–	–	–	–	–
CPU time (seconds)	3.97	4.57	30.27	49.18	13.58	13.55	312.95

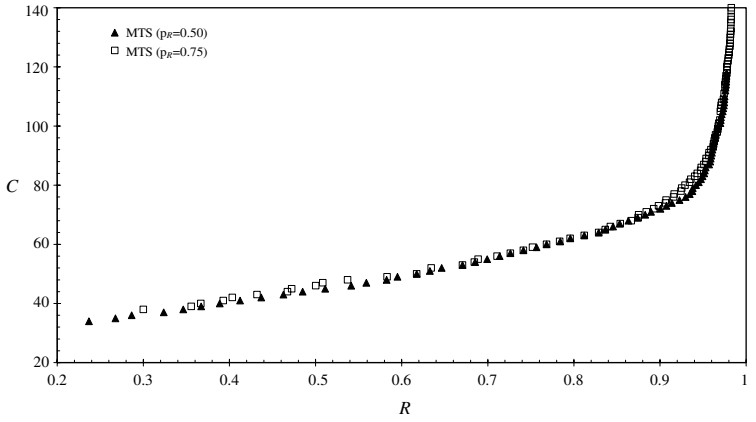

Fig. 3. Pareto fronts of MTS with $p_R = 0.50$ and 0.75.

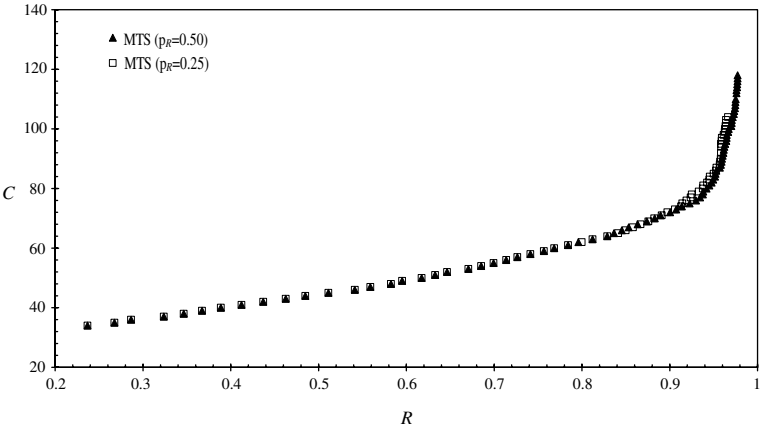

Fig. 4. Pareto fronts of MTS with $p_R = 0.50$ and 0.25.

stochastic case. The reason for this is that the probability of TS cycling increases. Therefore, to investigate the ability of finding a better Pareto front when the two objectives are alternated, the stopping criterion is changed from 1000 to 10,000 non-improving iterations of the ND solutions list. This does improve performance, albeit at significant additional computational cost (Fig. 6). Only six of the 75 ND solutions are dominated by the Pareto front of the $p_R = 0.50$ case whereas, in Fig. 5, almost half of the Pareto front-solutions (i.e., 33 of the 67), are dominated.

4.2. RAP with three objectives

Using the strategy of random diversification and equal probabilities for the multinomial random variable, three objectives were investigated. The problem considered was to determine the optimal design configuration to maximize system reliability (R) and to minimize system cost (C) and weight (W) when there are multiple component choices available for each of several k-out-of-n:G subsystems. The mathematical formulation of the problem is given below.

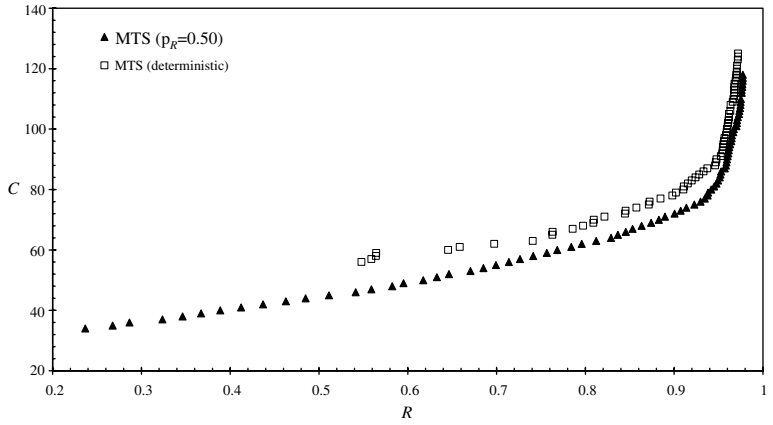

Fig. 5. Pareto fronts of MTS with $p_R = 0.50$ and deterministic.

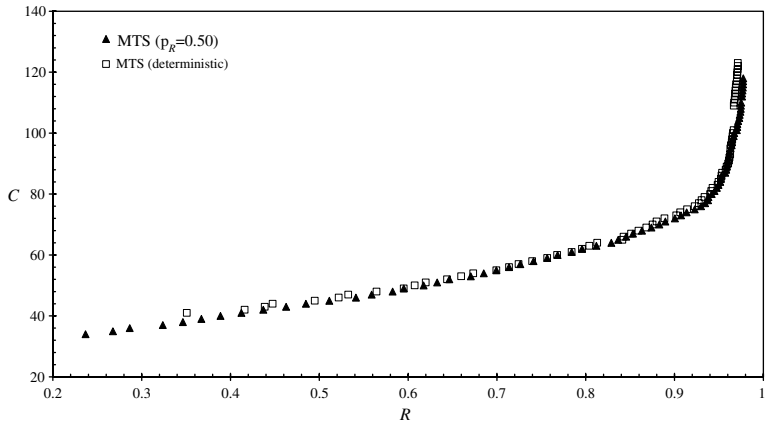

Fig. 6. Pareto fronts of MTS with $p_R = 0.50$ and deterministic in a longer trial.

$$\left.\begin{array}{l} \max\left(R=\prod_{i=1}^{s} R_i(x_i|k_i)\right),\ \min\left(C=\sum_{i=1}^{s} c_i(x_i)\right),\ \min\left(W=\sum_{i=1}^{s} W_i(x_i)\right) \\ \text{subject to} \\ \quad k_1 \leqslant \sum_{j=1}^{m_i} x_{ij} \leqslant n_{\max,i} \quad \forall i = 1,2,\ldots,s. \end{array}\right\} \tag{6}$$

4.2.1. The MTS algorithm for the RAP with three objectives

The MTS algorithm described in Section 3 and the components explained in Section 4.1.1 are used except that the objectives are altered.

Objectives: Randomly choose maximizing R, minimizing C, or minimizing W as an objective to actively evaluate candidate solutions using $p_R = p_C = p_W = 0.3333$ (i.e., $p_R + p_C + p_W = 1$).

4.2.2. Computational results

Fig. 7 shows the Pareto fronts of TAMOCO vs. MTS for a single replication of each algorithm. The figure shows the three dimensional results along with two dimensional projections on each set of pairwise objectives. Unlike the very narrow Pareto front of the TAMOCO, the MTS algorithm finds ND solutions spread over a wide range. Table 2 summarizes the results. The number of Pareto optimal solutions of TAMOCO is much less than that of MTS and all solutions found by the former are dominated by the solutions found by the latter. In the TAMOCO algorithm, the set of current solutions, S, was set to 10. However, since the number of moves and CPU seconds are higher in the MTS algorithm, a larger S, 50, was tried. This increase did not improve the Pareto front identified by TAMOCO.

5. Conclusions

TS has previously been demonstrated to be a successful optimization technique in many diverse single objective problems. However, its ability to solve multi-objective combinatorial optimization problems

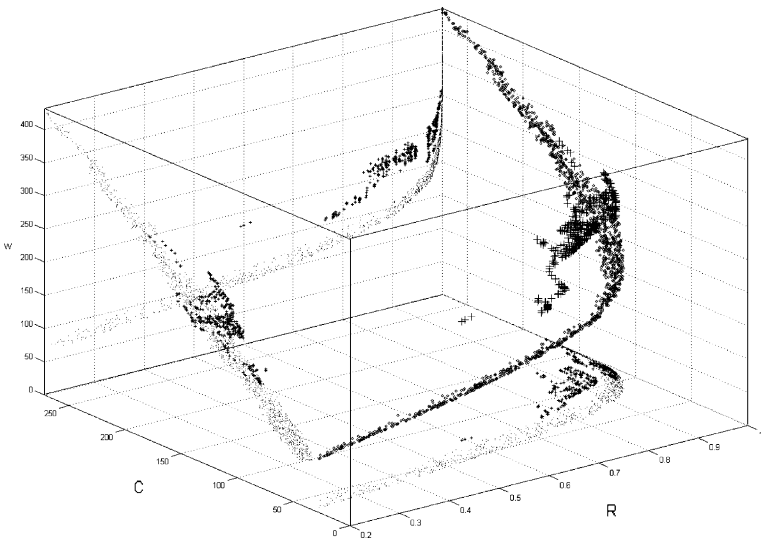

Fig. 7. Pareto fronts of MTS (small and light markers) vs. TAMOCO (large and dark x's) for three objectives (note that the projections along each planar axis are also shown on the appropriate planes).

Table 2
The summary of the results of the RAP with three objectives

	TAMOCO		MTS
	$S = 10$	$S = 50$	
Number of moves	143,810	557,100	424,576
Number of diversifications	–	–	1454
Number of DRIFT	550	1000	–
Number of Pareto optimal solutions found	1134	1059	5230
Number of solutions dominated by the other method	1134	1059	0
CPU time (seconds)	151.34	258.59	354.00

has not been widely studied. The main motivation of the approach developed in this paper is to remedy some general obstacles of the classical methods of multi-objective optimization (i.e., weighting and scaling of each objective), while maintaining computational ease. The proposed MTS algorithm randomly activates an objective using a multinomial probability mass function to select the best candidate solution in each move. Some alterations, including changing p_R and using a distance based diversification scheme, were examined. It was found that the simplest version of MTS (i.e., equal p_R values and random diversification) worked the best. MTS compared very favorably both to its deterministic variant and to a competing method from the literature, TAMOCO. MTS is simple, general and tractable. Results show that it is quite effective at identifying a wide Pareto front. Interesting future work includes extending the multinomial idea to problems in continuous optimization.

References

Alves, M.J., Clímaco, J., 2000. An interactive method for 0-1 multiobjective problems using simulated annealing and tabu search. Journal of Heuristics 6, 385–403.

Baykasoglu, A., 2001a. Goal programming using multiple-objective tabu search. Journal of the Operational Research Society 52 (12), 1359–1369.

Baykasoglu, A., 2001b. MOAPPS 1.0: Aggregate production planning using the multiple-objective tabu search. International Journal of Production Research 39 (16), 3685–3702.

Baykasoglu, A., Owen, S., Gindy, N., 1999. A taboo search based approach to find the Pareto optimal set in multiple objective optimisation. Journal of Engineering Optimization 31, 731–748.

Coit, D.W., Smith, A.E., Tate, D.M., 1996. Adaptive penalty methods for genetic optimization of constrained combinatorial problems. INFORMS Journal on Computing 8, 173–182.

Czyzac, P., Jaszkiewicz, A., 1998. Pareto simulated annealing—a metaheuristic technique for multiple objective combinatorial optimization. Journal of Multicriteria Decision Analysis 7, 34–47.

Deb, K., 2001. Multiobjective Optimization Using Evolutionary Algorithms. John Wiley & Sons, New York.

Ehrgott, M., Gandibleux, X., 2000. A survey and annotated bibliography of multiobjective combinatorial optimization. OR Spektrum 22, 425–460.

Fonseca, C.M., Fleming, P.J., 1995. An overview of evolutionary algorithms in multiobjective optimization. Evolutionary Computation 3 (1), 1–16.

Fourman, M.P., 1985. Compaction of symbolic layout using genetic algorithms. In: Proceedings of the First International Conference on Genetic Algorithms. pp. 141–153.

Fyffe, D.E., Hines, W.W., Lee, N.K., 1968. System reliability allocation and a computational algorithm. IEEE Transactions on Reliability 17, 74–79.

Glover, F., 1989. Tabu search—Part I. ORSA Journal on Computing 1, 190–206.

Glover, F., 1990. Tabu search—Part II. ORSA Journal on Computing 2, 4–32.

Glover, F., Laguna, M., 1997. Tabu Search. Kluwer Academic Publishers, Boston, MA.

Glover, F., Taillard, E., de Werra, D., 1993. A user's guide to tabu search. Annals of Operations Research 41, 3–28.

Hansen, M.P., 1997. Tabu search for multiobjective optimization: MOTS. Paper Presented at the 13th International Conference on Multi Criteria Decision Making (MCDM'97). Cape Town, South Africa.

Hansen, M.P., 2000. Tabu search for multiobjective combinatorial optimization: TAMOCO. Control and Cybernetics 29 (3), 799–818.

Hertz, A., Jaumard, B., Ibeiro, C.C., Formosinho Filho, W.P., 1994. A multi-criteria tabu search approach to cell formation problems in group technology with multiple objectives. Recherche Operationelle/Operations Research 28 (3), 303–328.

Jones, D.F., Mirrazavi, S.K., Tamiz, M., 2002. Multi-objective meta heuristics: An overview of the current state-of-the-art. European Journal of Operational Research 137, 1–9.

Kulturel-Konak, S., Coit, D.W., Smith, A.E., 2003. Efficiently solving the redundancy allocation problem using tabu search. IIE Transactions 35 (6), 515–526.

Kulturel-Konak, S., Norman, B.A., Coit, D.W., Smith, A.E., 2004. Exploiting tabu search memory in constrained problems. INFORMS Journal on Computing 16 (3), 241–254.

Kursawe, F., 1991. A variant of evolution strategies for vector optimization. In: Schwefel, H.-P., Manner, R. (Eds.), Parallel Problem Solving from Nature. Springer-Verlag, Berlin, pp. 193–197.

Schaffer, J.D., 1985. Multiple objective optimization with vector evaluated genetic algorithms. In: Proceedings of the First International Conference on Genetic Algorithms. pp. 93–100.

Smith, A.E., Tate, D.M., 1993. Genetic Optimization using a Penalty Function. In: Proceedings of the Fifth International Conference on Genetic Algorithms. pp. 499–505.

Ulungu, E.L., Teghem, J., 1994. Multi-objective combinatorial optimization problems: A survey. Journal of Multi-Criteria Decision Analysis 3, 83–104.

Viana, A., de Dousa, J.P., 2000. Using metaheuristics in multiobjective resource constrained project scheduling. European Journal of Operational Research 120, 359–374.