

Multiple Weighted Objectives Heuristic for the Redundancy Allocation Problem

David W. Coit and Abdullah Konak

Abstract—A new heuristic is proposed and tested for system reliability optimization. The multiple weighted objective heuristic is based on a transformation of the problem into a multiple objective optimization problem, and then ultimately, transformation into a different single objective problem. The multiple objectives are to simultaneously maximize the reliability of each individual subsystem. This is a logical approach because system reliability is the product of the subsystem reliabilities, so if they are maximized, the system reliability will also be high. This new formulation and associated heuristic are then based on solving a sequence of linear programming problems. It is one of the very few optimization approaches that allow for linear programming algorithms and software to be used for the redundancy allocation problem when mixing of functionally equivalent components is allowed. Thus, it represents an efficient solution method that relies on readily available optimization tools. The heuristic is tested on many example problems, and compared to competing solution approaches. Overall, the heuristic performance is observed to be very good on the tested problem, and superior to the max-min heuristic regarding both efficiency, and performance.

Index Terms—Multiple objective optimization, redundancy allocation, system reliability.

Acronyms¹

GA	genetic algorithm
MOGA	multiple objective genetic algorithm
MWO	multiple weighted objectives
TS	tabu search

Notation

C, W	system cost, weight constraint limits
s	number of subsystems
m_i	number of available component choices for subsystem i
r_{ij}, c_{ij}, w_{ij}	[reliability, cost, weight] of component j available for subsystem i
\mathbf{x}	$(x_{11}, x_{12}, \dots, x_{1m_1}, x_{21}, x_{22}, \dots, x_{2m_2}, \dots, x_{s1}, x_{s2}, \dots, x_{s,m_s})$

Manuscript received October 7, 2004; revised October 26, 2005. Associate Editor: M. Zuo.

D. W. Coit is with the Department of Industrial & Systems Engineering, Rutgers University, Piscataway, NJ 08854 USA (e-mail: coit@rutgers.edu).

A. Konak is with the Department of Information Sciences and Technology, Penn State Berks-Lehigh Valley, Reading, PA 19610 USA (e-mail: konak@psu.edu).

Digital Object Identifier 10.1109/TR.2006.879654

¹The singular and plural of an acronym are always spelled the same.

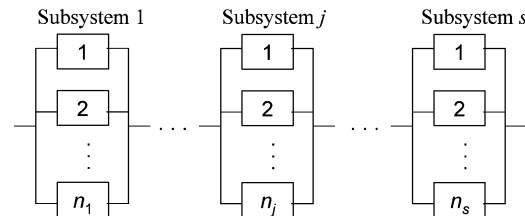


Fig. 1. Example series-parallel system.

\mathbf{x}_i	$(x_{i1}, x_{i2}, \dots, x_{i,m_i})$
x_{ij}	number of components of type j used in subsystem i
$R(\mathbf{x})$	system reliability
$R_i(\mathbf{x}_i)$	reliability of subsystem i
ω_i	objective weight assigned to the i^{th} subsystem
R_i^{\min}	minimum subsystem reliability for subsystem i

I. INTRODUCTION

THE objective of the redundancy allocation problem is to determine an optimal system design to maximize system reliability given constraints on the system. It is a difficult non-linear integer programming problem that has been extensively studied because it is widely applicable and relevant, but also because it is challenging to solve. A new multiple weighted objectives (MWO) heuristic has been developed by transforming the problem into one with the simultaneous objectives of maximizing each of the subsystem reliabilities for a series-parallel system. The problem can then be further transformed to allow for the use of standard linear programming algorithms in a sequence of updated problems. This new heuristic is attractive because it is easy to apply, it easily accommodates problems with a mix of components, and the observed performance is very good.

Fig. 1 depicts an example series-parallel system. For each subsystem, there are multiple, functionally equivalent components available to be used in the system. The design can include a single component selection for each subsystem, or there may be multiple components selected & arranged in parallel. For redundancy allocation, the decision variables are the component choices (from the available discrete choices), and the redundancy levels. Kuo *et al.* [1] provide a comprehensive review on system reliability optimization. For series-parallel systems with a single objective, Chern [2] demonstrated that the redundancy allocation problem is NP-hard.

The redundancy allocation problem has been solved using various mathematical programming, and other optimization approaches. Fyffe *et al.* [3] used dynamic programming to solve

the redundancy allocation problem by limiting the problem to only one type of component available for each subsystem. Nakagawa & Miyazaki [4] demonstrated that a surrogate constraints approach is efficient to accommodate multiple constraints with dynamic programming. Integer programming [5], genetic algorithms (GA) [6], and tabu search [7] have also been applied to determine the optimal design configuration. Other approaches [8] include the solution space reduction procedure, and branch-and-bound to find the optimal solution with multiple component choices. Mathematical programming approaches to the problem have generally restricted the solution space by (1) only allowing one component choice for each subsystem, or (2) allowing multiple component choices, but allowing no mixing within a subsystem once a component has been selected.

Because the problem has proven to be difficult, many heuristic approaches have also been proposed to yield efficient solutions that are intended to lead to good, if not optimal, solutions. Kuo *et al.* [1] also provides a summary of these heuristic approaches. Nakagawa & Nakashima [9] developed a very useful heuristic for problems with multiple component choices. Their heuristic is based on incrementally adding new components to the system design based on a sensitivity factor. The components are added until the system constraints are violated. Kuo *et al.* [10] later extended this research by modifying the sensitivity factor with impressive results. Li [11] also presents an interesting heuristic based on a bounded region.

Hsieh [12] presented a linear approximation of system reliability which resulted in efficient solutions to the problem. This was the first model that allowed standard linear programming algorithms to be applied to the redundancy allocation problem when a mixing of components was considered. Ramirez *et al.* [13] presented the max-min heuristic for this problem. This heuristic works by transforming the problem to maximize the minimum subsystem reliability. This created a surrogate problem that was easier to solve, but still yielded very good solutions to the original problem. Lee *et al.* [14] performed numerous problem simulations, and compared the max-min heuristic with the Nakagawa & Nakashima heuristic. They concluded that the max-min heuristic generally out-performed the Nakagawa & Nakashima heuristic in obtaining higher system reliabilities; however, it was also more computationally extensive. The MWO heuristic is based on a similar approach to the max-min heuristic, but generally outperforms it with respect to efficiency (CPU time), and performance (objective function).

Similar to the max-min heuristic, in this new formulation, it becomes possible to efficiently consider the case of mixing functionally equivalent components within a single subsystem. This is a primary benefit of this formulation because most previous mathematical programming formulations of the problem generally forbid mixing, either to linearize the problem, or to reduce the solution space. Thus, previous integer programming approaches identified an "optimal" solution to a restricted sub-set of the actual solution space. Of course, some system design problems do require that an identical component type be used to provide redundancy; but for many others, this is not the case. For example, a mechanical gyroscope is used in many airplane designs to provide redundancy for a primary electronic gyroscope.

Coit & Smith [6] demonstrated that consideration of component mixing increased the problem solution space, thereby resulting in higher maximum system reliability values. Levitin [15] also presented one of the first papers devoted to component mixing for the reliability allocation problem. There are other benefits to mixing functionally equivalent components including a potentially reduced system reliability estimation variance [16], and reduced susceptibility to common cause failures.

The MWO heuristic involves the transformation of a single objective problem into one with multiple objectives. Several other papers have been published which address redundancy allocation with multiple objectives. For multi-objective problems, there are various approaches with differing attributes and advantages, but the two most popular are to combine the multiple objectives into a single objective using objective weights or penalties, or alternatively, to determine a set or subset of Pareto optimal solutions, i.e., non-dominated solutions. Dhingra [17] applied a multiple-criteria optimization approach to maximize system reliability, and minimize resource consumption (cost, weight, and volume). Misra & Sharma [18], [19] used integer programming, and a min-max concept to obtain Pareto optimal solutions. Li [20] considered dynamic programming to solve Dhingra's problem. Coit *et al.* [21] proposed a multiple objective formulation where the objectives were to maximize an estimate of the system reliability, and to minimize the variance of that estimate.

Busacca *et al.* [22] proposed a Multiple Objective Genetic Algorithm (MOGA). Their approach is based on identifying Pareto optimal solutions. The objective function associated with each solution is the "rank." All Pareto optimal solutions have rank one. As they are removed from the GA population, the Pareto optimal solutions from the remaining population have rank of two, three, and so on. They demonstrate their methodology on the system design of a standby safety system of a nuclear power plant.

Assumptions:

- 1) Failures of individual components are *s*-independent.
- 2) All redundancy is active. Redundant components are in a hot-standby state. The redundant components fail at the same rate as the primary component, and they are immediately activated in response to a failure without any loss of operational capability.
- 3) Failed components do not damage the system, and are not repaired.

II. PROBLEM FORMULATION

The problem objective is to maximize system reliability, $R(\mathbf{x})$, given constraints on the system, usually for system cost and weight. The system is configured as a series-parallel system. The number of components in each parallel configuration, i.e., subsystem, and the choice of components are the decision variables. For each subsystem, there are m_i functionally equivalent component types available, each with different reliability, cost and weight.

The MWO heuristic is based on an alternate or surrogate formulation. For the surrogate problem, the objective is to maximize the reliability of each subsystem individually to form a

multiple-objective optimization problem. It is logical that, if the reliability of each subsystem is maximized, then the assembled system reliability will also be high.

For multiple objective problems, there are two general solution strategies. The first is to combine the multiple objective functions into one composite function. Examples of this approach are the use of utility functions and goal programming. This is the approach used here to develop the MWO heuristic. The second general approach is to determine a Pareto-optimal set. This would not be effective for problems with a series-parallel structure. If this criteria had been used, then there could be a possible “optimal” solutions with one subsystem with very high reliability (i.e., very close to one), and others that were very low. Technically, that may be a Pareto optimal solution if no other feasible solution had a higher reliability for the one subsystem, but in practice, it would be a very poor solution for a series-parallel system because the product of all of the subsystems would be very low.

There are several important, distinctive features of the formulation presented here. First, unlike most formulations of the redundancy allocation problem, by formulating the problem in this manner, an equivalent linear formulation can be obtained through transformations (assuming linear constraints). Thus, standard integer programming tools & software can be used. Second, this formulation of the problem allows mixing of components as part of the linearized formulation, thus, not restricting the solution space. Although system reliability is not maximized in this approach directly, the resulting solutions do yield high system reliability. Also, the simplicity and reduced computational effort required, together with very good results, makes this approach attractive for some reliability design problems.

The multiple-objective problem is transformed into an equivalent problem through a sequence of algebraic operations. Then, the multiple objectives are combined using specified numerical weights. The system fails due to the failure of *any* subsystem for a series-parallel system, and therefore, all objectives (subsystem reliability) are equally important, and assigned equal weights. This problem can then be solved to yield an initial system design solution. Then, in a search for better solutions, an additional sequence of problems is solved. To create the new problems, there are several plausible possibilities. One alternative is to iteratively & systematically adjust the objective weights, ω_i , while another alternative is to iteratively introduce constraints that limit the minimum subsystem reliability.

The original problem formulation, and the surrogate multiple objective formulation, are presented below as Problems P1 & P2.

Problem 1:

$$\begin{aligned} \max_{\mathbf{x}} &= R(\mathbf{x})R_1(\mathbf{x}_1)R_2(\mathbf{x}_2)\dots R_s(\mathbf{x}_s) \\ \text{subject to} & \sum_i \sum_j c_{ij}x_{ij} \leq C \\ & \sum_i \sum_j w_{ij}x_{ij} \leq W \\ & x_{ij} \in Z^+ \end{aligned}$$

Problem 2:

$$\begin{aligned} \max_{\mathbf{x}} & \{R_1(\mathbf{x}_1), R_2(\mathbf{x}_2), \dots, R_s(\mathbf{x}_s)\} \\ \text{subject to :} & \sum_i \sum_j c_{ij}x_{ij} \leq C \\ & \sum_i \sum_j w_{ij}x_{ij} \leq W \\ & x_{ij} \in Z^+ \end{aligned}$$

Problem P2 is the multiple objective formulation. For a series-parallel system, Problem P2 can be re-stated as Problem P3 below.

Problem 3:

$$\begin{aligned} \max_{\mathbf{x}} & \left\{ 1 - \prod_{j=1}^{m_1} (1-r_{1j})^{x_{1j}}, 1 - \prod_{j=1}^{m_2} (1-r_{2j})^{x_{2j}}, \dots, \right. \\ & \left. 1 - \prod_{j=1}^{m_s} (1-r_{sj})^{x_{sj}} \right\} \\ \text{subject to :} & \sum_i \sum_j c_{ij}x_{ij} \leq C \\ & \sum_i \sum_j w_{ij}x_{ij} \leq W \\ & x_{ij} \in Z^+ \end{aligned}$$

Problem P3 is a multiple-objective, nonlinear integer programming problem that is difficult to solve. Fortunately, an equivalent linear formulation can be obtained through a series of objective function transformations. Use of equivalent objective functions creates a surrogate problem that is easier to solve, and this forms the basis of the MWO heuristic. An “equivalent objective function” is one that has the same optimal solution or solutions. For this problem, the equivalent, although not equal, objective functions are based on observations that:

- 1) an identical constant value can be subtracted from each individual objective, and the optimal solution will remain the same (but not the objective function);
- 2) a maximization problem can be readily transformed into a minimization problem; and
- 3) the solution that maximizes the logarithm of subsystem reliability also maximizes subsystem reliability.

The following sequence of equivalent objective functions demonstrates the alternative formulation.

i.

$$\max_{\mathbf{x}} \left\{ 1 - \prod_{j=1}^{m_1} (1 - r_{1j})^{x_{1j}}, 1 - \prod_{j=1}^{m_2} (1 - r_{2j})^{x_{2j}}, \dots, 1 - \prod_{j=1}^{m_s} (1 - r_{sj})^{x_{sj}} \right\}$$

ii.

$$\min_{\mathbf{x}} \left\{ \prod_{j=1}^{m_1} (1-r_{1j})^{x_{1j}}, \prod_{j=1}^{m_2} (1-r_{2j})^{x_{2j}}, \dots, \prod_{j=1}^{m_s} (1-r_{sj})^{x_{sj}} \right\}$$

iii.

$$\min_{\mathbf{x}} \left\{ \sum_{j=1}^{m_1} x_{1j} \ln(1-r_{1j}), \sum_{j=1}^{m_2} x_{2j} \ln(1-r_{2j}), \dots, \sum_{j=1}^{m_s} x_{sj} \ln(1-r_{sj}) \right\}$$

iv.

$$\max_{\mathbf{x}} \left\{ \sum_{j=1}^{m_1} \gamma_{1j} x_{1j}, \sum_{j=1}^{m_2} \gamma_{2j} x_{2j}, \dots, \sum_{j=1}^{m_s} \gamma_{sj} x_{sj} \right\},$$

$$\gamma_{ij} = -\ln(1-r_{ij})$$

Solutions to multiple-objective problems are often determined by combining the objectives into a single objective problem. This requires the user to specify objective function weights, or a utility function to represent the relative importance of the individual objective function. More complex utility functions are available [23], but the most straightforward approach is to combine the multiple objectives using the weighted sum of multiple objective functions. ω_i is defined as the weight assigned to the i^{th} objective, i.e., subsystem reliability. The weights are defined so that the sum of the weights is one.

The new formulation, defined as Problem P4 below, is linear, and can be solved using existing integer programming algorithms & software. Thus, a difficult multiple objective nonlinear integer programming problem (P3) has been transformed into a linear integer programming problem (P4). Solving this problem requires the selection of the objective function weights. Because each subsystem similarly causes failure of the entire system, $\omega_i = 1/s$ for all i .

Problem 4:

$$\max_{\mathbf{x}} \omega_1 \left(\sum_{j=1}^{m_1} \gamma_{1j} x_{1j} \right) + \omega_2 \left(\sum_{j=1}^{m_2} \gamma_{2j} x_{2j} \right) + \dots$$

$$+ \omega_s \left(\sum_{j=1}^{m_s} \gamma_{sj} x_{sj} \right) = \sum_{i=1}^s \sum_{j=1}^{m_i} \omega_i \gamma_{ij} x_{ij}$$

$$\text{subject to: } \sum_i \sum_j c_{ij} x_{ij} \leq C$$

$$\sum_i \sum_j w_{ij} x_{ij} \leq W$$

$$x_{ij} \in Z^+$$

Generally, solving Problem P4 directly does not lead to satisfactory system designs because the least reliable subsystem is too low, i.e., often zero. However, the formulation can be further revised by adding a linear constraint that limits the minimum subsystem reliability. R_i^{\min} is defined as an unacceptable subsystem reliability level that must be exceeded by subsystem i .

This adds a cutting plane that removes undesirable regions of the solution space. The new constraint is defined as a 'greater than' constraint, but in practice, it is changed to a 'greater than or equal' constraint by introducing a small ε term. The final problem formulation is as follows,

Problem 5:

$$\max_{\mathbf{x}} \sum_{i=1}^s \sum_{j=1}^{m_i} \omega_i \gamma_{ij} x_{ij}$$

$$\text{subject to: } \sum_i \sum_j c_{ij} x_{ij} \leq C$$

$$\sum_i \sum_j w_{ij} x_{ij} \leq W$$

$$\sum_{j=1}^{m_i} \gamma_{ij} x_{ij} > -\ln(1-R_i^{\min}), \quad i = 1, \dots, s$$

$$x_{ij} \in Z^+$$

Problem P5 can be solved using readily available integer programming algorithms & software. Once an initial solution to Problem P5 has been determined, the heuristic continues by searching for better solutions by solving a sequence of modified problems. There are several different possible approaches to create the sequence of modified problems. One approach is to incrementally modify R_i^{\min} . An effective strategy is to set R_i^{\min} for $i = 1, \dots, s$, equal to the minimum subsystem reliability of the previous solution. This approach guarantees that a different solution will be found in the next iteration. Another alternative is to incrementally increase initial R_i^{\min} values by adding a pre-specified Δ value in each iteration.

When the weights are assigned a constant value ($w_i = 1/s$), they can be eliminated without impacting the algorithm or resulting solutions. However, they do demonstrate that the solution approach is based on standard multiple objective optimization principles. Also, modification of these weights is another possible way to iteratively find superior solutions once an initial solution is found.

III. MWO HEURISTIC

There are two variations of the MWO heuristic: MWO1, and MWO2. They are very similar, but differ based on the determination of an initial solution, the stopping criteria, and the manner in which R_i^{\min} is updated. MWO1 is easier to implement. It requires the user to initially determine some feasible solution to the problem, but no other heuristic parameters. MWO2 requires the user to pre-specify a Δ value. It has been experimentally observed to yield better solutions, but its performance depends on the parameter selections.

MWO1 Heuristic:

- Step 1) Determine a feasible solution (\mathbf{x}°) to the problem using any available method, e.g., a greedy heuristic such as Nakagawa-Nakashima [9], or Li [11]. Set $\omega_i = 1/s$, and initialize $R_i^{\min} = R^* = R(\mathbf{x}^\circ)$ and $q = 1$.
- Step 2) Solve problem P5 using any linear programming tool. If problem P5 is infeasible, go to Step 3.

TABLE I
RESULTS FOR TEST PROBLEM 1

Case	C	W	MWO2		TS	GA	Min-Max	
			R	CPU^*	MPI	MPI	MPI	CPU^*
1	130	191	0.98425	2.0	-19.43%	-18.43%	-3.59%	409.5
2	130	190	0.98455	2.6	-13.72%	-8.03%	1.60%	315.5
3	130	189	0.98445	2.2	-10.43%	-7.96%	3.40%	299.5
4	130	188	0.98426	2.1	-7.67%	-4.96%	8.17%	632.1
5	130	187	0.98367	2.0	-6.68%	-4.71%	-1.02%	12.0
6	130	186	0.98370	1.8	-3.00%	0.62%	7.27%	426.3
7	130	185	0.98350	1.8	0.00%	2.40%	6.41%	151.0
8	130	184	0.98272	1.9	-1.62%	2.36%	13.97%	49.0
9	130	183	0.98223	1.7	-0.17%	1.80%	5.73%	212.5
10	130	182	0.98014	1.8	-7.44%	-5.06%	2.87%	152.5
11	130	181	0.97995	1.8	-5.69%	-1.28%	0.00%	138.8
12	130	180	0.97823	1.6	-10.43%	-7.22%	2.62%	157.4
13	130	179	0.97804	1.5	-7.16%	-5.08%	0.00%	248.7
14	130	178	0.97714	1.3	-5.82%	-5.34%	-0.37%	126.1
15	130	177	0.97669	1.2	-3.48%	-2.24%	-2.59%	79.0
16	130	176	0.97649	1.2	-0.84%	0.40%	0.00%	38.0
17	130	175	0.97532	1.2	-1.60%	0.08%	7.18%	55.5
18	130	174	0.97316	1.1	-6.45%	-4.63%	-4.90%	13.5
19	130	173	0.97271	1.0	-4.26%	-3.44%	-0.28%	5.7
20	130	172	0.97252	1.0	-1.89%	-0.52%	9.54%	1.7
21	130	171	0.97135	1.0	-2.08%	-1.83%	6.52%	1.1
22	130	170	0.96895	0.9	-6.20%	-6.20%	0.58%	3.3
23	130	169	0.96760	0.7	-5.51%	-5.27%	3.65%	14.3
24	130	168	0.96692	0.7	-3.79%	-3.80%	4.91%	2.3
25	130	167	0.96634	0.6	0.00%	-0.01%	5.09%	1.3
26	130	166	0.96395	0.6	-3.13%	-3.12%	0.00%	1.5
27	130	165	0.96128	0.5	-6.70%	-6.70%	1.53%	1.9
28	130	164	0.95891	0.5	-9.36%	-9.35%	-1.52%	1.1
29	130	163	0.96064	0.5	1.66%	0.01%	4.36%	0.9
30	130	162	0.95827	0.4	0.15%	-2.08%	0.00%	0.7
31	130	161	0.95615	0.4	-1.80%	-4.49%	0.00%	0.9
32	130	160	0.95287	0.4	-6.15%	-6.31%	-4.26%	0.7
33	130	159	0.95268	0.3	-3.59%	-3.58%	0.00%	0.7

$MPI=100 \times (R_{MWO2} - R_{(Other\ Method)}) / (1 - R_{(Other\ Method)})$

* CPU seconds using AMPL/CPLEX v9.0 on a 2.6Ghz LINUX PC

Determine, and retain both values \mathbf{x}^q , and $R(\mathbf{x}^q)$; and go to Step 3.

Step 3) If $R(\mathbf{x}^q) < R^*$, or the problem is infeasible, STOP. The recommended solution is \mathbf{x}^{q-1} , and $R(\mathbf{x}^{q-1})$.

If $R(\mathbf{x}^q) \geq R^*$, set $R^* = R(\mathbf{x}^q)$, $R_i^{\min} = \min_j R_j(\mathbf{x}_j^q)$ for $i = 1, \dots, s$, and $q \leftarrow q + 1$; and return to Step 2.

MWO2 Heuristic:

Step 1) Select the Δ value. Set $R_i^{\min} = 0$ for $i = 1, \dots, s$, $\omega_i = 1/s$, and initialize $q = 1$.

Step 2) Solve problem P5 using any integer programming tool. If problem P5 is infeasible, go to Step 4. If feasible, retain \mathbf{x}^q , and $R(\mathbf{x}^q)$; and go to Step 3.

Step 3) $k = \arg \min_i (R_i(\mathbf{x}_i^q))$, $R_k^{\min} \leftarrow R_k(\mathbf{x}_k^q) + \Delta$, and $q \leftarrow q + 1$. Go to Step 2.

Step 4) Solutions are $R^* = \max_q R(\mathbf{x}^q)$,
 $v = \arg \max_q R(\mathbf{x}^q)$, $\mathbf{x}^* = \mathbf{x}^v$

IV. EXAMPLES

The performance of the MWO heuristic was tested using four different problems. The MWO heuristic does not perform as well as meta-heuristic approaches like GA or TS, but its performance is close to that of those approaches; and it can be implemented using standard optimization, and integer programming

algorithms & software. Compared to traditional heuristics, the performance of the MWO heuristic is generally superior. The MWO2 version was used for the comparisons. It provides superior performance to MWO1, but it also requires additional parameter settings, although still far fewer than GA or TS.

The heuristic was coded in AMPL/CPLEX v9.0 for LINUX, and all runs were performed on a PC with a 2.6 GHz CPU, and 1.5 GB memory. The results were compared to other methods from the literature. In the experiments, MWO2 was used with $\Delta = 0.00001$.

A. Test Problem 1

The first test problem originally proposed by Fyffe *et al.* [3] includes 14 subsystems with three or four component choices for each subsystem. This problem was also used by many other researchers [4], [6], [7], [13]. As in Coit & Smith [6], and Kulturel-Konak *et al.* [7], 33 variations of the unrestricted problem, where mixing functionally equal components within subsystems is allowed, were studied by fixing the cost constraint limit C at 130, and changing the weight constraint limit W from 159 to 191. The best results for these 33 problems were generally found by the GA approach of Coit & Smith [6], or the TS approach of Kulturel-Konak *et al.* [7].

The MWO heuristic is based on a concept similar to the max-min heuristic of Ramirez-Marquez *et al.* [13]. The max-min heuristic has been demonstrated [14] to generally

TABLE II
RESULTS FOR PROBLEMS 2, 3, AND 4

<i>i</i>	<i>C</i>	<i>W</i>	Problem 2		Problem 3			Problem 4	
			<i>R</i>	<i>MPI-TS</i>	<i>R</i>	<i>MPI-TS</i>	<i>CPU</i>	<i>R</i>	<i>MPI-TS</i>
1	100	100	0.05268	-1.07%	0.17265	nfs*	0.1	0.14722	-2.19%
2	100	130	0.07702	-0.97%	0.25035	Nfs	15.2	0.32840	-0.61%
3	100	160	0.07702	-0.97%	0.31659	0.31%	9.4	0.55503	-2.20%
4	100	190	0.07702	-0.97%	0.39429	0.41%	5.7	0.64773	1.09%
5	100	220	0.07702	-0.97%	0.48209	0.07%	9.5	0.72987	-1.35%
6	100	250	0.07702	-0.97%	0.58655	-0.22%	4.8	0.76426	-9.21%
7	130	100	0.08091	0.00%	0.24129	Nfs	13.2	0.31355	3.57%
8	130	130	0.26884	-0.40%	0.31593	0.26%	8.8	0.56078	0.38%
9	130	160	0.31506	-1.98%	0.39476	0.44%	8.6	0.66594	1.56%
10	130	190	0.31061	-1.55%	0.48277	-0.06%	4.4	0.74961	-1.26%
11	130	220	0.31470	-0.95%	0.59041	0.29%	3.7	0.83421	-2.10%
12	130	250	0.31061	-1.55%	0.71971	0.41%	4.9	0.89123	-1.46%
13	160	100	0.08091	0.00%	0.30389	Nfs	13.2	0.53033	4.46%
14	160	130	0.28595	0.31%	0.38961	-0.11%	14.7	0.64802	1.97%
15	160	160	0.52941	-3.16%	0.48297	-0.21%	3.6	0.74884	-2.00%
16	160	190	0.65762	0.48%	0.59047	-0.19%	3.7	0.82922	-6.21%
17	160	220	0.65762	0.48%	0.75473	11.58%	3.2	0.89751	-0.25%
18	160	250	0.65762	0.48%	0.88177	0.67%	10.3	0.92416	-1.81%
19	190	100	0.08091	0.00%	0.38067	Nfs	3.4	0.60364	-0.50%
20	190	130	0.28595	0.31%	0.48074	20.39%	12.8	0.72425	-4.06%
21	190	160	0.60371	1.35%	0.61995	7.14%	11.9	0.82697	-0.64%
22	190	190	0.75874	-2.96%	0.75809	12.79%	10.7	0.89718	2.04%
23	190	220	0.79847	-12.27%	0.88293	-0.92%	10.2	0.92675	1.09%
24	190	250	0.80097	-10.88%	0.90783	0.71%	7.8	0.94725	-2.97%
25	220	100	0.08091	0.00%	0.46579	Nfs	3.2	0.63909	-5.90%
26	220	130	0.28595	0.31%	0.58585	36.50%	3.9	0.79882	-0.37%
27	220	160	0.60371	1.15%	0.72885	44.78%	10.2	0.88788	-1.83%
28	220	190	0.79824	-5.12%	0.88214	8.32%	12.1	0.92239	1.24%
29	220	220	0.87281	-2.22%	0.90616	0.95%	19.2	0.94678	-2.60%
30	220	250	0.89454	-3.55%	0.92302	-2.70%	13.7	0.96779	1.56%
31	250	100	0.08091	0.00%	0.54857	Nfs	4.3	0.63909	-8.03%
32	250	130	0.28595	0.31%	0.71172	55.80%	3.9	0.84744	-3.30%
33	250	160	0.60371	1.15%	0.87444	74.43%	8.3	0.91345	0.46%
34	250	190	0.79824	-5.12%	0.90171	0.33%	12.6	0.94285	3.83%
35	250	220	0.88581	-4.46%	0.92388	0.48%	19.2	0.96401	-4.60%
36	250	250	0.92957	-1.07%	0.94080	1.68%	24.8	0.98031	2.52%

$$\text{MPI-TS} = 100 \times (R_{\text{MWO2}} - R_{\text{(Tabu Search)}}) / (1 - R_{\text{(Tabu Search)}})$$

* nfs=No feasible solution was found.

yield superior solutions based on numerous randomly generated problems, and comparisons to the Nakagawa & Nakashima heuristic [9]. Therefore, this heuristic is also included in the comparisons. The results herein for the min-max heuristic were found in this paper using the same software & hardware as with the MWO2 heuristic implementation.

The results are given in Table I. In this table, the maximum system reliability is given for the MWO2 heuristic. For the other methods, the Maximum Possible Improvement (MPI), which is the percent difference between the system reliability found by the MWO2 heuristic and the other methods with a maximum system reliability of one. Therefore, a positive MPI indicates that the MWO2 heuristic surpassed the maximum system reliability of the competing approach. The CPU times given in the table for the MWO2, and the max-min heuristic, are the total solution times to find final solutions. Because the experiments for these two heuristics were performed on the same computer using the same version of AMPL/CPLEX, a comparison of their CPU times is meaningful.

As seen in Table I, the MWO2 heuristic outperformed the min-max heuristic in the majority of the test cases. The MWO2 heuristic obtained the higher or same system reliability in 25 of the 33 cases. In addition to solution quality, a major advantage of the MWO2 heuristic compared to the max-min heuristic is

its computational efficiency. The maximum CPU time of the MWO2 heuristic was 2.6 seconds, while the max-min heuristic required more than a minute in most cases. This large difference in the computational efficiency of these two heuristics is not surprising. The linear programming relaxation of the MIP model used in the max-min heuristic is a degenerate max-min LP problem which is difficult to solve. On the other hand, Problem P5 has a very simple constraint structure, and its LP relaxation can be efficiently solved.

The MWO2 heuristic resulted in solutions that approached, but rarely exceeded, the comparable solutions from GA or TS. The MWO2 provided a higher system reliability than the GA in only 7 cases, and a higher system reliability than the TS in only 3 cases. However, the major advantage of the MWO2 heuristic compared to these meta-heuristics approaches is its ease of implementation. The MWO2 heuristic can be implemented using a commercially available MIP solver without requiring any programming skills. Alternatively, the GA, and TS approaches require advanced programming skills to customize, code, and test. This might be one reason that practitioners have been slow to adopt meta-heuristics although very promising results have been demonstrated. CPU times were not provided for the GA, and TS approaches; and a CPU time comparison is also not meaningful because different hardware was used.

TABLE III
DATA FOR PROBLEMS 2, 3, AND 4

<i>i</i>	Problem 2			Problem 3			Problem 4		
	c_{i1}, \dots, c_{i4}	w_{i1}, \dots, w_{i4}	r_{i1}, \dots, r_{i4}	c_{i1}, \dots, c_{i4}	w_{i1}, \dots, w_{i4}	r_{i1}, \dots, r_{i4}	c_{i1}, \dots, c_{i4}	w_{i1}, \dots, w_{i4}	r_{i1}, \dots, r_{i4}
1	3, 5, 8, 10	3, 5, 8, 10	.71, .82, .9, .99	3, 5, 6, 10	9, 7, 4, 1	.92, .9, .91, .92	3, 4, 7, 10	4, 3, 10, 7	.81, .8, .96, .98
2	2, 5, 8, 10	4, 6, 8, 10	.7, .82, .92, .98	2, 5, 7, 10	10, 6, 5, 3	.92, .9, .92, .91	3, 5, 7, 8	4, 2, 10, 7	.82, .82, .97, .98
3	3, 5, 7, 10	2, 6, 8, 10	.7, .83, .91, .97	1, 5, 7, 9	9, 6, 4, 1	.91, .93, .93, .92	1, 5, 6, 9	4, 1, 10, 7	.82, .8, .96, .97
4	4, 6, 8, 9	2, 6, 7, 9	.72, .82, .93, .97	1, 5, 7, 9	9, 7, 5, 3	.93, .93, .92, .9	2, 5, 6, 9	5, 2, 8, 7	.83, .8, .96, .96
5	2, 5, 7, 9	2, 5, 7, 10	.73, .81, .92, .97	1, 5, 6, 9	10, 7, 4, 2	.93, .91, .9, .91	3, 5, 7, 8	5, 2, 10, 7	.82, .81, .96, .98
6	3, 5, 7, 9	3, 5, 8, 10	.71, .81, .92, .99	2, 4, 7, 9	8, 7, 5, 1	.91, .9, .9, .92	1, 4, 7, 10	5, 3, 10, 6	.8, .82, .98, .98
7	2, 6, 8, 9	4, 5, 7, 10	.73, .83, .92, .98	1, 5, 6, 9	10, 7, 4, 3	.93, .9, .9, .9	1, 4, 6, 8	5, 2, 9, 7	.81, .83, .98, .98
8	4, 5, 7, 10	4, 5, 8, 10	.73, .8, .91, .98	2, 5, 7, 9	8, 7, 5, 3	.92, .92, .91, .92	3, 4, 6, 9	4, 1, 9, 7	.8, .83, .98, .98
9	4, 6, 7, 9	4, 6, 8, 10	.72, .82, .91, .99	1, 4, 6, 8	8, 7, 5, 2	.9, .91, .93, .9	2, 4, 6, 10	5, 3, 9, 6	.83, .82, .97, .96
10	3, 6, 8, 10	2, 6, 7, 9	.72, .83, .92, .99	1, 5, 7, 9	10, 7, 5, 1	.93, .91, .92, .91	2, 4, 6, 9	4, 3, 9, 7	.81, .81, .96, .98
11	4, 5, 7, 9	2, 5, 7, 10	.71, .83, .93, .97	1, 4, 6, 9	10, 6, 4, 2	.93, .93, .9, .91	1, 5, 6, 8	5, 1, 10, 6	.82, .81, .97, .98
12	2, 5, 8, 9	3, 6, 8, 9	.71, .81, .91, .97	1, 5, 7, 8	9, 6, 4, 1	.9, .93, .9, .9	2, 5, 6, 8	5, 3, 9, 6	.83, .83, .98, .97
13	2, 6, 8, 9	2, 5, 7, 10	.72, .83, .91, .98	1, 4, 6, 10	9, 6, 4, 1	.93, .9, .91, .92	1, 5, 6, 8	4, 3, 8, 6	.81, .81, .96, .98
14	2, 6, 7, 10	4, 6, 8, 10	.73, .83, .9, .98	2, 5, 6, 9	10, 6, 5, 1	.93, .92, .91, .91	3, 5, 6, 9	5, 3, 8, 6	.81, .8, .97, .96
15	3, 6, 8, 10	2, 5, 8, 10	.73, .83, .93, .98	2, 4, 6, 9	8, 7, 5, 3	.93, .93, .91, .9	2, 4, 6, 10	5, 1, 8, 7	.8, .82, .98, .98
16	4, 6, 7, 10	4, 5, 8, 10	.71, .83, .92, .98	2, 4, 6, 10	9, 6, 4, 3	.91, .9, .93, .91	1, 5, 6, 8	4, 2, 9, 7	.82, .82, .98, .98
17	3, 5, 7, 10	2, 6, 7, 9	.7, .8, .92, .97	2, 5, 7, 9	10, 6, 4, 3	.9, .9, .92, .92	1, 4, 7, 8	5, 2, 10, 7	.8, .83, .97, .96
18	2, 6, 7, 10	3, 5, 8, 9	.72, .8, .93, .99	1, 5, 6, 9	8, 7, 5, 2	.91, .92, .9, .91	3, 5, 6, 10	4, 2, 9, 7	.8, .8, .97, .96
19	2, 6, 8, 9	4, 5, 8, 10	.71, .8, .93, .97	3, 5, 7, 8	8, 6, 4, 3	.92, .9, .9, .93	2, 4, 7, 9	4, 2, 8, 6	.81, .83, .98, .98
20	4, 6, 8, 10	3, 5, 8, 9	.7, .83, .9, .99	1, 5, 6, 8	9, 7, 5, 2	.91, .91, .9, .93	1, 4, 6, 9	5, 3, 9, 6	.8, .8, .98, .98

B. Test Problems 2, 3, and 4

Although Problem 1 is a well-known test problem, and it has been extensively used in the literature to evaluate alternative approaches to solve the redundancy allocation problem, it has some drawbacks as a test problem. For example, some of component options clearly dominate others, i.e., they have lower cost and weight, but higher reliability than others. In fact, many component options can be eliminated if the problem data are carefully analyzed. This feature of Problem 1 favors meta-heuristic & heuristic approaches that depend on local perturbation operators because difficult tradeoffs do not exist among the component options.

In this section, three new problems with difficult tradeoffs between components are introduced to evaluate the MWO heuristic. Each problem has 20 subsystems with four available component options for each subsystem. The maximum number of components allowed in each subsystem is 8. These problems have an extremely large solution space with 4^{160} possible configurations.

The data for the problems were generated randomly according to the following rules. In Problem 2, component types are positively correlated (i.e., $c_{i1} < c_{i2} < c_{i3} < c_{i4}$, $w_{i1} < w_{i2} < w_{i3} < w_{i4}$, and $r_{i1} < r_{i2} < r_{i3} < r_{i4}$ for each subsystem i). In Problem 3, component types have negatively correlated costs, and weights (i.e., $c_{i1} < c_{i2} < c_{i3} < c_{i4}$, and $w_{i1} > w_{i2} > w_{i3} > w_{i4}$ for each subsystem i); and uniform reliability ($r_{i1} \approx r_{i2} \approx r_{i3} \approx r_{i4}$ for each subsystem i). The data of this problem reflects a real life case where component size and weight could be reduced at the tradeoff of increasing its cost. Problem 4 is a mixture of Problems 2 & 3

where $c_{i1} < c_{i2} < c_{i3} < c_{i4}$, $w_{i2} < w_{i1} < w_{i4} < w_{i3}$, and $r_{i1} \approx r_{i2} < r_{i3} \approx r_{i4}$ for each subsystem i . In this problem, there are two main groups of components; low reliability components with low cost, and weight (types 1, and 2); and high reliability components with high cost, and weight (types 3, and 4). In each group, there is a component with a relatively low cost, and relatively high weight (e.g., type 1); and one with a relatively high cost, and relatively low weight (e.g. type 2). The data of these problems are given in Table II.

These three problems were solved for 36 different combinations of cost & weight constraint limits by using the MWO2 heuristic, and the TS approach of Kulturel-Konak *et al.* [7], which found most of the best reported solutions for Problem 1. Using the same TS parameters given in their paper, the TS code (obtained from the authors) was run 10 times with different random number seeds on a Sun UltraSPARC Workstation with four 450 MHz CPU, and 4 GB memory; and the best solutions found in 10 runs were used in the comparisons. The min-max heuristic was also applied to the problems, but it could not solve the problems within a reasonable time. The results are summarized in Table III.

As seen in Table III, the TS approach outperformed the MWO2 in the majority of the 36 cases of Problem 2. Compared to Problem 1, however, their performances were much closer to the MWO2 heuristic, obtaining higher system reliabilities in 15 of the 36 cases. The MWO2 heuristic was proved to be superior to the TS approach in Problem 3. For this problem, the MWO2 heuristic obtained higher system reliabilities in 29 of the 36 cases, and the TS search approach could not find feasible solutions for 9 cases. In cases where the TS approach dominated,

the differences between these two approaches were also small. However, the MWO2 heuristic performed substantially better in several cases.

Problem 3 is the most challenging problem to solve among all problems studied. In this problem, constraint coefficients c_{ij} , and w_{ij} are negatively correlated, which makes it very difficult to satisfy the cost, and weight constraints simultaneously. It should be also noted that such conflicting constraint coefficients are very common in many actual system design problems. Notice that the MWO2 heuristic found feasible solutions in every case of Problem 3. In fact, if the MWO2 heuristic cannot find a feasible solution for a problem, then it implies that no feasible solution exists for the problem because Problems P4 & P5 have the same feasible region at the starting point of the MWO2 heuristic, where $R_i^{\min} = 0$ for each subsystem i . The results for Problem 4 are similar to those of Problem 2. The MWO2 heuristic found higher system reliabilities in 13 cases.

V. CONCLUSIONS

The MWO heuristic is based on a transformation of the original problem into a multiple objective problem. The heuristic provides an efficient method to determine solutions to this important reliability optimization problem. It was tested on many example problems with good results. Additionally, the MWO2 heuristic can provide a quick check of the feasibility for more difficult, nonlinear problem formulations.

As the example results indicate, in addition to its simplicity and ease of implementation, the MWO2 heuristic was proved to be a good contender to more elaborate approaches, such as GA, and TS, to solve the redundancy allocation problem. Another concern about the applicability of the MWO2 heuristic is solution time. For many combinatorial problems, integer programming formulations can be solved for small problems (i.e., less than five subsystems), but it is computationally impossible to solve large ones (i.e., greater than ten or so subsystems). In this section, the MWO2 heuristic was tested using large problems to evaluate its computational performance. Due to abbreviation, the CPU times are presented only for Problem 3. As seen in the table, most instances were solved in under 15 seconds. Considering the size of the problems, obtained CPU times are very promising, indicating applicability of the MWO2 heuristic to large sized problems.

REFERENCES

- [1] W. Kuo, V. Prasad, F. Tillman, and C. L. Hwang, *Optimal Reliability Design: Fundamentals and Applications*. Cambridge, UK: Cambridge University Press, 2000.
- [2] M. S. Chern, "On the computational complexity of reliability redundancy allocation in a series system," *Operations Research Letters*, vol. 11, pp. 309–315, June 1992.
- [3] D. E. Fyffe, W. W. Hines, and N. K. Lee, "System reliability allocation problem and a computational algorithm," *IEEE Transactions on Reliability*, vol. 17, pp. 64–69, June 1968.
- [4] Y. Nakagawa and S. Miyazaki, "Surrogate constraints algorithm for reliability optimization problems with two constraints," *IEEE Transactions on Reliability*, vol. 30, pp. 175–180, 1981.
- [5] P. M. Ghare and R. E. Taylor, "Optimal redundancy for reliability in series system," *Operations Research*, vol. 17, pp. 838–847, September 1969.
- [6] D. W. Coit and A. E. Smith, "Reliability optimization for series-parallel systems using a genetic algorithm," *IEEE Transactions on Reliability*, vol. 45, no. 2, pp. 254–260, June 1996.
- [7] S. Kulturel-Konak, A. E. Smith, and D. W. Coit, "Efficiently solving the redundancy allocation problem using Tabu search," *IIE Transactions*, vol. 35, no. 6, pp. 515–526, June 2003.
- [8] C. S. Sung and Y. K. Cho, "Branch and bound redundancy optimization for a series system with multiple-choice constraints," *IEEE Transactions on Reliability*, vol. 48, no. 2, pp. 108–117, June 1999.
- [9] Y. Nakagawa and K. Nakashima, "A heuristic method for determining optimal reliability allocation," *IEEE Transaction on Reliability*, vol. 26, pp. 156–161, 1977.
- [10] W. Kou, C. L. Hwang, and F. A. Tillman, "A note on heuristic method for in optimal system reliability," *IEEE Transaction on Reliability*, vol. 27, pp. 320–324, 1978.
- [11] J. Li, "A bound heuristic algorithm for solving reliability redundancy optimization," *Microelectronics & Reliability*, vol. 36, no. 3, pp. 335–339, 1996.
- [12] Y. C. Hsieh, "A linear approximation for redundant reliability problems with multiple component choices," *Computers & Industrial Engineering*, vol. 44, pp. 91–103, 2002.
- [13] J. E. Ramirez-Marquez, D. W. Coit, and A. Konak, "Reliability optimization of series-parallel systems using a max-min approach," *IIE Transactions*, vol. 36, no. 9, September 2004.
- [14] H. Lee, W. Kuo, and C. Ha, "Comparison of max-min approach and NN method for reliability optimization of series-parallel system," *Journal of System Science and Systems Engineering*, vol. 12, no. 1, pp. 39–48, 2003.
- [15] G. Levitin, A. Lisnianski, and D. Elmakis, "Structure optimization of power system with different redundant elements," *Electric Power Systems Research*, vol. 43, pp. 19–27, 1997.
- [16] T. Jin and D. W. Coit, "Variance of system reliability estimates with arbitrarily repeated components," *IEEE Transactions on Reliability*, vol. 50, no. 4, pp. 409–413, December 2001.
- [17] A. Dhingra, "Optimal apportionment of reliability & redundancy in series systems under multiple objectives," *IEEE Transactions on Reliability*, vol. 41, no. 4, pp. 576–582, 1992.
- [18] K. Misra and U. Sharma, "An effective approach for multiple criteria redundancy optimization problems," *Microelectronics and Reliability*, vol. 31, no. 2/3, pp. 303–321, 1991.
- [19] —, "Multi-criteria optimization for combined reliability and redundancy allocation in systems employing mixed redundancies," *Microelectronics and Reliability*, vol. 31, no. 2/3, pp. 323–335, 1991.
- [20] D. Li, "Interactive parametric dynamic programming and its application of large system reliability," *Journal of Mathematical Analysis and Applications*, vol. 191, pp. 589–607, 1995.
- [21] D. W. Coit, T. Jin, and N. Wattanapongsakorn, "System optimization considering component reliability estimation uncertainty: a multi-criteria approach," *IEEE Transactions on Reliability*, vol. 53, no. 3, September 2004.
- [22] P. G. Busacca, M. Marseguerra, and E. Zio, "Multiobjective optimization by genetic algorithms: application to safety systems," *Reliability Engineering & System Safety*, vol. 72, pp. 59–74, 2001.
- [23] R. L. Keeney and H. Raiffa, *Decisions With Multiple Objectives: Preferences and Value Tradeoffs*. Cambridge, UK: Cambridge University Press, 1993.

David W. Coit is an Associate Professor in the Department of Industrial & Systems Engineering at Rutgers University. He received a BS degree in Mechanical Engineering from Cornell University, an MBA from Rensselaer Polytechnic Institute, and MS & PhD in Industrial Engineering from the University of Pittsburgh. In 1999, he was awarded a CAREER grant from NSF to study reliability optimization. He also has over ten years of experience working for IIT Research Institute (IITRI), Rome, NY, where he was a reliability analyst, project manager, and an engineering group manager. His current research involves reliability prediction & optimization, risk analysis, and multi-criteria optimization considering uncertainty.

Abdullah Konak is an Assistant Professor of Information Sciences and Technology at Penn State Berks. He received his B.S. degree in Industrial Engineering from Yildiz Technical University, his M.S. in Industrial Engineering from Bradley University, and his Ph.D. in Industrial Engineering from University of Pittsburgh. Previous to this position, he was an instructor in the Department of Systems and Industrial Engineering at Auburn University for two years. His current research interest is in the application of Operations Research techniques to complex problems, including such topics as telecommunication network design, network reliability analysis/optimization, facilities design, and data mining.