Journal of the Korean Society of Marine Engineering, Vol. 38, No. 2 pp. 225~232, 2014 ISSN 2234-7925 Print
http://dx.doi.org/10.5916/jkosme.2014.38.2.225 ISSN 2234-8352 Online

A modified tabu search for redundancy allocation problem of complex systems of ships

Jae-hwan Kim[1] · Kil-woong Jang†

(Received January 15, 2014 ; Revised February 10, 2014 ; Accepted February 17, 2014)

Abstract: The traditional RAP (Redundancy Allocation Problem) of complex systems has considered only the redundancy of subsystem with homogeneous components. In this paper we extend it as a RAP of complex systems with heterogeneous components which is more flexible than the case of homogeneous components. We model this problem as a nonlinear integer programming problem, find its optimal solution by tabu search, and suggest an example of the efficient reliability design with heterogeneous components. In order to improve the quality of the solution of the tabu search, we suggest a modified tabu search to employ an adaptive procedure (1-opt or 2-opt exchange) to generate the efficient neighborhood solutions. Computational results show that our modified procedure obtains better solutions as the size of problem increases from 30 to 50, even though it requires rather more computing time. With some adjustment of the parameters of the proposed method, it can be applied to the optimal reliability designs of complex systems of ships.

Keywords: RAP, complex systems of ships, heterogeneous components, modified tabu search

1. Introduction

The RAP is a well-known combinatorial optimization problem which has been applied to reliability system designs such as semi-conductor integrated circuits, nanotechnology, and most electronic systems of ships, etc. The RAP is to determine the optimal number of redundant component in order to maximize the system reliability constrained to resource restrictions or system-level constraints for cost and weight. Many researchers have studied the RAP for various system structures of series, series-parallel, complex system, k-out-of-n, etc. Fyffe *et al*. [1] originallyset up the problem and suggested a solution algorithm utilizing a dynamic programming approach. Nakagawa and Miyazaki [2] developed 33 variations of Fyffe's problem, where the weight constraint varied its value from 159 to 191. Coit and Liu [3] proposed zero-one integer programming for small size of problems. They constrained the solution space so that only the homogeneous component type can be allowed for each subsystem.

On the contrary, Coit and Smith [4] extended Fyffe's problem in such a way that the parallel system could be more flexible. They allowed a mixing of component (i.e., heterogeneous component) types within a subsystem and employed GA (genetic algorithm) to obtain optimal solutions. RAP is usually formulated as a nonlinear integer programming problem which is generally difficult to find an exact optimal solution due to the considerable amount of computational effort. Therefore, various metaheuristic algorithms have been developed. Kultruel-Konak *et al.*

† Corresponding Author: Department of Data Information, Korea Maritime and Ocean University, 727 Taejoung-ro, Yeongdo-gu, Busan, 606-791, Korea, E-mail: jangkw@kmou.ac.kr, Tel: 051-410-4375

1 Department of Data Information, Korea Maritime and Ocean University, 727 Taejoung-ro, Yeongdo-gu, Busan, 606-791, Korea, E-mail: jhkim@kmou.ac.kr, Tel: 051-410-4374

[5] developed TS (tabu search) algorithm for the RAP of series-parallel system, and it is hitherto known that solutions of Kultruel-Konak etal. [5] are the best among the other metaheuristics such as GA (Coit & Smith [4]), ACO (ant colony optimization; Liang & Smith [6]), VNS (variable neighborhood search; Liang & Chern [7]), and hybrid meta-heuristics (Nahas *et al*. [8]). Recently, Ouzineb *et al.* [9] developed a hybrid metaheuristic which is com-bined TS with GA and obtained the same results of Kultruel-Konak *et al.* [5].

In the meanwhile, solutions for RAP of complex systems have been suggested by several authors (Ha & Kuo [10]). All authors have dealt with only the case of homogeneous components. In this paper, we suggest a RAPCH (RAP of complex systems with heterogeneous components) and find its optimal sol-ution by the TS algorithm, the best in the literature. In addition, in order to enhance TS in terms of the quality of the solution, we develop a MTS (modified tabu search) to employ an adaptive mechanism com-bined with the TS. Computational results show that the proposed method obtains better results as the size of problem increases, even though it requires rather more computing time.

The rest of this paper is organized as follows. In Section 2, the model of RAPCH can be formulated as a nonlinear integer programming problem. In Section 3, we propose a MTS algorithm for the RAPCH to improve the quality of the solution of TS. In Section 4, a comparison of the case of the homogeneous components with that of the heterogeneous compo-nents is provided with an example, and in Section 5 we evaluate the performance of the proposed algo-rithm through the computational efforts. Finally, con-clusions and future research are discussed in Section 6.

2. Statement of the problem

Notation

s	number of subsystems
x	the feasible solution
x_0	the initial feasible solution
x_c	the current solution
x_{bf}, x_{binf}	the best feasible or infeasible solution found so far
x_{ij}	the number of redundancy of the *jth* available component used in subsystem *i*
m_i	the number of available components for subsystem *i*
$R_s(x)$	the system reliability
R_i, Q_i	the reliability and unreliability of subsystem *i*
$R_p(x)$	the penalty function of the system reliability
C, W	system-level constraints limits for cost and weight
c_{ij} ,w_{ij} ,r_{ij}	the cost, weight and reliability for the *jth* available component for subsystem *i*
U	upper bound of x_{ij}
iter	the number of iterations of MTS
Adapiter	for adaptive procedure of MTS, the predetermined number of iterations without improving the value of $R_s(x)$
Stopiter	the maximum number of stopping iterations

Assumptions

In the RAPCH, assumptions are as follows:

1. The system and all of its subsystems are *s*-coherent.
2. The system consists of *s* subsystems, each of which is (1-out-of-x_{ij}: G).
3. All component states are s-independent.
4. Each constraint is an increasing function of x_{ij} and is additive among subsystems.

5. System reliability $R_s(x)$ is known in terms of the R_i.

The RAPCH optimization model can be generally formulated as the following nonlinear integer programming problem:

Maximize $R_s(\boldsymbol{x})$

subject to

$$\sum_{i=1}^{s}\sum_{j=1}^{m_i} c_{ij}x_{ij} \le C,$$

$$\sum_{i=1}^{s}\sum_{j=1}^{m_i} w_{ij}x_{ij} \le W,$$

$$1 \le \sum_{i=1}^{m_i} x_{ij} \le U, \text{ integers.} \tag{1}$$

This problem was proven to be a NP-hard problem (Chern [11]).

3. MTS Algorithm

To solve the RAPCH, we suggest a MTS to employ an adaptive procedure combined with the TS. The TS algorithm, first proposed by F. Glover [12], is a metaheuristic method to expand its search beyond local optimality using adaptive memory. The adaptive memory is a mechanism based on the tabu list of prohibited moves. The tabu list is one of the mechanisms to prevent cycling and guide the search towards unexplored region of the solution space. TS generally adopts the penalty function to allow to explore the search towards the promising infeasible region. The TS has been successfully applied to many combinatorial optimization problems such as ship routing problems, traveling salesman problems, time tabling problems, and resource allocation problems, etc.

Our MTS is based on the TS of Kultruel-Konak *et al.* [5]. The MTS consists of four parts which are i) construction of initial solutions and stopping criterion, ii) tabu list, iii) the structure of generating the neighborhood solutions, and iv) penalty function. Among them, iii) the structure of generating the neighborhood solutions is usually different according to the characteristics of the decision variables for the problem. The general steps of MTS can be summarized as follows:

```
Step 0. (Initialization) Set $iter$=0,and initialize
        tabu list.
Step 1. Randomly generate the initial solution
        $x_0$. Set $x_c = x_0$ and $x_{bf} = x_{binf} = x_c$.
Step 2.a. (Adaptive procedure)
        Set $iter = iter+1$.
        If $iter \le Adapiter$, generate the
        neighborhood of $x_c$ by the 1-opt
        exchange.
        Otherwise, generate the neighborhood
        of $x_c$ by the 2-opt exchange (see
        [13]).
Step 2.b. Select the best neighborhood which
        is not in the tabu list. Store it as
        the new current solution. Update
        the tabu list.
Step 3. If $x_c$ is feasible, then go to step 4.
        Else go to step 5.
Step 4. If $R_s(x_c) > R_s(x_{bf})$, then set $x_{bf} = x_c$,
        $iter = 0$, and initialize the tabu
        list. Go to step 2.a. Else go to step
        6.
Step 5. If $R_p(x_c) > R_s(x_{binf})$, then $x_{binf} = x_c$, $iter$
        $= 0$, and initialize the tabu list. Go
        to step 2.a.
Step 6. If $iter > Stopiter$, then goto step 7.
        Else go to Step 2.a.
```

Step 7. (End) Stop with the best feasible solution found so far.

3.1 Construction of initial solutions and stopping criterion

The initial solutions of MTS are randomly generated, and the scheme of randomly constructing the initial solutions is identical to the general scheme of local search heuristic. In our experiments, we try to find the optimal solution 10 times with different initial solutions for each problem. And the stopping criterion of MTS is defined as 1000 iterations without finding an improvement in the best feasible solution.

3.2 Tabu list

For the size of a tabu list, Kultruel-Konak *et al*. [5] showed that the dynamic size of the tabu list played a critical role in finding the better solutions for RAP. In our MTS, the size of the tabu list is also reset every 20 iterations to the value of between [s, $3s$] uniformly distributed for the long-term memory. Once the list is full, the oldest element of the tabu list should be removed as a new one is added.

3.3 The structure of generating the neighborhood solutions (Step 2.a: Adaptive procedure)

The TS generally adopts 1-opt exchange (Kultruel-Konak *et al*. [5]) as a strategy of generating the neighborhood solutions. It is apt to be trapped in a local optimum, regardless of the considerable amount of exploring solutions. In order to alleviate the risks of being trapped in such a local optimum, MTS employs an adaptive procedure (1-opt or 2-opt exchange) combined with the TS. In Step 2.a (Adaptive procedure), if $iter \leq Adapiter$, generate the neighborhood solutions of x_c by the 1-opt exchange. And if $iter > Adapiter$, generate the neighborhood solutions of x_c by the 2-opt exchange (see [13]). To efficiently find the neighborhood solutions, the 2-opt exchange undertakes the 1-opt exchange twice simultaneously in each iteration of MTS. As a result, it requires rather more computing time than the 1-opt exchange. That is, MTS repeats the 1-opt exchange until the value of *iter* is less than or equal to the value of *Adapiter.* After the predetermined value of *Adapiter* iteration, our MTS adopts the 2-opt exchange to escape the local optimum, even though it requires rather more computing time. From our computational experiments of sensitivity analysis, we noticed that the proper number of *Adapiter* (*i.e.,*the end phase of the 1-opt exchange heuristic) was 500.

3.4 Penalty function

The MTS generally adopts the penalty function to allow to explore the search towards the promising infeasible region. The penalty function of Kultruel-Konak *et al*. [5] is adopted in our MTS. The penalty function for two independent constraints (cost and weight) is as follows:

$$R_p(\mathbf{x}) = R_s(\mathbf{x}) - \left(R_{all} - R_{feas}\right)\left[\left(\frac{\Delta c}{NFT_c}\right)^{K_1} + \left(\frac{\Delta w}{NFT_w}\right)^{K_2}\right] \tag{2}$$

where R_{all} is the unpenalized (feasible or infeasible) system reliability of the best solution found so far, R_{feas} is the system reliability of the best feasible solution found so far, and Δc and Δw represent the magnitude for the violation of the cost and weight constraint, respectively. The initial values of NFT_c and NFT_w are set to 1% of the constraint limit for the cost and weight, respectively. The values of K_1 and K_2 are set to 1, though computational results are insensitive to these values.

The value of NFT_j at any given iteration j is updated as follows. If the current move is to a feasible solution, then

$$NFT_{j+1} = NFT_j\left(1+\frac{\rho_j}{2}\right) \tag{3}$$

If the current move is to a infeasible solution, then

$$NFT_{j+1} = NFT_j\left(\frac{1+\rho_j}{2}\right) \tag{4}$$

where ρ_j, a feasibility ratio at iteration j, is defined as

$$\rho_j = \frac{F_j}{T_j} \tag{5}$$

The tabu list size at any given iteration j is defined as T_j, and the number of feasible solutions on the current tabu list is defined as F_j. The value of *NFT* changes according to the count of the feasible versus infeasible solutions on the tabulist. If the current move is feasible, Eq. (3) has a property of encouraging the search toward the promising infeasible regions. That is, this decreases the penalty for an infeasible solution and moves the search toward the infeasible region. Reversely, if the current move is infeasible, Eq. (4) increases the penalty for an infeasible solution and explores the search backward the feasible region.

4. An Example

We suggest an example of RAPCH in this paper. The system structure and subsystem data are given by **Figure 1** and **Table 1**, respectively.

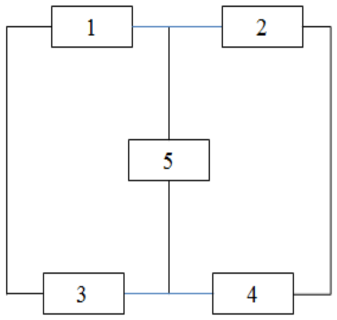


Figure 1: A bridge network system

The RAPCH is formulated as follows:

Maximize

$$R_s(x) = R_5(1-Q_1Q_3)(1-Q_2Q_4)+Q_5((1-(1-R_1R_2)(1-R_3R_4))$$

subject to

$$\sum_{i=1}^{s}\sum_{j=1}^{m_i} c_{ij}x_{ij} \le C,$$

$$\sum_{i=1}^{s}\sum_{j=1}^{m_i} w_{ij}x_{ij} \le W,$$

$$1 \le \sum_{i=1}^{m_i} x_{ij} \le U, \text{ integers.}$$

For example, R_1 is given by

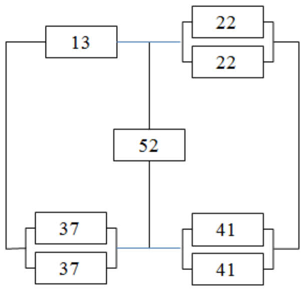
Table 1: Input data for heterogeneous components of bridge network system

S	1			2			3			4			5			6			7		
	c_{ij}	w_{ij}	r_{ij}	c_{ij}	w_{ij}	r_{ij}	c_{ij}	w_{ij}	r_{ij}	c_{ij}	w_{ij}	r_{ij}	c_{ij}	w_{ij}	r_{ij}	c_{ij}	w_{ij}	r_{ij}	c_{ij}	w_{ij}	r_{ij}
1	1	3	0.70	2	2	0.71	3	2	0.73	2	3	0.73	2	4	0.74	3	3	0.74	2	5	0.75
2	1	7	0.73	2	6	0.73	1	8	0.74	2	7	0.74	2	8	0.75	3	7	0.75			
3	1	5	0.65	2	5	0.66	1	6	0.67	3	4	0.69	3	5	0.70	4	3	0.71	4	4	0.72
4	3	5	0.63	3	6	0.64	5	4	0.65	5	5	0.66	4	6	0.67						
5	2	3	0.73	3	2	0.73	2	4	0.74	3	3	0.74	3	4	0.75	4	3	0.75	4	4	0.76

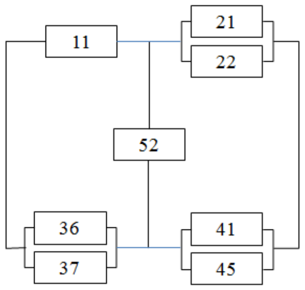
$$R_1 = \prod_{j=1}^{7}(1-(1-r_{1j})^{x_{1j}})$$

$=(1-0.3^{x_{11}}\times 0.29^{x_{12}}\times 0.27^{x_{13}}\times 0.27^{x_{14}}\times 0.26^{x_{15}}\times 0.26^{x_{16}}\times 0.25^{x_{17}})$

When C = 34 and W = 51 of this RAPCH, the TS yields an optimal solution, that is, $x_{11} = x_{21} = x_{22} = x_{36} = x_{37} = x_{41} = x_{45} = x_{52} = 1$, $R_s(x) = 0.981432$. The heterogeneous structure of this solution is shown in **Figure 2**(b).



(a) Homogeneous (R=0.980906)



(b) Heterogeneous (R=0.981432)

Figure 2: The optimal solutions for homogeneous and heterogeneous components

In **Figure 2**(a), the optimal solution for the case of homogeneous components is given by $x_{13} = 1$, $x_{52} = 1$, $x_{22} = 1$, $x_{37} = 1$, $x_{41} = 1$, $R_s(x) = 0.980906$. This result indicates that the design of heterogeneous components can provide the possibility of having the better system reliability than that of homogeneous components.

5. Computational Experiments

To additionally evaluate the performance of the TS and the proposed MTS, test problems for moderate size of system structure are designed. They are composed of 3 sets of 10 test problems, that is, 30 test problems totally. We replicated the data in **Table 1** by h-hold, where h ranges from 6, 8 to 10, i.e., the number of subsystems of each problem became $s = 5 \times h$ (*i.e.,* 30, 40, 50).

As mentioned in section 3, TS generally adopts 1-opt exchange procedure as a strategy of generating the neighborhood solutions. It is apt to be trapped in a local optimum, regardless of the considerable amount of exploring solutions. In order to alleviate the risks of being trapped in such a local optimum, the adaptive procedure of alternatively allowing 2-opt exchange procedure of generating the neighborhood solutions is employed in MTS, even though it requires rather more computing time. That is, our MTS exploits 1-opt exchange procedure at the first predetermined *Adapiter* iterations, and then adopts 2-opt exchange procedure after the *Adapiter* iterations to escape the local optimum.

At first, in order to determine the proper value of *Adapiter* as the flexible strategy of generating the neighborhood solutions of our MTS algorithm, we performed a sensitivity analysis for stopping criterion of TS. We varied the value of *Stopiter* from 100, 500 to 1000.

Table 2: A comparison of TS and MTS

(a) s=30

	TS(100)	TS(500)	TS(1000)	MTS
1	0.908632	0.908632	0.908632	0.908632
2	0.913549	0.917532	0.917532	0.917532
3	0.923191	0.924963	0.926519	0.926519
4	0.931247	0.931298	0.933629	*0.935594
5	0.938332	0.938856	0.938856	0.938856

6	0.943398	0.943398	0.944984	0.944984
7	0.94876	0.949251	0.949124	*0.949672
8	0.954468	0.954468	0.954468	0.954648
9	0.958789	0.959187	0.960928	0.960928
10	0.963613	0.963613	0.963613	0.963613
T(sec.)	14.7	48.7	90.3	117.3

(b) s=40

	TS(100)	TS(500)	TS(1000)	MTS
1	0.846467	0.877540	0.880071	0.880071
2	0.890079	0.890079	0.890079	*0.890789
3	0.899636	0.901050	0.903218	0.902824
4	0.909391	0.912833	0.912597	*0.915061
5	0.918731	0.919243	0.919105	*0.920186
6	0.926244	0.924689	0.926244	0.926244
7	0.932005	0.931954	0.932177	0.932068
8	0.937709	0.939276	0.939276	*0.940855
9	0.945833	0.946006	0.948247	0.948247
10	0.951250	0.951781	0.951747	*0.951781
T(sec.)	32.4	95.3	164.2	205.9

(c) s=50

	TS(100)	TS(500)	TS(1000)	MTS
1	0.850486	0.850379	0.850598	*0.852407
2	0.860910	0.862835	0.864261	*0.866084
3	0.875717	0.878649	0.878728	*0.880030
4	0.884350	0.889970	0.894979	0.893736
5	0.899826	0.899542	0.899826	0.899826
6	0.907803	0.907794	0.907418	*0.907900
7	0.915749	0.915705	0.914810	*0.917033
8	0.924259	0.925744	0.924412	*0.925879
9	0.932270	0.933408	0.934261	*0.935732
10	0.939978	0.940047	0.940013	*0.940048
T(sec.)	58.4	150.6	313.2	327.5

* : MTS obtains the better solutions than TS

We compared their performance by extending the size of subsystem from 30 up to 50. From our computational experiments, we noticed that the compromised number of *Adapiter* (*i.e.*, the end phase of the 1-opt exchange) was 500. As a result, MTS employs the 1-opt exchange at the first 500 iterations without improving $R_s(x)$, and then adopts the 2-opt exchange after the 500 iterations without improving $R_s(x)$. The computational results of sensitivity analysis for stopping criterion are shown in **Table 2**. Among TS(100), TS(500), and TS(1000), the TS(1000) obtains the best results in terms of solution quality. However, it requires the considerable amount of computing time. The performance of our MTS is also shown in the last column of **Table 2**. From **Table 2**, we notice that our MTS obtains the better solution in 2, 5 and 8 cases for s=30, 40 and 50, respectively than the TS(1000). From the computational results, we notice that the proposed method obtains the better results than the TS, as the size of the problem increases from 30 to 50, even though MTS requires rather more computing time than the TS.

6. Conclusions

Solutions for RAP of complex systems have been suggested by several authors (Ha & Kuo [10]). All authors have dealt with only the case of homogeneous components. In this paper, we suggested a RAP of complex systems with heterogeneous components and obtained its optimal solution by the TS algorithm which is the best in the literature. In addition, in order to enhance TS in terms of the quality of the solution, we developed a MTS to employ an adaptive procedure (1-opt or 2-opt exchange) combined with the TS. Computational results showed that the proposed method obtained the better results as the size of the problem increases, even though it requires rather more computing time.

To improve the quality of the solution, the development of more powerful metaheuristics for the RAPCH should be performed in future research. With some adjustment of the parameters of the proposed method, it can be applied to the optimal reliability designs of complex systems of ships.

References

[1] D. E. Fyffe, W. W. Hines, and N. K. Lee, "System reliability allocation and a computational algorithm", IEEE Transactions on Reliability, vol. 17, pp. 64-69, 1968.

[2] Y. Nakagawa and S. Miyazaki, "Surrogate constraints algorithm for reliability optimiza- tion problems with two constraints", IEEE Transactions on Reliability, vol. 30, pp. 175-180, 1981.

[3] D. E. Coit and J. Liu, "System reliability optimization with k-out-of-n subsystems", International Journal of Reliability, Quality and Safety Engineering, vol. 7, pp. 129-142, 2000.

[4] D. W. Coit and A. E. Smith, "Reliability optimization of series-parallel systems using a genetic algorithm", IEEE Transactions on Reliability, vol. 45, pp. 254-260, 1996.

[5] S. Kulturel-Konak, B. A. Norman, D. W. Coit, and A. E. Smith, "Exploiting tabu search memory in constrained problems", INFOR- MS Journal on Computing, vol. 16, pp. 241-254, 2004 .

[6] Y. C. Liang and A. E. Smith, "An ant colony optimization algorithm for the reliability allocation problem (RAP)", IEEE Transactions on Reliability, vol. 53, pp. 417-423, 2004.

[7] Y. C. Liang and Y. C. Chen, "Redundancy allocation of series-parallel systems using a variable neighborhood search algorithm", Reliability Engineering and System Safety, vol. 92, pp. 323-331, 2007.

[8] N. Nahas, M. Nourelfath, and D. Ait-Kadi, "Coupling ant colony and the degraded ceiling algorithm for the redundancy alloca- tion problem of series-parallel systems", Reliability Engineering and System Safety, vol. 92, pp. 211-222, 2007.

[9] M. Ouzineb, M. Nourelfath and M. Gendreau, "An efficient heuristic for reliability design optimization problems", Computers & Operations Research, vol. 37, pp. 223-235, 2010.

[10] C. Ha and W. Kuo, "Initial allocation compensation algorithm for redundancy allocation: The scanning heuristic", IIE Transactions, vol. 40, pp. 678-689, 2008.

[11] M. S. Chern, "On the computational complexity of reliability redundancy allocation in a series system", Operations Research Letters, vol. 11, pp. 309-315, 1992.

[12] F. Glover, "Future Paths for Integer Programming and Links to Artificial Intelligence", Computers and Operations Research, vol. 13, pp. 533-549, 1986.

[13] J. Renaud, F. F. Boctor, and J. Ouenniche, "A heuristic for the pick up and delivery traveling salesman problem", Computers and Operations Research, vol. 27, pp. 905-916, 2000.