以免疫演算法為基礎的二階段法於可靠度最佳化問題之應用

莊淑惠
國立虎尾科技大學工業工程與管理所研究生
e-mail：s91114162@hotmail.com

謝益智
國立虎尾科技大學工業工程與管理所教授
e-mail：yhsieh@nfu.edu.tw

摘要

可靠度問題在學術研究上已有數十年的文獻記載，通常要提高整體系統可靠度主要有兩種方法，一種是提升系統零件本身可靠度，另一種是增加系統零件的備份數量。本研究主要探討混合整數規劃的可靠度問題，即在不違反成本、重量、空間等非線性限制下，要同時決定零件備份數量(整數變數)及該零件可靠度(實數變數)，以使整個系統可靠度為最大化。在本研究中，提出以免疫演算法為基礎的二階段法來解決此混合整數規劃的問題，即在第一階段中以免疫演算法來求解此可靠度問題的零件備份數量及該零件可靠度，於第二階段中固定第一階段得出之零件備份數量，提出一個修正法來微調改善零件可靠度。根據數值結果顯示此二階段法明顯優於其他啟發演算法，均可獲得優於目前文獻最佳解。

關鍵詞：可靠度設計、免疫演算法、最佳化

Abstract

The problem of reliability on the academic research has recorded for a few decades. In generally, there are two ways to raise the whole system reliability. One is to improve its own component reliability of the system; The other is to increase the number of redundancy components of the system. This research mainly probes into Nonlinearly mixed-integer reliability design problems which don't violate nonlinear constrains, for example costs, weight and space, etc. The number of redundancy components and the corresponding component reliability in each subsystem are to be decided simultaneously so as to maximize the reliability of system. In this research, we propose the immune performs two stage laws based on algorithms to solve the problem that this mixes integer planning. In the first stage, we get the solution of the number of redundancy components and the corresponding component reliability of the reliability problem in immune algorithm. Anchoring the numbers of redundancy components in the first stage, and offered a correctional formula to fix the corresponding component reliability during the second stage, As reported, solutions obtained by the proposed approach are as well as or better than the previously best-known solutions.

Keywords: Reliability design; Immune algorithm; Optimization

1. 前言

可靠度系統最佳化是實務中非常重要的問題，在近來數十年中的文獻已有各式各樣的可靠度系統相關研究，以往大多數的設計者致力於改進製造系統或產品組成部分的零件可靠度，以使其系統或產品可靠度提高，以便在市場上更具有競爭力。一般而言，設計者常使用兩種主要的方式來獲得較高的系統可靠度，第一種方式是直接提升系統零件的可靠度，第二種方式是增加零件的備份數量[11]。若按照第一種方式，因技術的因素，系統可靠度通常只能提升到某種程度，若按照第二種方式，即增加系統零件的備份數量，然而系統的重量、體積、成本等亦將增加。除了上述兩種方法之外，同時提升系統零件可靠度與零件備份數是提升整體可靠度系統的另一種可行方式[6,11]，然而須注意的是上述之最佳化問題均欲在有限的資源下，以使系統可靠度為最佳。

一般而言，可靠度設計問題包括整數問題[10]和混合整數兩種可靠度問題，其中：(i)整數可靠度設計問題假設各零件可靠度為已知，而決策整數為零件備份數，(ii)混合整數

的可靠度設計問題假設系統中各零件可靠度、零件備份數均為決策整數。

在過去的相關文獻中，較少研究需要同時決定零件備份與零件可靠度的混合整數問題，故在本研究中我們探討此混合整數的可靠度設計問題。我們將提出一個以免疫演算法為基礎的二階段法來解決此混合整數規劃的可靠度設計問題，即在第一階段中提出一個免疫演算法來求解此可靠度設計問題的零件備份數量(為整數變數)及該零件可靠度(為實數變數)，於第二階段中我們固定第一階段得出之零件備份數量(為整數變數)，提出一個修正法來微調改善零件可靠度(為實數變數)。實驗結果顯示此法所求出的解明顯優於目前文獻之最佳解。

2. 問題描述

在本研究中，我們探討四個具非線性限制條件之混合整數可靠度設計問題，包括：串聯系統、串並聯系統、複雜(橋)系統[6,8,16]以及氣體渦輪機保護系統[4,17]。

此四個混合整數之可靠度設計問題具有多個非線性限制條件，其一般式簡要說明如下:

$$\text{Max} \quad R_s = f(\mathrm{r},\mathrm{n})$$
$$\text{Subject to } \mathrm{g}(\mathrm{r},\mathrm{n}) \le l$$
$$0 \le r_i \le 1, \quad n_i \in \text{positive integer}, \quad 1 \le i \le m$$

上式中之 r_i 和 n_i 分別為第 i 個子系統中的零件可靠度及零件數，而 g (r , n) 為限制函數，l 為資源限制，m 為子系統的數量, 其詳細的符號說明如表 1 所示：

表 1 符號說明

m	子系統的個數
n_i	子系統的零件數
n	整體系統內的備份數量
r_i	子系統內各個零件可靠度
r	整體系統內零件可靠度
q_i	第系統內各個零件的失敗機率
$R_i(n_i)$	子系統可靠度
R_s	整體系統可靠度
V	整體系統內數量的限制
C	整體系統內成本的限制
W	整體系統內重量的限制

此混合整數之可靠度設計問題的最主要目標為求出每個子系統內的零件數量和零件可靠度，使整體系統之可靠度最大化，過去文獻中有許多學者探討此可靠度設計問題，例如：學者 Xu et al.[16] and Hikita et al.[6]利用啟發式演算法來求解可靠度備份數量的問題，Kuo et al.[13]使用 Lagrange multiplier 與分枝界限法來求解五個子系統的串聯問題，Yokota et al.[17] and Hsieh et al.[8] 使用基因演算法來解決混合整數最佳化問題，他們指出基因演算法能有效的解決此最佳化問題，除此之外，Chen and You [1]使用免疫演算法解決備份數量分配問題。

3.方法介紹

本研究欲以二階段方法來解決所提之問題，該法包含第一階段的免疫演算法和第二階段的修正方法。免疫演算法是藉由人體免疫系統的特性，利用抗體及抗原之間的複製、交配及突變等來求解最佳化問題仿生物免疫系統的原理來解決複雜問題，免疫系統可識別外在入侵的抗原與自身的抗原，利用抗體所產生的免疫反應來抵抗外在的入侵，其最後會藉由隨機產生，淘汰選擇之抗體，將優良的抗體存在記憶區中，以做最有效的運用。入侵的抗原代表找尋最佳化問題的系統，相當於目標函數，而抗體相當於在系統中尋求最佳可能性的解，當抗體消滅抗原就如同解決複雜問題。免疫演算法具有特性為:一、自體辨識性，能辨別自身或是外來的抗原;二、專一性:可區別不同的抗原;三、雜異性:辨認抗原的不同構造以產生不同的反應;四記憶性:第二次感染相同抗原之病原體很快即可產生免疫反應;五、無性繁殖:無性複製逐代演化最佳解，可改善遺傳演算法較易收斂於局部解的缺點。本研究中第一階段免疫演算法之流程架構如圖 1，說明如下：

步驟一：隨機產生初始族群。

步驟二：評量族群中的每一個個體。

步驟三：由步驟二之評量結果，將族群中最好之 m 個個體選擇出來。

步驟四：接著將m個個體複製成另一新族群，而個體複製數目又會隨評量的高低而定，當評量結果越高，被複製的數目越多，反之則越少。

步驟五：將步驟四所產生複製後新族群進行基

因運算，包含基因交換、突變等交換程序[2]，進而產生新族群。

步驟六：將步驟五所產生的新族群個體重新評量，若該族群內有個體能改善記憶區內原有族群的個體，則把該新個體取代記憶區內的舊有個體。並且將部分結構相似性過高的個體剔除，以維持記憶區內個體的多樣性，進而能消滅各種不同的外來病毒。

步驟七：Check停止條件(例如最大的演化代數)是否被達成。

步驟八：結束。由記憶區中輸出較佳解。

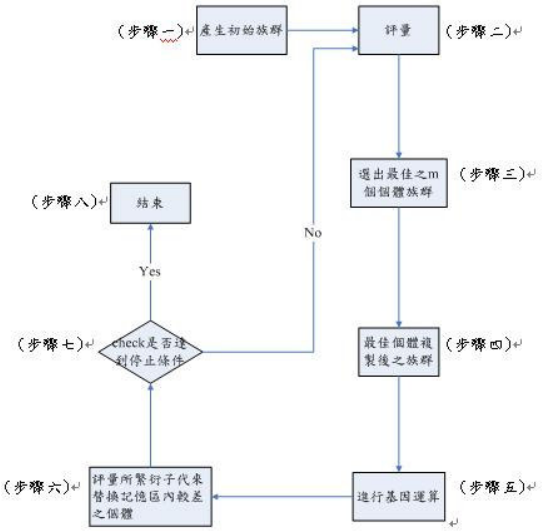

圖 1 免疫演算法流程

第二階段的修正方法為固定第一階段得出之零件備份數量(為整數變數)，以免疫演算法得出來的解當做修正方法的初始解，來修正微調改善零件可靠度(為實數變數)，進而修正改善為最佳解或近似最佳解。本研究第二階段之修正法流程架構如圖2，說明如下：

步驟一：參數設定，免疫演算法所求出的解。

步驟二：預先給定k_1、k_2的初始值。

步驟三：假設取第一個零件可靠度加上k_1，第二個零件可靠度減掉k_2，以二分法求出k_2於(0,1)之最少可行數值，當k_2值的變動絕對值<=10^{-9}停止找尋。

步驟四：判斷系統可靠度是否有改善，若有改善則回到步驟五，反之則直接跳到步驟六。

步驟五：記錄目前較佳解。

步驟六：判斷 k_1 值乘上 0.99 的次數是否大於 100 次。

步驟七：取步驟五所記錄下來的較佳解資料來做再次修正。

步驟八：判斷系統可靠度的變動絕對值是否小於 10^{-9}，若否，則到步驟九，反之則到步驟十。

步驟九：將原本的 r 更新為新組解之 r。

步驟十：得出修正後之較佳解或近似最佳解。

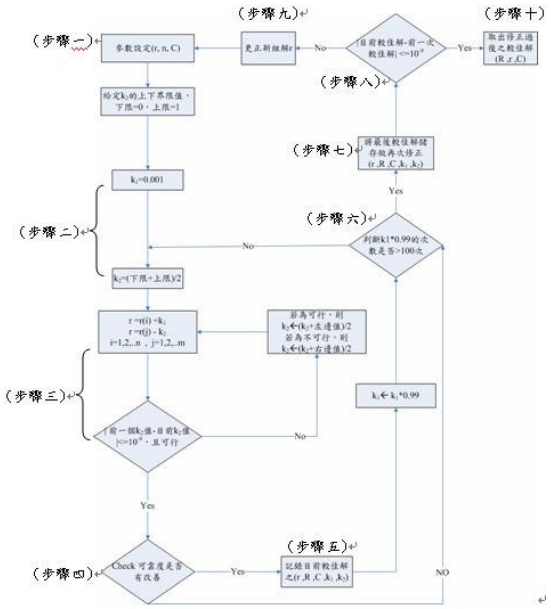

圖 2 第二階段修正法流程

4.測試問題、數值結果與討論

本研究以MATLAB撰寫程式，其中免疫演算法的參數設定分為，記憶區大小90，突變率0.01，交配率0.92，世代數150，再以提出之二階段法來測試四個具非線性限制條件之混合整數可靠度設計問題，包括：串聯系統、串並聯系統、複雜(橋)系統[6,8,16]以及氣體渦輪機保護系統[4,17]，我們將與其他學者使用之啟發式演算法求出的解做比較。下列為此四個測試問題。

問題 1：串聯系統 [6]，其數學規畫模式如下。

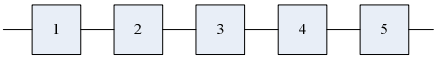

圖 3 串聯系統

$$\text{Max} \quad f(\mathrm{r},\mathrm{n})=\prod_{i=1}^{m} R_i(n_i)$$

$$\text{Subject to } g_1(\mathrm{r},\mathrm{n})=\sum_{i=1}^{m} w_i v_i^2 n_i^2 \le V$$

$$g_2(\mathrm{r},\mathrm{n})=\sum_{i=1}^{m} \alpha_i(-1000/\ln r_i)^{\beta_i}(n_i/4)) \le C$$

$$g_3(\mathrm{r},\mathrm{n})=\sum_{i=1}^{m} w_i n_i \exp(n_i/4) \le W$$

$0 \le r_i \le 1, \quad n_i \in \text{positive integer}, \quad 1 \le i \le m$

$C(r_i) = \alpha_i[-T / \ln(r_i)]^{\beta_i}$

問題 2：串並聯系統 [6]，其數學規畫模式如下。

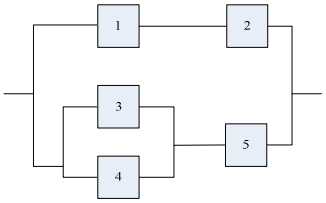

圖 4 串並聯系統

Max $f(\mathrm{r,n}) = 1-(1-R_1R_2)(1-(1-(1-R_3)(1-R_4))R_5)$

Subject to $g_1(\mathrm{r,n}) = \sum_{i=1}^{m} w_i v_i^2 n_i^2 \le V$

$g_2(\mathrm{r,n}) = \sum_{i=1}^{m} \alpha_i(-1000/\ln r_i)^{\beta_i}(n_i/4)) \le C$

$g_3(\mathrm{r,n}) = \sum_{i=1}^{m} w_i n_i \exp(n_i/4) \le W$

$0 \le r_i \le 1, \quad n_i \in \text{positive integer}, \quad 1 \le i \le m$

問題 3：複雜(橋)系統[6]，其數學規畫模式如下。

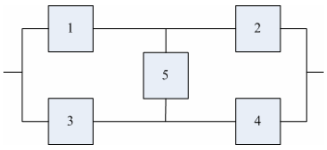

圖 5 複雜(橋)系統

Max $f(\mathrm{r,n}) = R_1R_2 + R_3R_4 + R_1R_4R_5 + R_2R_3R_5 - R_1R_2R_3R_4 - R_1R_2R_3R_5 - R_1R_2R_4R_5 - R_1R_3R_4R_5 - R_2R_3R_4R_5 + 2R_1R_2R_3R_4R_5$

Subject to $g_1(\mathrm{r,n}) = \sum_{i=1}^{m} w_i v_i^2 n_i^2 \le V$

$g_2(\mathrm{r,n}) = \sum_{i=1}^{m} \alpha_i(-1000/\ln r_i)^{\beta_i}(n_i/4)) \le C$

$g_3(\mathrm{r,n}) = \sum_{i=1}^{m} w_i n_i \exp(n_i/4) \le W$

$0 \le r_i \le 1, \quad n_i \in \text{positive integer}, \quad 1 \le i \le m$

問題 4：氣體渦輪機保護系統[4,17]，其數學規畫模式如下。

Max $f(\mathrm{r,n}) = \prod_{i=1}^{m}[1-(1-r_i)^{n_i}]$

Subject to $g_1(\mathrm{r,n}) = \sum_{i=1}^{m} v_i n_i^2 \le V$

$g_2(\mathrm{r,n}) = \sum_{i=1}^{m} C(r_i)\cdot[n_i + \exp(n_i/4)] \le C$

$g_3(\mathrm{r,n}) = \sum_{i=1}^{m} w_i n_i \exp(n_i/4) \le W$

$1.0 \le n_i \le 10, \quad n_i \in \text{positive integer}$

$0.5 \le r_i \le 1\text{-}10^{-6}, \quad r_j \in \text{real number}$

表 2 問題 1、3 所需之參數設定[6]

i	$10^5\alpha_i$	β_i	$w_i v_i^2$	W_i	***V***	***C***	***W***
1	2.33	1.5	1	7	110	175	200
2	1.45	1.5	2	8			
3	0.541	1.5	3	8			
4	8.05	1.5	4	6			
5	1.95	1.5	2	9			

表 3 問題 2 所需之參數設定[6]

i	$10^5\alpha_i$	β_i	$w_i v_i^2$	W_i	***V***	***C***	***W***
1	2.5	1.5	2	3.5	180	175	100
2	1.45	1.5	4	4			
3	0.541	1.5	5	4			
4	0.541	1.5	8	3.5			
5	2.1	1.5	4	4.5			

表 4 問題 4 所需之參數設定[6]

i	$10^5\alpha$	β_i	$w_i v_i^2$	W_i	***V***	***C***	***W***	***T***
1	1	1.5	1	6	250	400	500	1000h
2	2.3	1.5	2	6				
3	0.3	1.5	3	8				
4	2.3	1.5	2	7				

從附錄 1 中的表 5 到表 8 的數值結果可看出：

1. 本研究所使用的二階段法於串聯問題有較佳表現。表 5 包含學者 Hikita et al.[6]、Kuo et al.[12]、Xu et al.[16]、Hsieh et al.[8]、Gopal et al.[5]、Hikita et al.[7]、Cichocki et al.[3]、Jacobson et al.[9]、Prasad et al.[15]、Mitsuo et al.[14]、Chen [2]等用啟發式演算法求解串聯問題之結果，其求出的解都非常地接近，在過去研究中又以 Chen [2]、 Prasad et al.[15]求出的解較好，而本研究所使用的二階段法求出的解 0.93168230 顯然比 Chen [2]、Prasad et al.[15]所求出的解 0.931678 要好。
2. 本研究所使用的二階段法於串並聯問題有較佳表現。表 6 包含學者 Hikita et al.[6]、Hsieh et al.[8]、Chen[2]等用啟發式演算法求解，在過去研究中又以 Chen [2]求出的解較好，而本研究所使用的二階段法求出的解 0.9999766487 又比 Chen [2]所求出的解 0.99997658 要好。
3. 本研究所使用的二階段法於橋系統問題有較佳表現。表 7 包含學者 Hikita et al.[6]、Hsieh et al.[8]、Chen[2]等用啟發

式演算法求解，在過去研究中又以 Chen [2]求出的解較好，本研究所使用的二階段法求出的解 0.9998893502 又比 Chen [2]所求出的解 0.99988921 要好。

4. 本研究所使用的二階段法於氣體渦輪機保護系統問題有較佳表現。表 8 包含學者 Dhingra[4]、Yokota et al.[17]、Chen [2]等用啟發式演算法求解，在過去研究中又以 Chen [2]求出的解較好，本研究所使用的二階段法求出的解 0.99994615 顯然地比 Chen [2]所求出的解 0.999942 要好。

為了要測量改進的幅度，我們使用 MPI 來做為衡量的指標，其公式為：

$$MPI(\%)= (R_{s\_myself} - R_{s\_other})/(1-R_{s\_other})$$

可從表 5 到表 8 中得知串聯系統、串並聯系統、橋系統、 氣體渦輪機保護系統以 Chen [2]求得的解為比較之 MPI 分別為 0.0063%、0.2933%、0.1265%、7.1552%，特別注意到二階段法於問題 4：「氣體渦輪機保護系統」的改善幅度明顯優於其他系統(見表 8)。

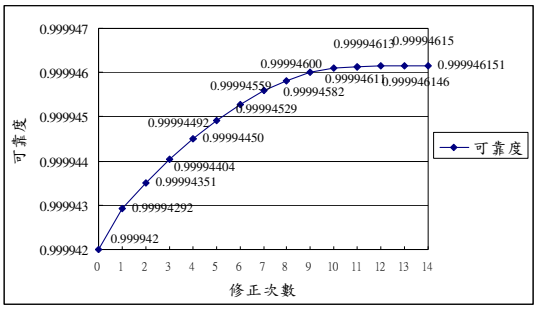

圖 6 氣體渦輪機保護系統修正趨勢變動圖

圖 6 為使用二階法於氣體渦輪機保護系統的系統可靠度修正趨勢變動圖，由圖 6 可以得知一開始的改善較為快速，直到修正次數到第 10 之後，其改善就漸漸趨於平緩，而於第 14 次修正結束並求得最佳解或近似最佳解。

5.結論

在本研究中，我們提出一個以免疫演算法為基礎的二階段法來解決此混合整數規劃的可靠度設計問題，即在第一階段中提出一個免疫演算法來求解此可靠度設計問題的零件備份數量(為整數變數)及該零件可靠度(為實數變數)，於第二階段中我們固定第一階段得出之零件備份數量(為整數變數)，提出一個修正法來微調改善零件可靠度(為實數變數)。我們以四個混合整數規劃的可靠度問題(包括串聯系統、串並聯系統、複雜(橋)系統以及氣體渦輪機保護系統)來做測試此新提出之二階段法，並與過去學者之演算法求出的解做比較，數值結果顯示此新提出之二階段法明顯優於其他啟發演算法，均可獲得優於目前文獻最佳解之結果。

參考文獻

[1] Chen, T.C., You, P.S., “Immune algorithm based approach for redundant reliability problems,” ***Computers in Industry***, Vol. 56, pp. 195-205, 2005.

[2] Chen, T.C., “IAs based approach for reliability redundancy allocation problems,” ***Applied Mathematics and Computation***, Vol. 182, pp. 1556-1567, 2006.

[3] Cichocki, A., Unbehauen, R., “Neural networks optimization and signal processing,,” ***New York:Wiley***,1994.

[4] Dhingra, A.K., “Optimal apportionment of reliability & redundancy in series systems under multiple objectives,” ***IEEE Transactions on Reliability***, Vol. 41, No. 4, pp. 576-582, 1992.

[5] Gopal, K., Aggarwal, K., Gupta J.S., “An improved algorithm for reliability optimization,” ***IEEE Trans Reliab***, Vol. 27, No. 5, pp. 325-328, 1978.

[6] Hikita, M., Nakagawa, Y., Harihisa, H., “Reliability optimization of systems by a surrogate constraints algorithm,” ***IEEE Transactions on Reliability***, Vol. 41, No. 3, pp. 473-480, 1992.

[7] Hikita, M.Y., Nakagawa, Y., Nakashima, K., Narihisa, H., “Reliability optimization of system by a surrogate constraints algorithm,” ***IEEE Trans Reliab***, Vol. 7, No. 5, pp. 325-328, 1978.

[8] Hsieh, Y.C., Chen, T.C., Bricker, D.L., “Genetic algorithms for reliability design problems,” ***Microelectronic Reliability***, Vol. 38, pp. 1599-1605,1998.

[9] Jacobson, D.W., Arora, S.R., “Simultaneous allocation of reliability and redundancy using simplex search,” ***Proceedings of annual reliability and maintainability symposium***, pp. 243-250, 1996.

[10] Kevin, Y.K. NG., Sancho, N.G.F., “A hybrid dynamic programming/depth-first search algorithm, with an application to redundancy allocation,” ***IIE Transactions***, Vol. 33, pp. 1047-1058, 2001.

[11] Kuo, W., Prasad, V.R., “An annotated overview of system-reliability optimization,”

IEEE Transaction on Reliability, Vol. 49, No. 2, pp. 176-187, 2000.

[12] Kuo, w., Hwang, C.L., Tillman, F.A., "A note on heuristic methods in optimal system reliability," ***IEEE Transactions on Reliability***, Vol. 27, pp. 320-324,1978.

[13] Kuo, W., Lin, H., Xu, A., Zhang, W., "Reliability optimization with the Lagrange multiplier and branch and bound technique," ***IEEE Transaction on Reliability***, Vol. 36, 624-630, 1987.

[14] Mitsuo, G., Young, S.Y., "Soft computing approach for reliability optimization: State-of-the-art survey," ***Reliability Engineering and System Safety***, Vol. 91, pp.1008-1026, 2006.

[15] Prasad, V.R., Kuo, W., Reliability optimization of coherent systems, ***IEEE Trans Reliab***, Vol. 49, No. 3, pp. 323-330, 2000.

[16] Xu, Z., Kuo, W., Lin, H.H., "Optimization limits in improving system reliability," ***IEEE Transactions on Reliability***, Vol. 39, No. 1, pp. 51-60, 1990.

[17] Yokota, T., Gen, M., Li, Y.X., "Genetic algorithm for nonlinear mixed-integer programming problems and its application," ***Computers and Industrial Engineering***, Vol. 30, No. 4, pp. 905-91, 1996

附錄 1

表 5 二階段法與其他啟發式法於串聯系統之數值結果比較

	Hikita et al.[6]	Kuo et al.[12]	Xu et al.[16]	Hsieh et al.[8]	Gopal et al.[5]	Hikita et al.[7]	Cichocki et al.[3]
n	(3,2,2,3,3)	(3,3,2,3,3)	(3,2,2,3,3)	(3,2,2,3,3)	(3,2,2,3,3)	(3,2,2,3,3)	(3,2,2,3,3)
	0.777143	0.7796	0.77939	0.779427	0.8	0.774887	0.759193
	0.867514	0.80065	0.87183	0.869482	0.8625	0.870065	0.853255
r	0.896696	0.90227	0.90288	0.902674	0.90156	0.898549	0.909354
	0.717739	0.71044	0.71139	0.714038	0.7	0.716524	0.732406
	0.793889	0.85947	0.78779	0.786896	0.8	0.791368	0.764129
Rs	0.931363	0.92975	0.931677	0.931578	0.930289	0.931451	0.926224
MPI(%)	0.4652%	2.7506%	0.0077%	0.1524%	1.9987%	0.3374%	7.3985%
slacks of(g1~g3)	27	27	27	27	27	27	27
	0	0.00001	0.013773	0.121454	0.026500	0.108244	3.584388
	7.518918	10.572480	7.518918	7.518918	7.518918	7.518918	7.518918

$MPI(\%)=(R_{s\_myself}-R_{s\_other})/(1-R_{s\_other})$

Jacobson et al.[9]	Prasad et al.[15]	Mitsuo et al.[14]	Chen[2]	二階段修正
3,3,2,3,2)	(3,2,2,3,3)	(3,2,2,3,3)	(3,2,2,3,3)	(3,2,2,3,3)
0.774559	0.77978	0.780874	0.779266	0.7792660000
0.834191	0.87232	0.871292	0.872513	0.8718182501
0.873006	0.90245	0.902316	0.902634	0.9030000323
0.72007	0.71081	0.711945	0.710648	0.7113657305
0.848304	0.78816	0.786995	0.788406	0.7878828947
0.922909	0.931678	0.931676	0.931678	0.93168230
11.3804%	0.0063%	0.0092%	0.0063%	
27	27	27	27	27
0.001000	0.008200	0.003352	0.001559	0.0000002
10.572000	7.518918	7.518918	7.518918	7.518918

表 6 二階段法與其他啟發式法於串並聯系統之數值結果比較

	Hikita et al.[6]	Hsieh et al.[8]	Chen[2]	二階段修正法
n	(3,3,1,2,3)	(2,2,2,2,4)	(2,2,2,2,4)	(2,2,2,2,4)
	0.838193	0.785452	0.812485	0.8191144490
	0.855065	0.842998	0.843155	0.8448997192
r	0.878859	0.885333	0.897385	0.8956612970
	0.911402	0.917958	0.894516	0.8955060000
	0.850355	0.870318	0.87059	0.8685863470
Rs	0.99996875	0.99997418	0.99997658	0.9999766487
MPI(%)	25.2758%	9.5612%	0.2933%	
slacks of(g1~g3)	53	40	40	40
	0	1.19444	0.002627	0.00000002
	7.110849	1.609289	1.609289	1.609289

$MPI(\%)=(R_{s\_myself}-R_{s\_other})/(1-R_{s\_other})$

表 7 二階段法與其他啟發式法於橋系統之數值結果比較

	Hikita et al.[6]	Hsieh et al.[8]	Chen[2]	二階段修正法
n	(3,3,2,3,2)	(3,3,3,3,1)	(3,3,3,3,1)	(3,3,3,3,1)
	0.814483	0.81409	0.812485	0.8164937013
	0.821383	0.864614	0.867661	0.8686510000
r	0.896151	0.890291	0.861221	0.8588856825
	0.713091	0.70119	0.713852	0.7104589571
	0.814091	0.734731	0.756699	0.7536865004
Rs	0.99978937	0.99987916	0.99988921	0.9998893502
MPI(%)	47.4672%	8.4328%	0.1265%	
slacks of(g1~g3)	18	18	18	18
	1.854075	0.376347	0.001494	0.00000001
	4.264770	4.264770	4.264770	4.264770

$MPI(\%)=(R_{s\_myself}-R_{s\_other})/(1-R_{s\_other})$

表 8 二階段法與其他啟發式法在求解氣體渦輪機保護系統之數值結果比較

	Dhingra[4]	Yokota et al.[17]	Chen [2]	二階段修正法
n	(6,6,3,5)	(3,6,3,5)	(5,5,5,5)	(5,5,5,5)
	0.81604	0.965593	0.9038	0.8990533897
	0.80309	0.760592	0.874992	0.8856240646
r	0.98364	0.972646	0.919898	0.9162158164
	0.80373	0.80466	0.890609	0.8855166063
Rs	0.99961	0.999468	0.999942	0.9999461500
MPI(%)	86.1923%	89.8778%	7.1552%	
slacks of(g1~g3)	65	92	50	50
	0.064	-70.733576	0.002152	0.00000100
	4.348	127.583189	28.803701	28.803701

$MPI(\%)=(R_{s\_myself}-R_{s\_other})/(1-R_{s\_other})$