콤플렉스 시스템의 신뢰도 최적화를 위한 발견적 합성해법의 개발

김 재 환*

A Hybrid-Heuristic for Reliability Optimization in Complex Systems

Jae-Hwan Kim*

〈목 차〉

Abstract

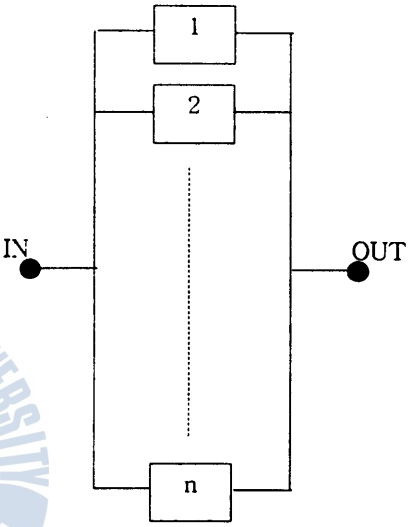
This study is concerned with developing a hybrid heuristic algorithm for solving the redundancy optimization problem which is very important in system safety. This study develops a HH(Hybrid Heuristic) method combined with two strategies to alleviate the risks of being trapped at a local optimum. One of them is to construct the populations of the initial solutions randomly. The other is the additional search with SA(Simulated Annealing) method in final step. Computational results indicate that HH performs consistently better than the KY method proposed in Kim[8]. Therefore, the proposed HH is believed to an attractive to other heuristic methods.

+ 본 연구는 1997년도 전반기 한국과학재단의 해외 post-doc. 연구지원에 의해 수행되었음.
* 한국해양대학교 응용수학과

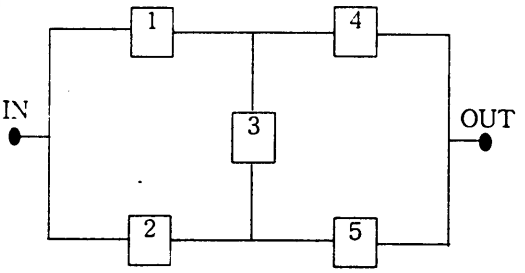
1. 서 론

본 논문에서 다루고자 하는 complex 시스템의 신뢰도 최적화 문제는 선박, 비행기, 통신시스템 등의 신뢰도를 높이기 위해 자원이나 기술적 요인에 따른 다양한 제약하에서 하부시스템(subsystem)의 중복(redundancy)설계를 최적화하는 것이다. 이 문제는 만족스러운 서비스 제공 뿐만 아니라 인간의 생명과 안전과도 밀접한 관계가 있을만큼 중요한 문제인데도 불구하고 기존의 연구를 살펴보면, 직렬구조(<그림1>)나 병렬구조(<그림2>)의 시스템[15]에 대한 연구에 비해서, complex 시스템(<그림3>)에 대한 연구는 아직까지 많이 수행되지 않은 실정이다[10]. 그 이유는 이 문제가 NP-Complete인 비선형 정수계획 문제에 속해서 다루기 힘들기 때문이다([3], [9]). 비선형 정수계획의 해법은 크게 정확한 방법(exact method)과 발견적 해법(heuristic method)으로 구성되는데, 정확한 방법([11], [14], [16])은 global 최적해를 구할 수 있는 장점이 있지만 중복설계 문제를 포함한 모든 비선형 정수계획문제에 대해 개발된 것은 아니며, 개발된 방법조차도 계산상의 복잡도(computational complexity)에 의한 계산노력이 너무 많이 소요되기 때문에 실용화하기 어려운 단점이 있다[4]. 반면에, 발견적 방법([5], [7], [9], [13])은 정확한 방법에 비해 계산시간은 적게 소요되지만, exchange 방법에 의하여 근접해 있는 영역만을 탐색하는 sequential improvement[2] 과정을 바탕으로 확장 및 보완된 것이어서 local 최적해에 도달할 수 있는 단점이 있다.

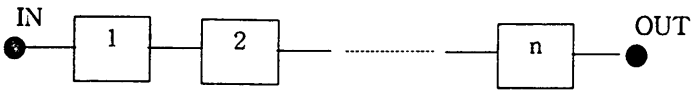
따라서, 이러한 위험을 줄이고자 여러가지 전략이 고려된 발견적 방법들이 개발되고 있는데, 예를 들면 pairwise exchange 방법을 확장하여 2-way 이상의 exchange([2], [12])를 수행하는 방법과 oscillating 절차[5] 등과 탐색 영역을 "feasible region" 뿐만 아니라 "infeasible region"까지 확장하는 전략을 가미한 방법들이 개발되었다([13],[8]). 최근에는 인공지능분야에서 SA (Simulated Annealing), TS(Tabu Search), GA(Genetic Algorithm)등이 유용한 대안으로 등장하고 있으며, manufacturing, job scheduling,



<그림 2> 병렬구조 시스템



<그림 3> Complex 시스템 구조



<그림 1> 직렬구조 시스템

facility location 문제 등을 비롯한 많은 최적화 문제에 적용되고 있다([1]. [4]. [6]).

본 논문에서는 국부최적해(local optimum)에 도달하는 위험을 줄이고, global 최적해에 좀 더 가까운 해를 탐색하기 위해 GA에서 "population"을 구성하는 것과 유사하게 초기해 집단을 랜덤하게 구성한 후, 각각의 초기해에 대해 먼저 Kim[8]에서 다른 알고리듬에 비해 성능면에서 우수하다고 평가된 KY방법을 적용하여 해를 개선시키고, 최종적으로 요즘 인공지능 분야에서 좋은 반응을 얻고 있는 SA방법을 한번 더 적용하여 탐색하는 효율적인 발견적 합성해법(Hybrid Heuristic Algorithm)을 개발하였다.

3장의 예제에 대해 분석한 결과, 이 초기해 집단과 SA방법을 합성시킨 방법이 KY 방법보다 더 좋은 해를 얻을 수 있었으며, 4장에서는 좀 더 큰 규모의 complex 시스템에 대해 본 논문에서 개발한 발견적 합성해법의 성능을 비교 분석하였다.

1.1 기호

n : 하부시스템의 수.

m : 제약식의 수.

x_i : i번째 하부시스템의 부품의 갯수.양의 정수, i=1,2..... n.

x : $(x_1, x_2, \cdots\cdots, x_n)$.

x^0 : 초기 가능해.

x^f : excursion 탐색을 통해 개선된 해.

x^* : 알고리즘 수행후에 발견된 최적해.

x^c : excursion 탐색 중 어느 시점까지의 최적해.

$g_{j\,i}(x_i)$: i번째 하부시스템에 소요되는 j번째 자원의 양.

b_j : j번째 자원의 최대 사용 가능량.

$b_j{}^c$: $b_j - \sum_{i=1}^{n} g_{j\,i}(x_i)$.

r_i : i번째 하부시스템의 부품의 신뢰도.

$R_i(x_i)$: x_i개의 부품을 중복설계 했을 때 i번째 하부시스템의 신뢰도.

$Q_i(x_i)$: x_i개의 부품을 중복설계 했을 때 i번째 하부시스템의 비신뢰도.

$R_s(x)$: 전체 시스템의 신뢰도.

$x(\pm i)$: i번째 하부시스템의 부품을 하나 증가시키거나(+) 감소시킴(-).

F-집합 : excursion 탐색이 실패했을 때 탐색과정에서 발생한 모든 해들의 집합.

E-집합 : excursion 탐색 동안에 발생한 해들의 집합.

BFR : 한정된 가능해의 영역 (bounded feasible region).

$BIFR$: 한정된 비가능해의 영역(bounded infeasible region).

$\triangle_j$: BFR 또는 $BIFR$의 경계를 결정해 주는 상수.

NTOT : 초기해의 집단의 총 갯수

NIT : 한단계에서의 SA 알고리듬의 iteration 의 총 갯수

T^o : SA 알고리듬의 초기상수

T^s : SA 알고리듬의 중단상수

1.2 가 정

1. 시스템과 하부시스템은 모두 coherent하다.
2. 시스템은 n개의 하부시스템으로 구성되고, 각각은 (1-out-of- x_i:G) 이다.
3. 모든 부품 상태들은 통계적으로 독립이다.
4. i번째 하부시스템에 소요되는 j번째 자원의 양은 x_i 의 증가함수이며, 각 하부시스템의 j번째 자원 소비량은 j번째 자원의 전체 소비량에 대해 가법성을 갖는다.
5. 전체 시스템 신뢰도 Rs 는 R_i이나 Q_i의

함수로 주어진다.

1.3 모형

본 연구에서 다루고자 하는 시스템의 안전을 위한 중복설계 문제는 다음과 같은 비선형 정수계획 모형이다.

$$(P) \qquad \text{maximize} \quad R_s(x)$$

subject to

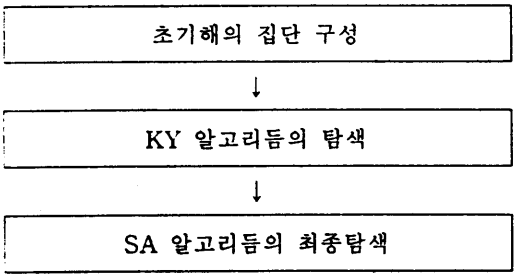
$$\sum_{i=1}^{n} g_{j\,i}(x_i) \le b_j, \quad j = 1, 2, \cdots\cdots, m$$

$$x_i \ge 1, \text{ 정수}, \quad i = 1, 2, \cdots\cdots, n.$$

2. 발견적 합성해법

2.1 발견적 합성해법의 개요

본 논문에서 개발한 발견적 합성해법인 HH (Hybrid Heuristic) 방법은 다음과 같은 3가지 단계로 구성된다.



<그림 4> HH의 구성

즉, HH는 Kim[8]에서 성능면에서 Shi[8]와 Kohda/Inoue[10]의 방법보다 우수하다고 알려진 KY 알고리듬에 두 가지 탐색 전략(초기해 집단 구성, SA 알고리듬의 최종탐색)을 합성시킨 발견적 합성해법이다.

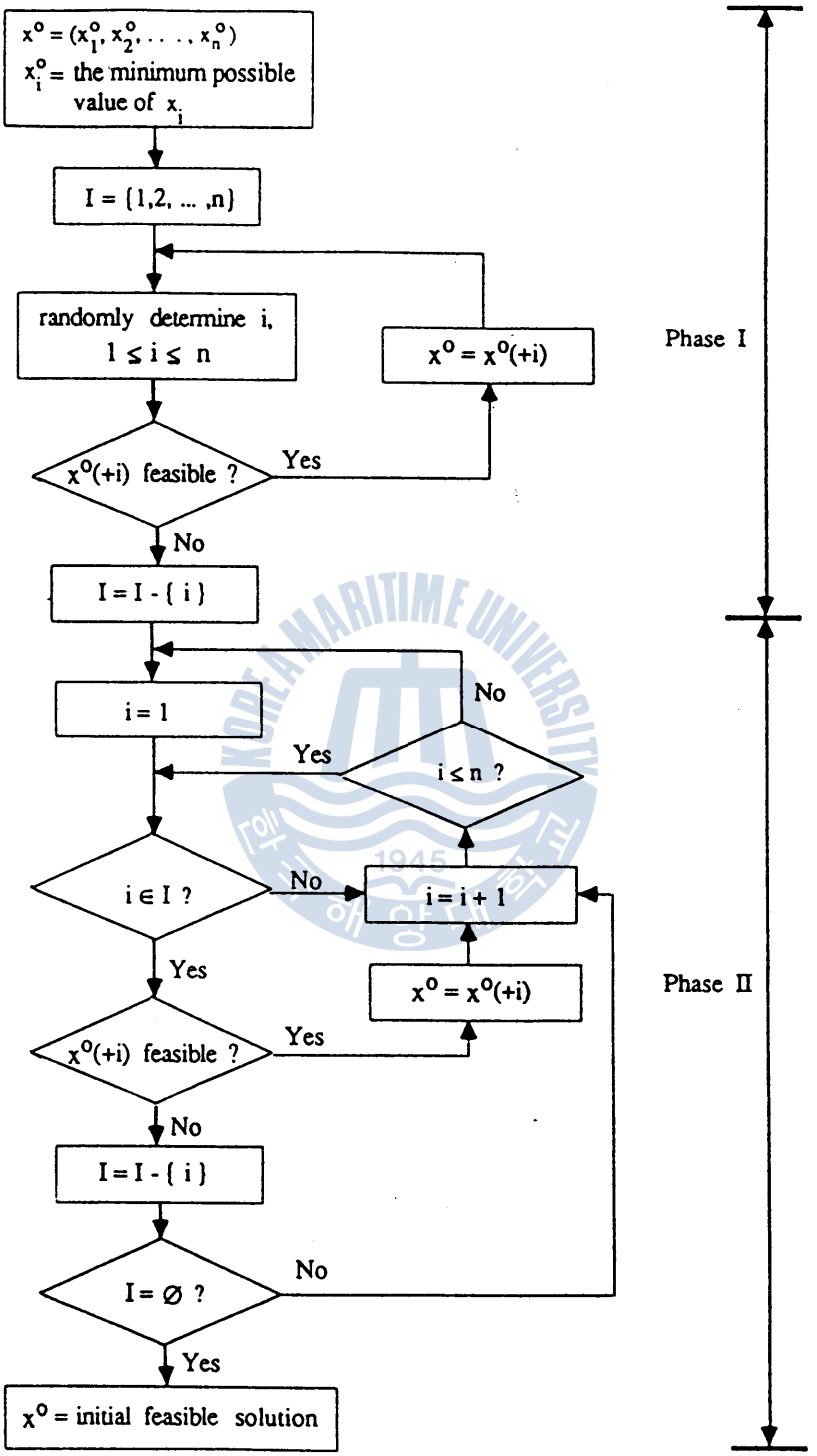
초기해의 집단 구성은 GA에서 "population"을 구성하는 것과 유사하게 N개의 초기해를 < 그림 5>의 two-phase 방법에 의해 랜덤하게 구성하여 초기해의 집단을 구한다. 1개가 아닌 N개의 초기해를 구성하는 목적은 국부최적해에 도달하는 위험을 줄이기 위함이다.

4장의 전산실험을 통해 이 초기해의 집단구성이 생성해의 질을 높이는데 주요한 역할을 하는 것을 관측할 수 있었다. 또한, SA알고리듬에 의한 최종 탐색을 통해 global 최적해에 가까운 해를 찾도록 하였고, 4장의 전산실험 결과, 해를 개선시키는데 주요한 역할을 하는 것을 관측할 수 있었다.

2.2 HH의 구체적인 단계

본 연구에서 개발한 HH의 단계는 다음과 같다.

0. $\triangle_j$ 를 결정한다. num=0.
1. 초기해 x^0 를 구성한다. $x = x^0$.
   num=num+1. num> NTOT 이면, 단계17로 간다.
2. $x^c = x$. $E = F = \emptyset$.
3. $x = x(+i)$.
   $|b_j{}^c| > \triangle_j$ 이면, 단계 7로 간다.
4. $E = E \cup x$.
   $x \in F$ 이면, $F = F \cup E$, $E = \emptyset$.
   단계 3으로 간다.
   $x \notin F$ 이면, $x = x(-i)$.
5. $x \notin FR$ 이면, 단계 4로 간다.
   $x \in FR$ 이면, $x^f = x$. x^f 를 개선시킨다.
6. $R_s(x^f) > R_s(x^c)$ 이면, $x = x^f$. 단계 2로 간다.



<그림 5> 초기해의 구성방법

$R_s(x^f) \leq R_s(x^c)$ 이면,

$x = x^c$. $F = F \cup E$, $E = \emptyset$.

단계 3으로 간다.

7. $x = x^c$, $R_s(x) = R_s(x^c)$

8. $N = 1$, $T = T^0$.

9. $i = 1$.

10. $j \neq i$ 인 [1, n]사이의 정수 j 를 랜덤하게 선택한다.

$x = x(+i)$. $x = x(-j)$.

$D = R_s(x) - R_s(x^c)$.

11. $D > 0$ 이면, 단계 12로 간다.

$D \leq 0$ 이면 (0, 1) 사이의 난수(random unmber) U 를 구한다.

$U < e^{D/T}$ 이면, 단계 12로 간다.

$U \geq e^{D/T}$ 이면, $i = i + 1$. 단계 13으로 간다.

12. $x^c = x$, $i = i + 1$.

13. $i > n$ 이면, 단계 14로 간다.

$i \leq n$ 이면, 단계 10으로 간다.

14. $N = N + 1$.

$N >$ NIT 이면, 단계 15로 간다.

$N \leq$ NIT 이면, 단계 9로 간다.

15. $T = T/10$.

$T < T^s$ 이면, 단계 16으로 간다.

$T \geq T^s$ 이면 단계 8로 간다.

16. $x^* = x^c$, $R_s(x^*) = R_s(x^c)$. 단계 1로 간다.

17. 중단.

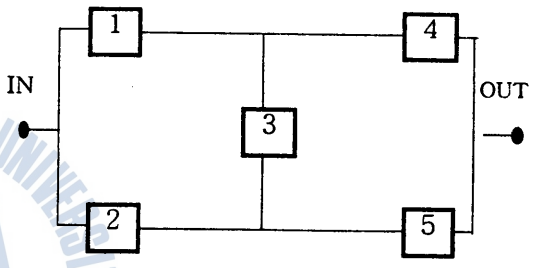
3. 예 제

Kim[8]에서 고려한 다음 complex 네트워크 구조에 대해 HH의 성능을 분석하였다.

하부시스템의 자료 : $n = 5$, $m = 1$

i	1	2	3	4	5
r_i	0.70	0.85	0.75	0.80	0.90
c_{1i}	2	3	2	3	1

네트워크 구조 :



<그림 6> Bridge네트워크 시스템

제약식 :

$$\sum_{i=1}^{5} c_i x_i \leq 20$$

신뢰도 목적함수 :

$$\begin{aligned} R_s(x) = {} & R_1(x_1)R_2(x_2) + Q_1(x_1)R_3(x_3)R_4(x_4) \\ & + R_1(x_1)Q_2(x_2)R_3(x_3)Q_4(x_4) \\ & + R_1(x_1)Q_2(x_2)Q_3(x_3)R_4(x_4)R_5(x_5) \\ & + Q_1(x_1)R_2(x_2)R_3(x_3)Q_4(x_4)R_5(x_5) \end{aligned}$$

여기서 $R_i(x_i) = (1 - (1 - r_i)^{x_i})$로 주어지고 $Q_i(x_i) = 1 - R_i(x_i)$이다.

이 예제를 통해 중복설계 문제를 간략히 설명하면 중복설계 갯수 x_i를 증가시키면 시스템의 안전도와 관련이 있는 전체 시스템의 신뢰도 $Rs(x)$는 향상된다. 그러나, 무게나 비용 등의

주어진 제약식 때문에 x_i를 무한정 증가시킬 수가 없다. 결국 이 문제는 주어진 제약식을 만족시키는 범위내에서 어떤 조합의 x_i를 선택해야 전체 시스템의 신뢰도를 최대화 시킬 수 있느냐는 문제로 귀착된다.

이 예제에 대한 global 최적해는 Ryoo/Sahinidis[17] 이 개발한 소프트웨어인 BARON (Brand And Reduce Optimization Navigator)을 이용하여 구하였으며, parser를 이용한 BARON의 구체적인 입력 형태는 부록A에 수록되어 있다. global 최적해는 $x^* = (3,2,2,1,1)$, $Rs(x^*) = 0.9932$ 이며, 이 예제에 대해 KY 방법과 본 논문에서 개발한 합성해법 HH방법의 성능비교 결과는 <표 1>에 나타나 있다.

<표 1> KY와 HH의 비교

	초기해	KY의 최적해	HH의 최적해
1	0.9765	0.9932	0.9932
2	0.9689	0.9923	0.9932
3	0.9695	0.9932	0.9932
4	0.9893	0.9923	0.9932
5	0.9913	0.9932	0.9932
6	0.9932	0.9932	0.9932
7	0.9802	0.9923	0.9932
8	0.9923	0.9923	0.9932
9	0.9634	0.9932	0.9932
10	0.9684	0.9932	0.9932

<표 1>에서 10개의 초기해 집단으로부터 KY는 6번, HH는 10번 각각 global 최적해를 얻었다. KY에 SA방법을 합성시킨 HH는 10개의 초기해 집단으로부터 10번 모두 global 최적해를 찾아 주는 좋은 결과를 얻을 수 있었다.

4. 전산 실험결과

발견적 합성 해법 HH의 성능을 평가하기 위해, Kim[8] 에서 분석한 경우 중에서 n=10, 15인 네트워크 구조에 대해 제약식의 갯수 m이 10개인 경우를 추가하여 <표2>에서와 같이 12가지 조합에 대해 분석하였다.

<표 2> KY와 HH의 성능비교를 위한 12가지 조합

		n = 10	n = 15
m = 1	$w_j = 2$	①	②
	$w_j = 3$	③	④
m = 5	$w_j = 2$	⑤	⑥
	$w_j = 3$	⑦	⑧
m = 10	$w_j = 2$	⑨	⑩
	$w_j = 3$	⑪	⑫

각각의 조합에는 10문제가 랜덤하게 구성되어 있으며, 여기서 w_j는 b_j값을 정하기 위한 난수(random number)로서 $b_j = w_j \times \sum_{j=1}^{n} C_{ji}$ 이다. <표2>에서 $w_j = 2$ 는 (1.5, 2.5) , $w_j = 3$ 은 (2.5, 3.5)의 값을 각각 의미한다. 제약식과 신뢰도 목적함수 R_s는 모형(P)와 같으며, 신뢰도 목적함수는 Kim[8]에서 언급된 recursive disjoint 방법에 의해 구해졌다. <표2>의 12가지 조합에 대해 KY 알고리듬과 HH 알고리듬의 성능을 비교한 결과는 <표3>에 나타나 있다. <표3>으로부터 HH 알고리듬은 각 조합의 10문제에 대해 적을때는 4번(n=10, m=1, w_j=3), 많을때는 9번(n=15, m=5,10, w_j=3)까지 KY 알고리듬을 개선시킨 것을 알 수 있고, 일관성있게 KY보다 좋은 결과를 제공해 주고 있다. 구체적인 최적신뢰도 값은 <부록 B>에 수록되어 있다.

<표 3> KY와 HH의 전산실험결과

		n = 10		n = 15	
		KY	HH	KY	HH
m = 1	$w_j = 2$	5/10	10/10	6/10	10/10
	$w_j = 3$	6/10	10/10	4/10	10/10
m = 5	$w_j = 2$	4/10	10/10	3/10	10/10
	$w_j = 3$	3/10	10/10	1/10	10/10
m = 10	$w_j = 2$	3/10	10/10	4/10	10/10
	$w_j = 3$	4/10	9/10	1/10	10/10

5. 결 론

본 논문에서는 선박, 비행기, 통신등의 콤플렉스 시스템의 안전도를 높이기 위한 중복설계 문제를 다루었으며 좀 더 좋은 최적해를 찾기 위해 기존의 KY 알고리듬에 두 가지의 전략인 초기해 집단 구성과 최종적인 SA 탐색을 합성한 HH(Hybrid Heuristic) 알고리듬을 개발하였다.

다양한 경우에 대한 전산실험결과, 본 논문에서 KY알고리듬에 두 가지 전략을 합성한 HH는 KY보다 일관성 있게 좋은 결과를 제공해 줌을 알 수 있었다.

따라서, 본 논문에서 개발한 발견적 합성해법인 HH는 본 논문에서 다룬 시스템의 안전도를 높이기 위한 신뢰도 최적화 문제나 이와 유사한 유형의 문제의 알고리듬으로서 유용한 대안이 된다고 사료된다.

참고문헌

[1] Cerny, V.,"A thermodynamical approach to thetraveling salesman problem : An efficient simulation algorithm" JOTA, vol 45, 1985, 41-45.

[2] Christopler, J.P., Wilhelm, W.E., "A perturbation scheme to improve Hiller's solution to the facilities layout problem", Management Science, vol. 30, 1238-1249, 1984.

[3] Cooper, M.W., "A survey of methods for pure nonlinear integer programming", Management Science, vol 27, 353-361, 1981.

[4] Glover, F., "Future paths for I.P. and Links to A.I.", Comput. & Ops. Res., vol 13, 553-549, 1986.

[5] Glover, F., "Heuristics for integer programming using surrogate constraints" Decision Science. vol 8, 156-166, 1977.

[6] Goldberg, D.E., "Genetic Algorithms inSearch, Optimization & MachineLearning", Addison-Wesley, 1989.

[7] Hillier, F.S., "Quantitative tools for plant layout analysis", J. Indust. Engineering, vol 14, 22-40, 1963.

[8] Kim, J.H., & Yum, B.J., "A heuristic method for solving redundancy optimization in complex systems", IEEE Trans. on Rel., vol 42, 572-578, 1993.

[9] Kirpatrick, S., Gelatt, C.D., & Vecchi, M.P., "Optimization by simulated annealing", Science, vol 220, 671-680, 1983.

[10] Kohda, T., & Inoue, K., "A reliability optimization method for complex systems with the criterion of local optimality", IEEE Trans. Reliability, vol R-31, 109-111, 1982.

[11] Lawler, E.L & Bell, M.D., "A method for solving discrete optimization problems", Operations Res., vol 14, 1098-1112, 1966.

[12] Liggett, R.S., "The quadratic assignment problem: An experi-mental evaluation of solution strategies", Management Sci-ence, vol 27, 442-458, 1981.

[13] Mitchell, T.J., "An algorithm for the construction of D-optimal experimental designs", Technometrics, vol 16, 203-201,

1974.

[14] Morin, T.L., & Marsten, R.E., "An algorithm for nonlinear knapsack problem", Management Science, vol 22, 1147-1158, 1976.

[15] Nakagawa,Y., & Nakashima, K.,"A heuristic method for determining optimal reliability allocation", IEEE Trans. Reliability, vol R-26, 156-161, 1971.

[16] Nakagawa, Y., Nakashima, K., & Hattori, Y., "Optimal reliability allocation by Branch and Bound techniques", IEEE Trans. Reliability, vol 27, 31-38, 1978.

[17] Ryoo, H.S., &Sahinidis, N.V., "A branch-and-reduce approach to gloval optimization", Journal of Global Opt., 8:2, 107-139, 1996.

[18] Shi, D.H., "A new heuristic algorithm for constrainted redundancy optimization in complex systems", IEEE Trans. Reliablility, vol R-36, 621-623, 1987

부록A. BARON의 parser의 입력자료

```
OPTIONS {
results: 1;
maxiter: 200;
summary: 1;
times: 1;
}
MODULE: NLP;
INTEGER_VARIABLES x1,x2,x3,x4,x5;
LOWER_BOUNDS {
x1: 1;
x2: 1;
x3: 1;
x4: 1;
x5: 1;
}
EQUATION e1;
e1: 2*x1+3*x2+2*x3+3*x4+x5 <= 20;
OBJ: minimize
-((1-.3^x1)*(1-.15^x2) +
 .3^x1*(1-.25^x3)*(1-.2^x4) +
 (1-.3^x1)*.15^x2*(1-.25^x3)*(1-.2^x4) +
  (1-.3^x1)*.15^x2*.25^x3*(1-.2^x4)*(1-.1^x5) +
 .3^x1*(1-.15^x2)*(1-.25*x3)*.2^x4*(1-.1^x5));
```

부록B. KY와 HH의 신뢰도 최적해

<표 4> n = 10, 15 , m = 1 인 경우

			n = 10		n = 15	
			KY	HH	KY	HH
m = 1	$w_j = 2$	1	.994935	.994935	.998747	.998747
		2	.966560	.966560	.977592	.982624
		3	.958630	.963278	.998178	.998178
		4	.981659	.985698	.975448	.976062
		5	.985217	.986492	.946917	.953793
		6	.999206	.999206	.929367	.929367
		7	.954568	.954568	.876539	.892591
		8	.995063	.996690	.955328	.955328
		9	.953225	.962895	.866420	.866420
		10	.986751	.986751	.821225	.821225
	$w_j = 3$	1	.999909	.999909	.999986	.999986
		2	.999777	.999777	.999284	.999284
		3	.999452	.999452	.999974	.999974
		4	.999272	.999450	.998206	.998509
		5	.999063	.999151	.998199	.998458
		6	.9999925	.999993	.998834	.998834
		7	.998715	.998715	.991143	.992096
		8	.999873	.999893	.996091	.997570
		9	.998626	.998626	.993246	.993544
		10	.999640	.999640	.991612	.991644

<표 5> n = 10, 15 , m = 5 인 경우

			n = 10		n = 15	
			KY	HH	KY	HH
m = 5	$w_j = 2$	1	.982908	.987567	.866576	.903623
		2	.942387	.951164	.736567	.740860
		3	.911197	.927202	.761984	.802917
		4	.910806	.945282	.896113	.896113
		5	.935018	.937574	.764524	.781204
		6	.952777	.967218	.787669	.788723
		7	.941158	.941158	.810096	.810096
		8	.929935	.929935	.892412	.904850
		9	.903751	.903751	.876191	.889088
		10	.936537	.936537	.925820	.925820
	$w_j = 3$	1	.999621	.999621	.989787	.995311
		2	.996726	.996964	.977018	.982046
		3	.993363	.993555	.987635	.988466
		4	.997653	.997653	.988250	.993918
		5	.996539	.997179	.969389	.980021
		6	.996049	.998152	.986174	.986174
		7	.997961	.998365	.988583	.990379
		8	.996721	.997760	.993302	.994408
		9	.994633	.996011	.990595	.993985
		10	.997586	.997586	.996486	.996746

<표 6> n = 10, 15 , m = 10 인 경우

			n = 10		n = 15	
			KY	HH	KY	HH
m = 10	$w_j = 2$	1	.922614	.922920	.868255	.903623
		2	.905499	.913184	.839148	.839148
		3	.876864	.898462	.896113	.896113
		4	.952017	.964607	.878791	.878791
		5	.889307	.889307	.815999	.850595
		6	.921018	.939137	.853042	.862337
		7	.914826	.919438	.798798	.812645
		8	.940642	.940642	.766307	.769345
		9	.950043	.950337	.740496	.748476
		10	.918179	.926051	.879625	.879625
	$w_j = 3$	1	.996146	.996137	.989976	.995312
		2	.995221	.995221	.983351	.988950
		3	.994766	.994766	.988250	.993193
		4	.997528	.997993	.982582	.990580
		5	.994402	.994402	.986828	.988262
		6	.995620	.997085	.992101	.992101
		7	.995053	.996356	.980854	.983845
		8	.998230	.998375	.972216	.974681
		9	.997869	.998471	.974702	.978979
		10	.994559	.995483	.989907	.990146