

理學碩士 學位論文

Web - site

Solver

A study on the Development of Web - site
Solver for solving the Reliability Redundancy
Optimization problems

指導教授 金 宰 煥

2001年 2月

韓國海洋大學校 大學院

應 用 數 學 科

元 海 淵

本 論 文 元海淵 理學碩士 學位論文 認 准

主 審 : 理 學 博 士 琴 尙 昊

委 員 : 工 學 博 士 金 宰 煥

委 員 : 理 學 博 士 裴 在 國

2001年 2月

韓國海洋大學校 大學院

應 用 數 學 科

元 海 淵

Abstract	ii
.	1
1.1	1
1.2	4
.	5
Web-site Solver	5
2.1	5
2.2	6
2.3	6
2.4	14
2.4.1	14
2.4.2 SA	19
2.5	21
.	25
3.1 1	25
3.2 2	30
.	35
.	36
A	37
B	53

ABSTRACT

This paper deals with developing the Web-site solver for solving three classes of redundancy reliability optimization problems which are generated in series systems, parallel systems and complex systems. Inputs of the solver are completely processed on the Web-site, and consisted in four parts, that is, user authentication, select system, input data and confirmation.

HH(Hybrid-Heuristic) algorithm is incorporated in our solver for solving the given three classes of problems. The algorithm is moderately combined GA(Genetic Algorithm) with the modified SA(Simulated Annealing) algorithm to alleviate the risks of being trapped in a local optimum. We notice that the good solutions are obtained in examples.

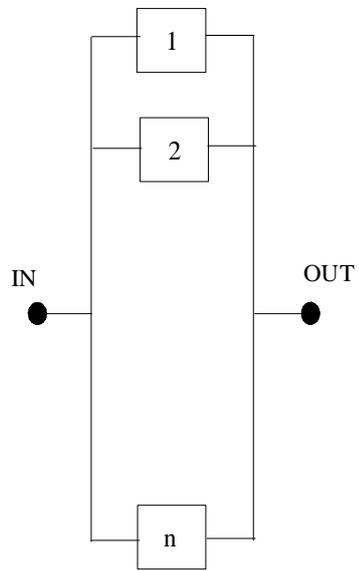
I.

1.1

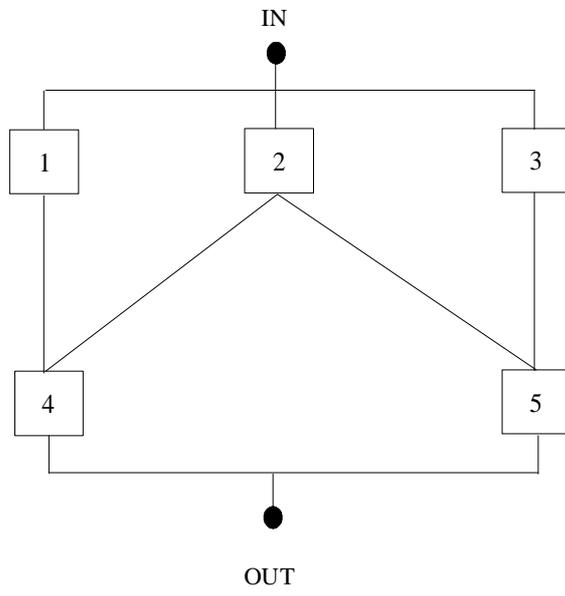
ARPANET (1969) .
NSFNET (1986) 가
(1990) .
E-mail, Ftp, Newsgroup
1991 CERN WWW(World Wide Web) 가
가
가 ,
Web-site solver .
solver , , ,
4 Web-site .
solver < 1>, < 2>, < 3> ,
,
가 ,
(Hybrid-Heuristic Algorithm) 가 .



< 1 >



< 2 >



< 3 >

1.2

Web-site solver ,
Web-site PHP (Professional HTML
Preprocessor) . , solver (local
optimum) , .

II.

Web-site Solver

Web-site solver
 (local optimun)
 (Hybrid-Heuristic Alogrithm)

2.1

Web-site solver

n :
 m :
 x_i : i
 $i = 1, 2, \dots, n$.
 x : (x_1, x_2, \dots, x_n)
 u_i : x_i upper bound.
 x^* :
 $g_{ji}(x_i)$: i j
 b_j : j 가
 $R_i(x_i)$: x_i i
 $R_S(x)$:
 $Q_i(x_i)$: x_i i

pop_size :
GTOT : GA iteration
STOT : SA iteration

2.2

software

: pentium(r) pro processor, 128.0 MB RAM
OS : Redhat LINUX 6.0 kernel 2.2.9-6kr
Web : Apache 1.3.9
Backend : PHP 3.0.12
DBMS : MySQL 3.22.22

2.3

user가

NRO Package

4

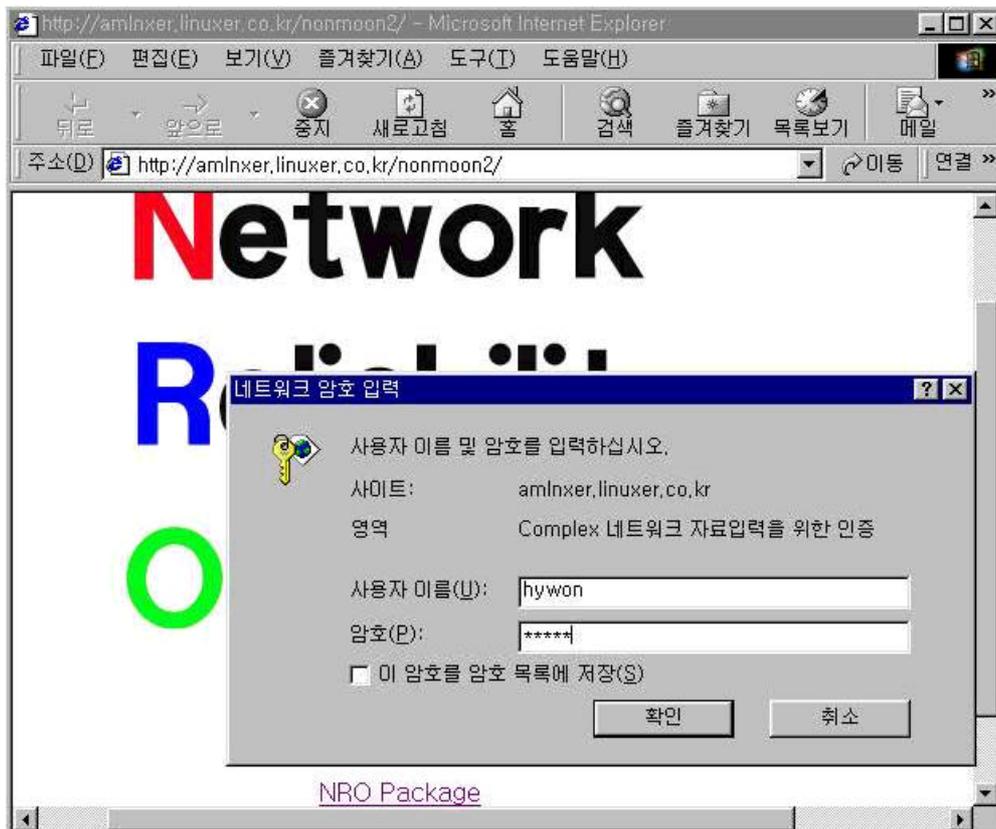
- i)
- ii)
- iii)
- iv)

i)

(User Authentication) DBMS MySQL

Header()

Authentication Request



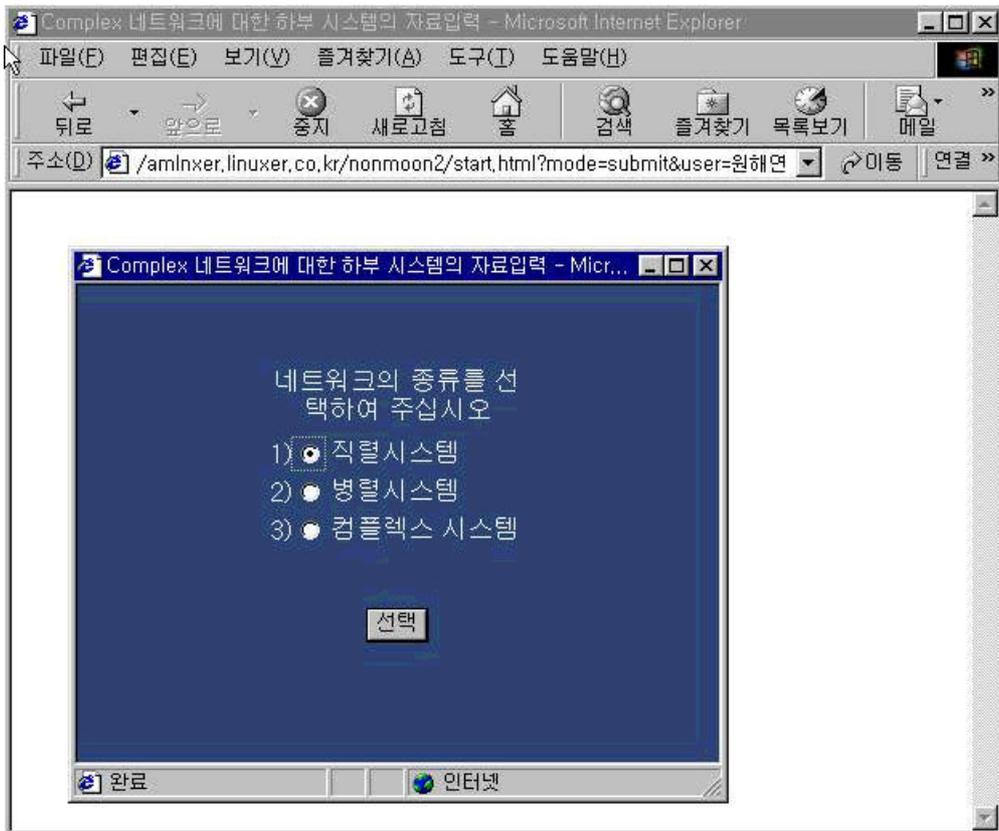
< 4 >

< 4 >

가

5>

< 6>



< 6>

iii)

(n)

(< 7>) \Rightarrow (< 8>) \Rightarrow (m) (<

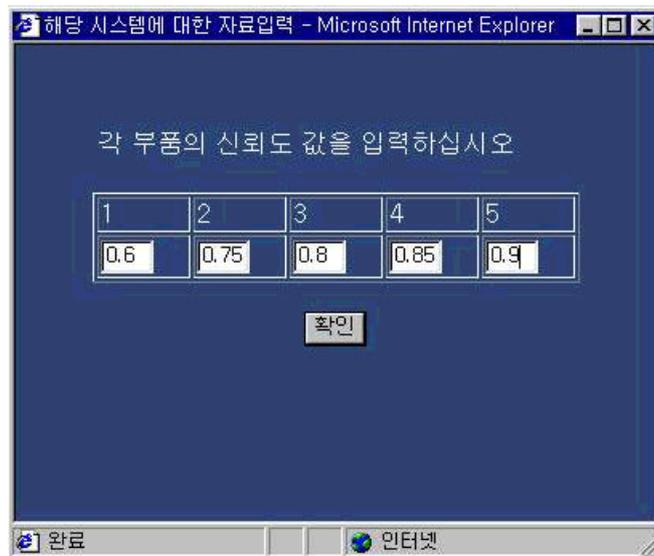
9>) \Rightarrow (< 10>) 4 ,

incidence matrix (< 11>)

가 .



< 7 >



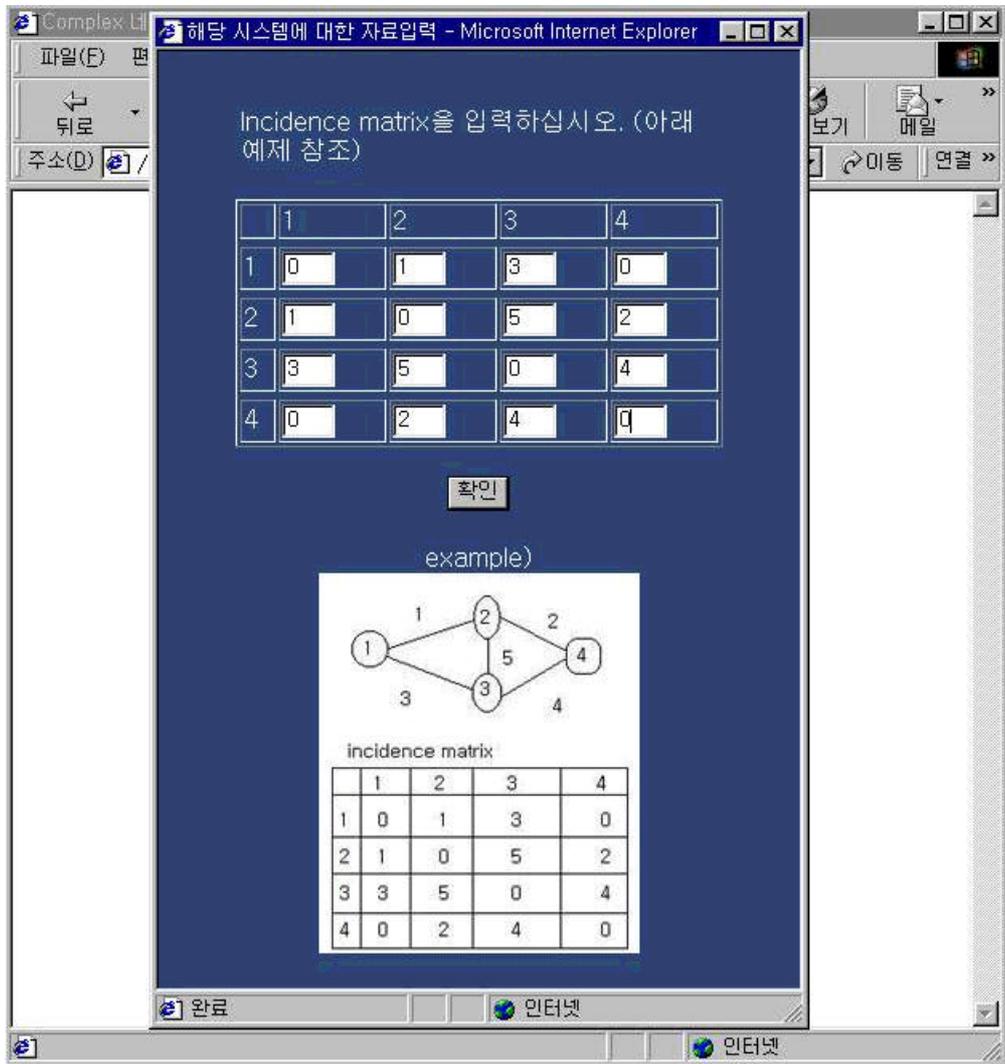
< 8 >



< 9 >



< 10 >



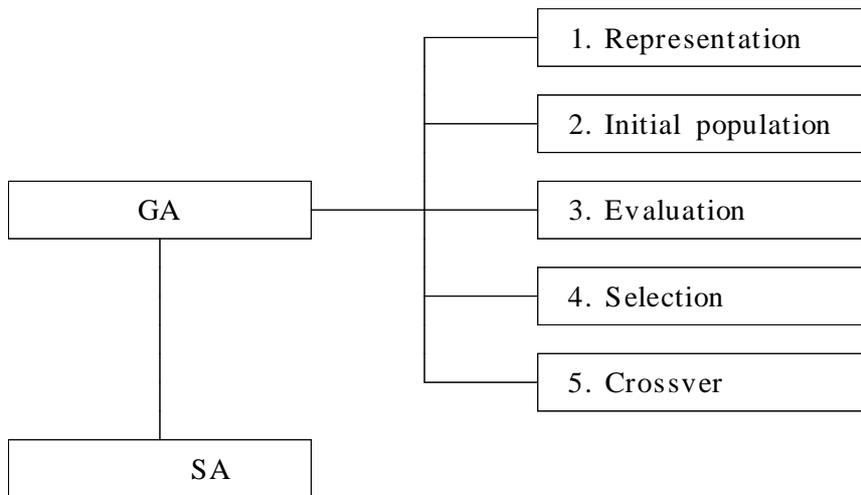
< 11> incidence matrix

iv)

(< 12>) 4가
 -> -> -> 4가

2.4

(local optimum)
 (reoptimization)
 < 14>
 procedure) SA



< 14>

2.4.1

(fitness)가 , (generation)
 가 (crossover)
 (population)
 (reproduction)
 (mutation)

5가

- i) (Representation)
- ii) (Initial population)
- iii) 가(Evaluation)
- iv) (Selection)
- v) (Crossover)

i)

가

$$x_i \quad 1 \leq x_i \leq u_i$$

, x_i u_i

$u_1 = 4$ x_i 3bit $u_1 = 6, u_2 = 4,$

$u_3 = 6, u_4 = 4, u_5 = 10$ x_1, x_2, x_3, x_4, x_5

3, 3, 3, 3, 4 bit 16bit가

$x = (2, 3, 3, 3, 7)$

$$x = [x_{13} x_{12} x_{11} x_{23} x_{22} x_{21} x_{33} x_{32} x_{31} x_{43} x_{42} x_{41} x_{54} x_{53} x_{52} x_{51}]$$

$$= [0 1 0 0 1 1 0 1 1 0 1 1 1 1]$$

$$x_{ij} \quad x_i \quad j \quad \text{bit}$$

ii)

pop_size $\langle 1 \rangle$ 가

Procedure : Initialization

```
begin
  count = 1;
  for  $i \leftarrow 1$  to pop_size do
    produce a random chromosome  $v_i$  ;
    if (  $v_i$  is not feasible ) then
      count  $\leftarrow$  count+1 ;
    end
  end
  if ( count == pop_size ) then
    goto begin;
  end
end
```

< 1 >

pop_size = 5 , $u_1 = 6, u_2 = 4, u_3 = 6, u_4 = 4,$
 $u_5 = 10$, .

$$x_1 = [0\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 1] = (2, 3, 3, 3, 7)$$

$$x_2 = [0\ 1\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 0\ 0\ 1] = (3, 6, 3, 3, 1)$$

$$x_3 = [1\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 0\ 1] = (6, 5, 3, 6, 5)$$

$$x_4 = [0\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 1] = (2, 3, 3, 3, 7)$$

$$x_5 = [1\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 1\ 1] = (6, 2, 3, 3, 7)$$

iii) x_i 가 $eval(x_i)$.

$$eval(x_i) = \begin{cases} R_S(x_i) & : x_i \text{가 가} \\ R_S(x_i) - M & : x_i \text{가 가 가} \end{cases}$$

, $M (0 < M < 1)$ x_i 가 가 가 penalty .

iv) . 가

1. x_i (fitness) .

$$eval(x_i) = R_S(x_i), i = 1, 2, \dots, pop_size$$

2. .

$$F = \sum_{i=1}^{pop_size} eval(x_i)$$

$$\bar{F} = \frac{F}{pop_size}$$

3. x_i E_i .

$$E_i = \frac{eval(x_i)}{\bar{F}}$$

4. x_i .

$$A_i = [E_i + 0.5]$$

$$. \quad x_i \quad A_i \quad x_i \quad 3$$

v)

(crossover) 2 .

$$x_1, x_2 \text{가} \quad 3 \quad ,$$

$$x_1', x_2' \quad .$$

↓

$$x_1 = [a_1 a_2 a_3 a_4 a_5 a_6 a_7 a_8 a_9 a_{10}]$$

$$x_2 = [b_1 b_2 b_3 b_4 b_5 b_6 b_7 b_8 b_9 b_{10}]$$

⇓

$$x_1' = [a_1 a_2 a_3 b_4 b_5 b_6 b_7 b_8 b_9 b_{10}]$$

$$x_2' = [b_1 b_2 b_3 a_4 a_5 a_6 a_7 a_8 a_9 a_{10}]$$

```

Procudure : Crossover

begin
  k ← 1 ;
  while ( k ≤  $\frac{pop - size}{2}$  ) do
     $x_i, x_j$  .
    -- (  $i \neq j$  ,  $x_i, x_j$  . )
    p .
    -- (  $x_i$  bit . )
    p .

    k ← k + 1 ;
  end
end
end

```

< 2 >

2.4.2 SA

SA Metropolis [8] , Kirkpatrick [6]
 SA , Cerny [1]
 TSP (Travelling Salesman Problem) SA .
 SA (reoptimization procedure)

$$x_c = (x_1, x_2, \dots, x_n)$$

, SA .

```

Procure :      SA

begin
  k ← 1 ;
  while ( k ≤ STOT ) do

    x'_c = x_c ;
          i, j
    --(      , 0 ≤ i ≠ j ≤ n      .)
          r
    --(      , r = 0      1      .)
    , r = 0
    x'_c = (x'_1, x'_2, ... , x'_i - 1, ... , x'_j - 1, ... , x'_n);
          ,
    x'_c = (x'_1, x'_2, ... , x'_i + 1, ... , x'_j + 1, ... , x'_n);
    eval(x'_c) > eval(x_c)
    x_c = x'_c ;
    k ← k + 1 ;
  end
end

```

< 3> SA

2.5

step- 1. pop_size, GTOT, STOT . num = 1.

step- 2. (Initial population) pop_size

step- 3.(Evaluation) $x_i, i = 1, 2, \dots, \text{pop_size}$
(fitness)

$$eval(x_i) = \begin{cases} R_S(x_i) & : x_i \text{가 가} \\ R_S(x_i) - M & : x_i \text{가 가 가} \end{cases}$$

step- 4.(Selection)

4.1

$$F = \sum_{k=1}^{\text{pop_size}} eval(x_i),$$

$i = 1, 2, \dots, \text{pop_size}$

$$\bar{F} = \frac{F}{\text{pop_size}}$$

4.2 x_i E_i

$$E_i = \frac{eval(x_i)}{\bar{F}}, i = 1, 2, \dots, \text{pop_size}$$

4.3 x_i A_i

$$A_i = \lfloor E_i + 0.5 \rfloor,$$

4.4 $i = 1, 2, \dots, \text{pop_size}$ x_i

```

begin
  k = 1;
  i = 1;
  while ( i ≤ pop_size ) do
    if ( Ai > 1 ) then
      j = 1;
      while ( j ≤ Ai ) do
        x'k = xi;
        j ← j + 1;
        k ← k + 1;
      end
    else
      continue ;
    end
  end

  k = 1;
  while ( k ≤ pop_size) do
    xk = x'k;
  end
end

```

step- 5.(Crossover)

x_i

```

begin
  k ← 1 ;
  while ( k ≤  $\frac{pop\_size}{2}$  ) do
     $x_i, x_j$ 
    -- (  $i \neq j$  ,  $x_i, x_j$  . )
    p
    -- (  $x_i$  bit . )
    p

    k ← k + 1 ;
  end
end

```

step-6.(Finish check)

num > *GTOT* *eval*(x_i) step-7

step-3

step-7.(SA)

7.1 num = 1.

7.2 $x'_c = x_c$.

$0 \leq i \neq j \leq n$ i, j .

7.3 $r = 0$ 1 .

7.4 , $r = 0$

$$x'_c = (x'_1, x'_2, \dots, x'_i - 1, \dots, x'_j - 1, \dots, x'_n)$$

,

$$x'_c = (x'_1, x'_2, \dots, x'_i + 1, \dots, x'_j + 1, \dots, x'_n)$$

$$7.5 \quad eval(x'_c) > eval(x_c) \quad x_c = x'_c .$$

num \leftarrow num+1;

$$7.6 \quad num > STOT \quad \text{step-8} \quad . \quad 7.2 \quad .$$

step-8. STOP.

list < B> .

III.

solver

3.1 1

solver < 6 >

: $n = 5, m = 1$

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

:

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

$$x_i \geq 1, i = 1, 2, 3, 4, 5.$$

:

$$\begin{aligned}
R_S(x) &= R_1(x_1)R_2(x_2)Q_3(x_3)Q_5(x_5) \\
&+ Q_1(x_1)R_3(x_3)R_4(x_4)Q_5(x_5) \\
&+ [R_1(x_1)R_3(x_3) + (R_3(x_3)R_5(x_5) \\
&+ R_5(x_5)R_1(x_1) - 2R_1(x_1)R_3(x_3)R_5(x_5))] \\
&\times [R_2(x_2) + R_4(x_4) - R_2(x_2)R_4(x_4)]
\end{aligned}$$

$$R_i(x_i) = (1 - (1 - r_i)^{x_i}) \quad Q_i(x_i) = 1 - R_i(x_i)$$

x_i

$$\left\{ \begin{array}{l}
x_1 \leq \frac{20 - (3x_2 + 2x_3 + 3x_4 + x_5)}{2} \\
x_2 \leq \frac{20 - (2x_1 + 2x_3 + 3x_4 + x_5)}{3} \\
x_3 \leq \frac{20 - (2x_1 + 3x_2 + 3x_4 + x_5)}{2} \\
x_3 \leq \frac{20 - (2x_1 + 3x_2 + 2x_3 + x_5)}{3} \\
x_5 \leq 20 - (2x_1 + 3x_2 + 2x_3 + 3x_4)
\end{array} \right.$$

$$x_1 \leq 6, x_2 \leq 4, x_3 \leq 6, x_4 \leq 4, x_5 \leq 10 \quad x_i$$

16bit

pop_size = 10

$$\begin{aligned}
x_1 &= [1010010010010001] \\
x_2 &= [0010010110010001] \\
x_3 &= [1111111110111101] \\
x_4 &= [0111011111111001] \\
x_5 &= [111110110110011] \\
x_6 &= [1110110111011011] \\
x_7 &= [0011111111111001] \\
x_8 &= [0010010010010011] \\
x_9 &= [0011111111110001] \\
x_{10} &= [0111011011110101]
\end{aligned}$$

x_i (fitness) x_i
penalty

$$eval(x_i) = \begin{cases} R_S(x_i) & : x_i \text{가 가} \\ R_S(x_i) - 0.9 & : x_i \text{가 가 가} \end{cases}$$

x_i

$$eval(x_1) = R_S(x_1) = 0.966387$$

$$eval(x_2) = R_S(x_2) = 0.960302$$

$$eval(x_3) = R_S(x_3) - 0.9 = 0.1$$

$$eval(x_4) = R_S(x_4) - 0.9 = 0.099998$$

$$eval(x_5) = R_S(x_5) - 0.9 = 0.099997$$

$$eval(x_6) = R_S(x_6) - 0.9 = 0.099995$$

$$eval(x_7) = R_S(x_7) - 0.9 = 0.099982$$

$$eval(x_8) = R_s(x_8) - 0.9 = 0.897191$$

$$eval(x_9) = R_s(x_9) - 0.9 = 0.099981$$

$$eval(x_{10}) = R_s(x_{10}) - 0.9 = 0.099974$$

(fitness) 가 x_i 가
< 4> .

string no.	Initial population	x value	$R_s(x)$	$\frac{f_i}{\sum f}$	$\frac{f_i}{f}$	actual count
1.	[1010010010010001]	(3,1,1,1,1)	0.966387	0.27	2.74	3
2.	[0010010110010001]	(1,1,3,1,1)	0.960302	0.27	2.74	3
3.	[111111110111101]	(7,7,7,3,13)	0.100000	0.03	0.28	1
4.	[011101111111001]	(3,5,7,7,9)	0.099998	0.03	0.28	0
5.	[111110110110011]	(7,7,3,3,3)	0.099997	0.03	0.28	0
6.	[1110110111011011]	(7,3,3,5,11)	0.099995	0.03	0.28	0
7.	[011111111111001]	(1,7,7,7,9)	0.099982	0.03	0.28	0
8.	[0010010010010011]	(1,1,1,1,3)	0.897191	0.25	2.55	3
9.	[0011111111110001]	(1,7,7,7,1)	0.099981	0.03	0.28	0
10.	[0111011011110101]	(3,5,5,7,5)	0.099974	0.03	0.28	0
Sum			3.523870	1.0	10.0	
Average			0.352381	0.1	1.0	
Max			0.966387	0.27	2.74	

< 4>

< 4> actual count (population)
 2.7 step-4. step-5.
 (population) < 5> .

(randomly selected)

string no.	Mate	crossover site		New population	$R_s(x)$
1.	[1010010010010001]	4	8	[1010010010010001]	0.966387
2.	[1010010010010001]	7	7	[1010010110111101]	0.098762
3.	[1010010010010001]	9	2	[1010010010010011]	0.969380
4.	[0010010110010001]	1	8	[0010010110010001]	0.960302
5.	[0010010110010001]	8	1	[0010010010010011]	0.897191
6.	[0010010110010001]	10	10	[0010010110010011]	0.965402
7.	[111111110111101]	2	7	[1111111010010001]	0.099995
8.	[0010010010010011]	5	1	[0010010110010001]	0.960302
9.	[0010010010010011]	3	2	[0010010010010001]	0.891312
10.	[0010010010010011]	6	10	[0010010010010001]	0.891312
Sum					7.700346
Avg					0.770035
Max					0.969380

< 5 >

100 2.7 step-7. SA 1000

< 6 > .

global $x^* = (3, 2, 2, 1, 1), R_s(x^*) =$

0.993216

10 9

1 GA- step 8 2.7 step-7. SA

	GA	(x_c)	(x^*)
1	0.965402		0.993216
2	0.963420		0.993216
3	0.897250		0.968858
4	0.919844		0.993216
5	0.968957		0.993216
6	0.963420		0.993216
7	0.921133		0.993216
8	0.968957		0.993216
9	0.993216		0.993216
10	0.924330		0.993216

< 6 >

3.2 2

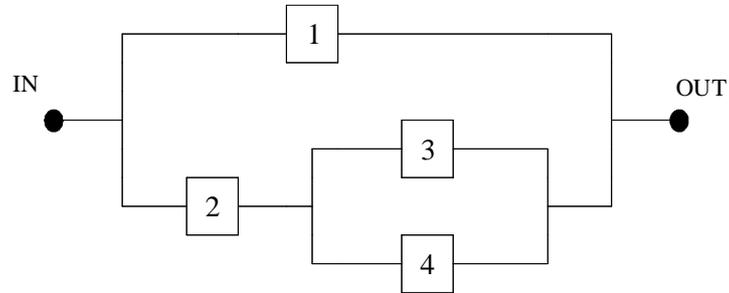
가

.

: $n = 4, m = 2$

i	1	2	3	4
r_i	0.80	0.75	0.70	0.65
c_{1i}	6	4	3	2
c_{2i}	9	4	4	3

:



:

$$\sum_{i=1}^4 c_{1i}x_i \leq 30$$

$$\sum_{i=1}^4 c_{2i}x_i \leq 40$$

$$x_i \geq 1, \quad , i = 1, 2, 3, 4.$$

:

$$R_S(x) = R_1(x_1) + Q_1(x_1)R_2(x_2)R_4(x_4) + Q_1(x_1)R_2(x_2)R_3(x_3)Q_{4(x)4}$$

x_i				12bit	pop
x_i	(fitness)	actual count	< 7 >		
x_i				(population)	

string no. Initial population x value $R_S(x)$ $\frac{f_i}{\sum f}$ $\frac{f_i}{f}$ actual count

1.	[001011011001]	(1,3,3,1)	0.995015	0.19	1.86	2
2.	[001011001011]	(1,3,1,3)	0.994343	0.19	1.86	2
3.	[001001011101]	(1,1,3,5)	0.949979	0.18	1.78	2
4.	[011101011111]	(3,5,3,7)	0.099992	0.02	0.19	0
5.	[001001101001]	(1,1,5,1)	0.949872	0.18	1.78	2
6.	[111001101111]	(7,1,5,7)	0.099997	0.02	0.19	0
7.	[001001001011]	(1,1,1,3)	0.948071	0.18	1.78	2
8.	[111001001001]	(7,1,1,1)	0.099996	0.02	0.19	0
9.	[111001101111]	(7,1,5,7)	0.099997	0.02	0.19	0
10.	[011111001011]	(3,7,1,3)	0.099897	0.02	0.19	0

Sum		5.337159	1.0	10.0
Avg		0.533716	0.1	1.0
Max		0.995015	0.19	1.86

< 7 >

< 7 >	actual count	(population)
2.7	step-4.	step-5.
(population)	< 8 >	.

(randomly selected)

string no.	Mate	crossover site	New population	$R_S(x)$	
1.	[001011011001]	10	8	[001011011011]	0.096647
2.	[001011011001]	3	2	[001011001011]	0.994343
3.	[001011001011]	2	2	[001011011001]	0.995015
4.	[011011001011]	8	7	[001011001001]	0.976203
5.	[001001011101]	9	9	[001001011011]	0.949826
6.	[001001011101]	7	5	[001001101001]	0.949872
7.	[001001101001]	6	5	[001001011101]	0.949979
8.	[001001101001]	4	7	[001001101011]	0.049984
9.	[001001001011]	5	9	[001001001101]	0.949764
10.	[001001001011]	1	8	[001001001001]	0.934250
Sum					7.845883
Avg					0.784588
Max					0.995015

< 8 >

100 2.7 step-7. SA 1000

< 9 > .

global $x^* = (3, 1, 1, 1), R_S(x^*) = 0.997370$

10 8

8 GA-step , 2

SA-step 가 .

	GA	(x_c)	(x^*)
1	0.997370		0.997370
2	0.997370		0.997370
3	0.997370		0.997370
4	0.994343		0.995989
5	0.997370		0.997370
6	0.997370		0.997370
7	0.976203		0.992943
8	0.997370		0.997370
9	0.997370		0.997370
10	0.997370		0.997370

< 9 >

IV.

Web-site solver . solver

4 Web-site .
(Hybrid-Heuristic algorithm)

가 .
(local optimum) 1 GA
(Genetic Algorithm) , (reoptimization
procedure) SA(Simulated Annealing)

solver ,

- [1] Cerny, V., "Thermodynamical Approach to the Traveling Salesman Problem" JOTA, vol 45, 41-51, 1985.
- [2] Cooper, M.W., "A survey of methods for pure nonlinear integer programming", Management Science, vol 27, 353-361, 1981.
- [3] Glover, F., "Future paths for IP. and Links to A.I.", Comput, & Ops. Res., vol 13, 553-549, 1986.
- [4] Glover, F., "Heuristics for integer programming using surrogate constraints", Decision Science. vol 8, 156-166, 1977.
- [5] Hillier, F.S., "Quantitative tools for plant layout analysis", j. Indust. Engineering, vol 14, 22-40, 1973.
- [6] Kirkpatrick, S., Gelatt, C.D., & Vecchi, M.P., "Optimization by simulated annealing", Science, vol 220, 671-680, 1983.
- [7] Lawler, E.L & Bell, M.D., "A method for solving discrete optimization problems", Operations Res., vol 14, 1098-1112, 1966.
- [8] Metropolis, N., "Equation of State Calculations by Fast Computing Machines", Journal of Chemical Physics, vol 21, 1087-1092, 1953.
- [9] Mitchell, T.J., "An algorithm for the construction of D-optimal experimental designs", Technometrics, vol 16, 201-203, 1974.
- [10] Morin, T.L., & Marsten, R.E., "An algorithm for nonlinear knapsack problem", Management Science, vol 22, 1147-1158, 1976.


```

        echo("<input type='hidden'
name='r[$i]' value='$r[$i] '>");
    }
    echo("
</TABLE>
</CENTER>
</FORM>
");
}
if($mode == 'input3')
{
    echo(" <FORM name='form'
method='post' action='$PHP_SELF'>
<CENTER><BR>
<table border='0'
width='80%'>
<tr>
<td align='center '>
.</td>
</tr>
</table><br>
<TABLE border='1' width='80%'>
");
    for ($i=1, $p=1, $count=1; $i <= ($m+1)
; $i++) {
        echo("<tr>");
        if($i=='1'){
            echo("<td> &nbsp; </td>");
            for ($j=1; $j <= $node; $j++) {
                echo("<td> $j </td>");
            }
            echo("<td> RSH </td>");
            echo("</tr>");
        } else {
            echo("<tr>");
            echo("<td>      $p </td>");
            $p++;
            for ($j=1; $j <= $node+1; $j++) {
                echo("<td> <input type='text '
name='c[$count]' size=3
maxlength=5> </td>");
                $count++;
            }
            echo("</tr>");
        }
    }
    echo("
</TABLE>
<TABLE><BR>
<input type='submit '
name='submit' value='      ' >
<input type='hidden'
name='mode' value='input4'>
<input type='hidden'
name='node' value='$node' >
<input type='hidden' name='m'
value='$m'>
<input type='hidden'
name='network' value='$network' >
<input type='hidden'
name='count' value='$count' >
<input type='hidden'
name='username' value='$username' >
");
    for ($i=1; $i <= $node; $i++) {
        echo("<input type='hidden'

```

```

name='r[$i]' value='$r[$i]'>");
    }
/*
    for ($i=1;$i<$count;$i++) {
        echo("<input type='hidden'
name='c[$i]'
value='$c[$i]'>");
    }
*/
echo("
</TABLE>
</CENTER>
</FORM>
");
}
if($mode == 'input4')
{
    echo("
<CENTER>

<table width='80%'><tr><td
align='center'>
");
    if($network == 1) {
        $network1 = "        ";
    } else {
        $network1 = "        ";
    }
    echo(" $network1");
    echo("
:
</td></tr></table>
</CENTER>
<BR><BR>
<CENTER>
<TABLE width='80%' border='1'>
<tr>
<td> &nbsp; </td>
");
    for ($i=1;$i<=$node;$i++)
        echo("<td> $i </td>");
    echo("<td> RHS </td></tr><tr>");
    echo("<td> </td>");
    for ($i=1;$i<=$node;$i++)
        echo("<td> $r[$i] </td>");
    echo("<td> &nbsp; </td></tr>");
    for ($i=1,$spos=1,$step=$node,$step1=1;$
i<=$m;$i++){
        echo("<tr>");
        echo("<td> $i </td>");

        for ($j=$spos;$j<=($step+$step1);$j++, $p
os++) {
            echo("<td> $c[$j] </td> ");
        }
        echo("</tr>");
        $step = $step+$node;
        $step1++;
    }
    $que=" insert into system_data1
values('','$username','$network','$node',
'$edges','$m','$r[1]','$r[2]','$r[3]',
'$r[4]','$r[5]','$r[6]','$r[7]',
'$r[8]','$r[9]','$r[10]','$r[11]','$r[

```

```

12] ', '$r [13] ', '$r [14] ', '$r [15] ', '$r [16
] ', '$r [17] ', '$r [18] ', '$r [19] ', '$r [20] '
, '$r [21] ', '$r [22] ', '$r [23] ', '$r [24] ', '
$r [25] ', '$r [26] ', '$r [27] ', '$r [28] ', '$r
[29] ', '$r [30] ', '$c [1] ', '$c [2] ', '$c [3] '
, '$c [4] ', '$c [5] ', '$c [6] ', '$c [7] ', '$c [8
] ', '$c [9] ', '$c [10] ', '$c [11] ', '$c [12] ',
'$c [13] ', '$c [14] ', '$c [15] ', '$c [16] ', '$
c [17] ', '$c [18] ', '$c [19] ', '$c [20] ', '$c [
21] ', '$c [22] ', '$c [23] ', '$c [24] ', '$c [25
] ', '$c [26] ', '$c [27] ', '$c [28] ', '$c [29] '
, '$c [30] ', '$e [1] ', '$e [2] ', '$e [3] ', '$e [
4] ', '$e [5] ', '$e [6] ', '$e [7] ', '$e [8] ', '$
e [9] ', '$e [10] ', '$e [11] ', '$e [12] ', '$e [1
3] ', '$e [14] ', '$e [15] ', '$e [16] ', '$e [17]
', '$e [18] ', '$e [19] ', '$e [20] ', '$e [21] ',
'$e [22] ', '$e [23] ', '$e [24] ', '$e [25] ', '$
e [26] ', '$e [27] ', '$e [28] ', '$e [29] ', '$e [
30] ')";
$result=mysql_query($que,$connect);
if($result)
{
}
else
{
    echo(" <script>
        window.alert('DB   가
        .')
        history.go(-1)
        </script>
    ");
    exit;
}
echo("

```

```

</center>
</table>
<center><br>
<table width='30%'>
    <tr>
        <td><a
href='$PHP_SELF?mode=end&username=$use
rname '>|    |</a>
        </td> " );
    $temp .= $username;
    $temp .= '=';
    $temp .= $network;
    $temp .= '=';
    $temp .= $node;
    $temp .= '=';
    $temp .= $m;
    for ($i=1;$i<=$node;$i++){
        $temp .= '=';
        $temp .= $r[$i];
    }
    for ($i=1;$i<=($node*$m+$m);$i++){
        $temp .= '=';
        $temp .= $c[$i];
    }
    echo("
        <td><a
href='$PHP_SELF?mode=edit1&emp=$temp '
>|    |</a>
        </td>
    </tr>
</table>
</center>
");

```

```

}
if($mode=='end')
{
    echo("<center><br><font
color='red'>");
    echo("$username");
    echo("</font>
가<br>
<br>");
    echo("
    <br>
        e-mail
            .</center>");
}
if($mode=='2')
{
    $data = explode("=", $network);
    $network = $data[0];
    $username = $data[1];
    echo("
        <FORM name='form'
method='post' action='$PHP_SELF'>
        <CENTER><BR>
        <TABLE border='0' width='80%'>
            <tr>
                <td align='center'>

(n)      (edges)

            </td>
        </tr>
    </FORM>
    </CENTER>
    </TABLE border='0' width='80%'>
    <tr><td height='30'> &nbsp;</td></tr>
    <tr>
        <td align='center'># of
edges = <input type='text'
name='edges' size=3
maxlength=5></td></tr><tr>
        <td align='center'># of
nodes = <input type='text'
name='node' size=3
maxlength=5></td>
    </tr>
</TABLE><BR>
<TABLE><BR>
    <input type='submit'
name='submit' value='    ' >
    <input type='hidden'
name='mode' value='complex1'>
    <input type='hidden'
name='network' value='$network'>
    <input type='hidden'
name='username' value='$username'>
</TABLE>
</CENTER>
</FORM>
");
}
if($mode == 'complex1')
{
    echo(" <FORM name='form'
method='post' action='$PHP_SELF'>
    <CENTER><BR>
    <TABLE border='0' width='80%'>

```



```

    ");
echo("
    </TABLE>
    <TABLE><BR>
    <input type='submit'
name='submit' value='    ' >
    <input type='hidden'
name='mde' value='complex4'>
    <input type='hidden'
name='node' value='$node'>
    <input type='hidden'
name='edges' value='$edges'>
    <input type='hidden'
name='network' value='$network'>
    <input type='hidden'
name='username' value='$username'>
    ");

for ($i=1;$i<=($node*$node);$i++) {
    echo("<input type='hidden'
name='e[$i]' value='Se[$i]'>");
}
for ($i=1;$i<=$node;$i++) {
    echo("<input type='hidden'
name='r[$i]' value='Sr[$i]'>");
}
echo("
    </TABLE>

    </CENTER>
    </FORM>
");
}

if($mode == 'complex4')
{
    echo(" <FORM name='form'
method='post' action='$PHP_SELF'>
        <CENTER><BR>
        <table border='0'
width='80%'>
        <tr>
            <td align='center' >
                .</td>
        </tr>
        </table><br>
        <TABLE border='1' width='80%'>
            ");
    for ($i=1, $p=1, $count=1; $i <= ($m+1)
; $i++) {
        echo("<tr>");
        if ($i=='1'){
            echo("<td &nbsp; </td>");
            for ($j=1;$j<=$node;$j++) {
                echo("<td> $j </td>");
            }
            echo("<td> RSH </td>");
            echo("</tr>");
        } else {
            echo("<tr>");
            echo("<td>          $p </td>");
            $p++;
            for ($j=1;$j<=$node+1; $j++) {
                echo("<td> <input type='text'
name='c[$count]' size=3

```



```

echo("<td>          </td>");
for ($i=1;$i<=$node;$i++)
    echo("<td> $r [$i] </td>");
echo("<td> &nbsp; </td></tr>");

for ($i=1, $pos=1, $step=$node, $step1=1;$
i<=$m;$i++){
    echo("<tr>");
    echo("<td>          $i </td>");

for ($j=$pos;$j<=($step+$step1);$j++, $p
os++) {
    echo("<td> $c [$j] </td> ");
}
echo("</tr>");
$step = $step+$node;
$step1++;
}
echo(" </center>
</table> ");

echo(" <center><br>
<TABLE border='0' width='80%'>
<tr>
<td>
Incidence matrix.
</td>
</tr>
</TABLE><br>
<table border='1' width='80%'>
");

for ($i=1, $p=1, $count=1;$i<=$node+1;$i+
+) {
    echo("<tr>");
    if ($i=='1'){
        echo("<td> &nbsp; </td>");
        for ($j=1;$j<=$node;$j++) {
            echo("<td> $j </td>");
        }
        echo("</tr>");
    } else {
        echo("<tr>");
        echo("<td> $p </td>");
        $p++;

for ($j=$count;$j<=($node+$count-1);$j+
+) {
            echo("<td> $e [$j] </td>");
        }
        $count=$count+$node;
        echo("</tr>");
    }
}

$que=" insert into system_data1
values ('', '$username', '$network', '$nod
e', '$edges', '$m', '$r [1]', '$r [2]', '$r [3]
', '$r [4]', '$r [5]', '$r [6]', '$r [7]', '$r
[8]', '$r [9]', '$r [10]', '$r [11]', '$r [12]
', '$r [13]', '$r [14]', '$r [15]', '$r [16]',
'$r [17]', '$r [18]', '$r [19]', '$r [20]', '$
r [21]', '$r [22]', '$r [23]', '$r [24]', '$r [
25]', '$r [26]', '$r [27]', '$r [28]', '$r [29]
', '$r [30]', '$c [1]', '$c [2]', '$c [3]', '$
c [4]', '$c [5]', '$c [6]', '$c [7]', '$c [8]',

```



```

}
if( $mode == 'edit2')
{
    $data = explode("=", $temp);
    $username = $data[0];
    $network = $data[1];
    $node = $data[2];
    $edges = $data[3];
    $m = $data[4];

    for ($i=5, $j=1; $i<=(4+$node); $i++, $j++)
    {
        $r[$j] = $data[$i];
    }
    for ($k=(5+$node), $h=0, $j=1; $h<$m;
    $h++){
        for ($i=$k; $i<=($k+$node); $i++) {
            $c[$j] = $data[$i];
            $j++;
        }
        $k=$k+$node+1;
    }

    for ($h=1, $k=(5+$node+($node*$m+$m)), $j
    =1; $h<=$node; $h++){
        for ($i=$k; $i<=($k+$node); $i++) {
            $e[$j] = $data[$i];
            $j++;
        }
        $k=$k+$node+1;
    }
    $que="delete from system_data1 where
    name='$username' and
                                system='$network'";
    $result=mysql_query($que, $connect);
    if($result)
    {
    }
    else
    {
        echo(" <script>
                window.alert('DB 7t
                .')
                history.go(-1)
                </script>
                ");
        exit;
    }
    echo("
        <CENTER>
        <table width='80%'><tr><td
        align='center'>
        ");
        $network1 = "
        ";
        echo(" $network1");
        echo("
                :
                </td></tr></table>
                </CENTER>
                <BR><BR>
                <form name='form'
                method='post' action='$PHP_SELF'>
                <CENTER>
                <TABLE width='80%' border='1'>

```

```

        <tr>
            <td> &nbsp; </td>
        </td>
    </tr>
    ");
    for ($i=1;$i<=$node;$i++)
        echo("<td> $i </td>");
    echo("<td> RHS </td></tr><tr>");
    echo("<td> </td>");
    for ($i=1;$i<=$node;$i++)
        echo("<td><input type='text '
name='r[$i]' size=3 maxlength=5
value='$r[$i]'> </td>");
    echo("<td> &nbsp; </td></tr>");

    for ($i=1,$pos=1,$step=$node,$step1=1;$
i<=$m;$i++){
        echo("<tr>");
        echo("<td> $i </td>");

        for ($j=$pos;$j<=($step+$step1);$j++,$p
os++){
            echo("<td><input type='text '
name='c[$j]' size=3 maxlength=5
value='$c[$j]'> </td> ");
        }
        echo("</tr>");
        $step = $step+$node;
        $step1++;
    }
    echo(" </center>
</table> ");
    echo(" <center><br>
<table border='0' width='80%'>
    <tr>
        <td>
            Incidence matrix.
        </td>
    </tr>
</table><br>
<table border='1' width='80%'>
    ");
    for ($i=1,$p=1,$count=1;$i<=$node+1;$i+
+) {
        echo("<tr>");
        if($i=='1'){
            echo("<td> &nbsp; </td>");
            for ($j=1;$j<=$node;$j++) {
                echo("<td> $j </td>");
            }
            echo("</tr>");
        } else {
            echo("<tr>");
            echo("<td> $p </td>");
            $p++;
        }
        for ($j=$count;$j<=($node+$count-1);$j+
+) {
            echo("<td><input type='text '
name='e[$j]' size=3 maxlength=5
value='$e[$j]'> </td>");
        }
        $count=$count+$node;
        echo("</tr>");
    }
    echo("

```

```

        </TABLE>
        <TABLE><BR>
        <input type='submit '
name='submit' value='      ' >
        <input type='hidden '
name='mode' value='complex5'>
        <input type='hidden '
name='node' value='$node'>
        <input type='hidden '
name='edges' value='$edges'>
        <input type='hidden' name='m'
value='$m'>
        <input type='hidden'
name='network' value='$network'>
        <input type='hidden'
name='username' value='$username'>
    ");
    echo("
        </TABLE>
        </CENTER>
        </FORM>
    ");
}
if( $mode == 'edit1')
{
    $data = explode("=", $temp);
    $username = $data[0];
    $network = $data[1];
    $node = $data[2];
    $m = $data[3];

    for ($i=4, $j=1; $i<=(3+$node); $i++, $j++)
    {
        $r[$j] = $data[$i];
    }
    for ($k=(4+$node), $h=0, $j=1; $h<$m ;
    $h++){
        for ($i=$k; $i<=( $k+$node+$h ); $i++)
        {
            $c[$j] = $data[$i];
            $j++;
        }
        $k=$k+$node+1;
    }
    $que="delete from system_data1 where
name='$username' and
    system='$network'";
    $result=mysql_query($que, $connect);
    if($result)
    {
    }
    else
    {
        echo(" <script>
            window.alert('DB      가
                .')
            history.go(-1)
        </script>
        ");
        exit;
    }
    echo("
        <CENTER>
        <table width='80%'><tr><td
align='center'>
        ");

```


C

```

/*
 * genetic.h
 */
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define TRUE 1
#define FALSE 0
/*
 *
 */
#define PCP 100
/* node */
#define VAR 15
/* digit */
#define DIGIT 45
/*
 */
#define M2
#ifndef EXIERN
#define EXIERN extern
#endif
typedef struct {
    int binary[DIGIT];
    int decimal[VAR];
    int actual_count;
    float fitness;
    float expected_count;
} DATA;
EXIERN DATA population[PCP];
EXIERN DATA Temp[PCP];
EXIERN float Max_value;
EXIERN int RUN;
EXIERN int Re_pop;
EXIERN int C[M][VAR];
EXIERN int R[S][M];

EXIERN float R[VAR];
EXIERN int BIT[VAR];
/*
 * main.c
 */
#define EXIERN
#include "genetic.h"
int main(int argc, char *argv[])
{
    int i;
    if( argc < 2 ) {
        printf("Type the number of
            generation \n");
        printf(" EX) #./genetic 100
            \n");
        exit(0);
    }

    RUN = atoi(argv[1]);
    printf("# of GA Step :
        %d\n",RUN);
    Read_data();
    Re_pop = -1;
    while( Re_pop == -1){
        Initial_pop();
    }
    for(i=0;i<RUN;i++) {
        Repr oduction(i+1);
        Grossover ();

        if(Finish_check()==TRUE) {
            printf("Finish check
                true\n");
        }
    }
    printf("***

```

```

        **\n\n");
printf("[ ");
    for(i=0; i<VAR; i++)
printf("%d",
    population[0].decimal[i]);
printf("\t F(X) = %f \n",
    population[0].fitness);
Reoptima();

printf("\n\n***SA

        **\n\n");
printf("[ ");
for(i=0; i<VAR; i++)
printf("%d",
    population[0].decimal[i]);
printf("\t F(X) = %f \n",
    population[0].fitness);
exit(0);
}
}
printf("\n**

        **\n\n");
printf("[ ");
for(i=0; i<VAR; i++)
printf("%d",
    population[0].decimal[i]);
printf("\t F(X) = %f \n",
    population[0].fitness);

Reoptima();
printf("\n\n***SA

        **\n\n");
printf("[ ");
for(i=0; i<VAR; i++)
printf("%d",

population[0].decimal[i]);
printf("\t F(X) = %f \n",
    population[0].fitness);

return 0;
}
/*
 * crossover.c
 */

#include "genetic.h"
void Crossover()
{
int mate[100],k1,k2,i,j,state,v,inx,i,k;

for(i=0;i<PCP;i++)
    mate[i]=-1;
k1=rand()%PCP;
k2=rand()%PCP;
if(k1==k2)
    k2=(k2+1)%PCP;
mate[k1]=k2;
mate[k2]=k1;
state=1;
i=0;
while(state<(PCP/2)) {
    while(1) {
        if(mate[i]==-1)
            break;
        i++;
    }
    k1=rand()%PCP;

while(1) {
    if((mate[k1]==-1)&&(k1!=i))
        break;
    else

```

```

        k1=(++k1)%PCP;
    }

    mate[i]=k1;
    mate[k1]=i;
    state++;
}
for (v=0;v < PCP;v++)
    if(mate[v] != -1) {
        k=rand()%DIGIT;
        for(j=k;j < DIGIT;j++) {

            imsi=population[v].binary[j];

population[v].binary[j]=population[mate[v]
    ].binary[j];

population[mate[v]].binary[j]=imsi;
        }
        mate[mate[v]]=1;
        mate[v]=1;
    }
}
/*
 * finish_check.c
 */
#include "genetic.h"
int Finish_check()
{
    int i,j,x[VAR],xx[VAR];

    for(i=0;i<VAR;i++)
x[i]=population[0].decimal[i];
    for(i=1;i < PCP;i++){
        for(j=0;j<VAR;j++){
            if(x[j] ==

population[i].decimal[j])
                continue;
            else
                return FALSE;
        }
        return TRUE;
    }
/*
 * function_value.c
 */
#include "genetic.h"
void Function_value( int i )
{
    float x1[VAR],tmp2;
    float f=1.0,y[VAR];
    int tmp[M,j],x[VAR],k,flag ;
    for(j=0;j<VAR;j++) {

x1[j]=(float)population[i].decimal[j];
        x[j] = population[i].decimal[j];
    }

    for(j=0;j<VAR;j++)
        y[j]=pow(1.0-R[j],x1[j]);

    for(j=0;j<VAR;j++)
        f*=(1-y[j]);

    for(j=0;j<Mj++){
        tmp[j]=0;
        for(k=0;k<VAR;k++)
            tmp[j]+=C[j][k]*x[k];
    }

    flag=0;
    for(j=0;j<Mj++)

```

```

    if(tmp[j] <= RHS[j])
        flag++;
    if(flag == M)
        population[i].fitness = f;
    else {
        tmp2=f-0.5;
        if(tmp2 < 0.0)
            population[i].fitness =
                (tmp2*-1);
        else
            population[i].fitness = tmp2;
    }
}
/*
 * initial_pop.c
 */
#include "genetic.h"
#include <sys/types.h>
void Initial_pop()
{
    int i,j, it1, it2, in; i=0, power, k, pos1, pos2;
    float sum=0.0;
    srand((unsigned int)time((time_t
    *)NULL));
    for(j=0;j < PCP; j++)
        for(k=0;k < DIGIT;k++)
            population[j].binary[k]=rand()%2;
    for(i=0;i<PCP; i++)
        for(j =0, pos1=0;j <VAR;j++){
            pos1+=BIT[j];
            population[i].binary[pos1-1]=1;
            population[i].decimal[j]=0;
        }
    for(i=0;i<PCP;i++){
        Function_value(i);
        for(i=0;i<PCP;i++){
            sum += population[i].fitness;
        }
        if( sum > (0.2*PCP))
            Re_pop = 1;
            Max_value=0.0;
    }
}
/*
 * read_data.c
 */
#include "genetic.h"
void Read_data()
{
    int C_tmp[M][VAR]={{5,4,9,7,7,5,6,9,
    4,5,6,7,9,8,6}, {8,9,6,7,8,8,9,6,7,8,9,7,6,
    5,7}};
    int RHS_tmp[M]={400,414};
}

```

```

float R_tmp[VAR]={0.9,0.75,0.65,
0.8,0.85,0.93,0.78,0.66,0.78,0.91,0.79,0.7
7,0.67,0.79,0.67};
int BIT_tmp[VAR]={3,3,3,3,3,3,3,3,3,
3,3,3,3,3,3};
int i,j;
for (i=0;i<Mi++){
RHS[i]=RHS_tmp[i];
for (j=0;j<VAR;j++){
C[i][j]=C_tmp[i][j];
BIT[j]=BIT_tmp[j];
R[j]=R_tmp[j];
}
}
}
/*
* reoptima.c
*/
#include "genetic.h"
#include <sys/types.h>
void Reoptima()
{
int imsi[VAR],small[M,i,j],
loop = 0, s1, s2, s3, s4;
int diff[M],Sum[M],flag;
float imsi_value;
srand((unsigned int)time((time_t
*)NULL));
while( loop++ < 1000)
{
for(i=0; i < VAR ; i++)
imsi[i] =
population[0].decimal[i];
imsi_value =
population[0].fitness;

s1 = rand()%VAR;

s2 = rand()%2;
if( (s2==0)&&
(population[0].decimal[s1] > 1) )
--population[0].decimal[s1];
else
++population[0].decimal[s1];

s3 = rand()%VAR;
if(s1 == s3)
s3 = (s3+1)%VAR;
s4 = rand()%2;
if((s4==0)&&
(population[0].decimal[s3] > 1))
--population[0].decimal[s3];
else
++population[0].decimal[s3];
Function_value(0);
if( population[0].fitness <
imsi_value ) {
for(i=0; i<VAR; i++)
population[0].decimal[i] = imsi[i];
population[0].fitness = imsi_value;
}

while(TRUE)
{
for(i=0;i<Mi++){
Sum[i]=0;
for(j=0; j<VAR; j++)
Sum[i]+=(population[0].decimal[j]*C[i][j])
;
diff[i]=RHS[i]-Sum[i];
}
flag=0;
}
}
}

```

```

for (i=0;i<M;i++){
    small[i]=Sort_M(i);
    if( diff[i] >= small[i] )
        flag++;
}

if( flag == M)
{
    for (i=0; i<VAR; i++)
        insi[i] =
population[0].decimal[i];
    for (i=0; i<VAR; i++)
    {
        flag=0;
        for (j=0;j<M;j++)
            if(C[j][i] <=
                diff[j])
                flag++;

        if( flag == M)
        {
            insi[i]++;
            for (j=0;j<M;j++)
                diff[j]=diff[j]-C[j][i];

            for (j=0; j<VAR; j++)
                population[0].decimal[j]=insi[j];
        }
    }
} else
    break;
}

/*
* reproduction.c
*/
#include "genetic.h"
void Reproduction(int v)
{
    int i,count=0,j,k=0;
    float sum=0.0,average;
    for (i=0;i < PCP;i++){
        sum+=population[i].fitness;
        population[i].actual_count=0;
    }
    average=sum/((float)PCP);

    for (i=0;i < PCP;i++) {
        population[i].expected_count=population[i]
        .fitness/((float)average);
    }
    Sort_pop();
    for (i=0;i < PCP; i++) {
        if(count < PCP){
            population[i].actual_count=(int)(populatio
            n[i].expected_count+0.5);
            count+=population[i].actual_count;
        }else{
            population[i].actual_count=0;
        }
    }
    if(count < PCP)
        for (i=0;(i<PCP)&&(count <=
        PCP);i++)
            if(population[i].actual_count
            == 0)
                population[i].actual_count=1;
}

```

```

        count++;
    }
for (i=0, count=0; (i<PCP)&&(count<PCP); i++)
    if (population[i].actual_count >
        0){
        count += population[i].actual_count;
for (j =0; j < population[i].actual_count; j++)
        Temp[k++] = population[i];
    }
for (i=0; i < PCP; i++)
    population[i] = Temp[i];
}
/*
 * sort_m c
 */
#include "genetic.h"
int Sort_M (int i )
{
    int insi [VAR], j, k, sw, temp;

for (j =0; j < VAR; j++)
    insi [j] = C[i] [j];
for (j =(VAR- 1); j >0; j --){
    sw=1;
for (k=0; k<j; k++)
    if (insi [k] > insi [k+1]){
        temp=insi [k];
        insi [k]=insi [k+1];
        insi [k+1]=temp;
        sw=0;
    }
    if (sw==1)
        break;
}
return(insi [0]);
}

/*
 * sort_pop.c
 */
#include "genetic.h"
void Sort_pop( )
{
    int i, j, sw=1;
for (i=(PCP- 1); i>0; i-- )
    {
        for (j =0; j < i; j++)
            if (population[j].expected_count <
                population[j +1].expected_count )
                {
                    Temp[0]=population[j];
                    population[j] = population[j +1];
                    population[j +1]=Temp[0];
                    sw=0;
                }
            if (sw ==1)
                break;
        if (population[0].fitness >
            Max_value )
            Max_value = population[0].fitness;
    }
}

```

가 ,