An Ant System Approach to Redundancy Allocation

Yun-Chia Liang
Department of Industrial Engineering,
University of Pittsburgh, Pittsburgh, PA 15261
Email: yulst8@pitt.edu

Alice E. Smith
Department of Industrial Engineering,
University of Pittsburgh, Pittsburgh, PA 15261
Email: aesmith@engrng.pitt.edu

Abstract - This paper solves the redundancy allocation problem of a series-parallel system by developing and demonstrating a problem-specific Ant System. The problem is to select components and redundancy-levels to maximize system reliability, given system-level constraints on cost and weight. The Ant System algorithm presented in this paper is combined with an adaptive penalty method to deal with the highly constrained problem. An elitist strategy and mutation are introduced to our AS algorithm. The elitist strategy enhances the magnitude of the trails of good selections of components. The mutated ants can help explore new search areas. Experiments were conducted on a well-known set of sample problems proposed by Fyffe, Hines, and Lee.

1. Introduction

This paper describes the use of an Ant System (AS) to solve the redundancy allocation problem for a series-parallel system. The reliable performance of a system for a predefined mission time under various conditions is very important in many industrial and military applications. The series-parallel system with k-out-of-n:G subsystem redundancy, addressed in this paper, is a common representation for many system design problems.

In the formulation of a series-parallel system problem, for each subsystem, multiple component choices (assuming an unlimited supply of each component) are used in parallel. For those systems designed using off-the-shelf component types, with known cost, reliability, and weight, system design and component selection becomes a combinatorial optimization problem. The problem is then to select the optimal combination of parts and redundancy levels to meet cost and weight constraints collectively while maximizing system reliability.

AS is one of the adaptive meta-heuristic optimization methods which include simulated annealing (SA), genetic algorithms (GA), evolutionary strategies (ES), and Tabu Search (TS). AS has been inspired by the behavior of real ants. Ethologists studied how blind animals, such as ants, could establish shortest paths from their nest to food sources. The medium that is used to communicate information among individual ants regarding paths consists of pheromone trails. A moving ant lays some pheromone on the ground, thus marking the path by a pheromone trail.

If an isolated ant moves randomly, it will detect a previously laid trail and decide where to go. The trail with more pheromone has a higher probability to be chosen by the following ants. The process is thus characterized by a positive feedback loop, where the probability with which an ant chooses a path increases with the number of ants that previously chose that same path.

1.1 The System Reliability Problem: Literature Review

Tillman, Hwang, and Kuo [1] provide a thorough review related to optimal system reliability with redundancy. They divided optimal system reliability models with redundancy into series, parallel, series-parallel, parallel-series, standby, and complex classes. They also categorized optimization methods into integer programming, dynamic programming, linear programming, geometric programming, generalized Lagrangian functions, and heuristic approaches. The authors concluded that many algorithms have been proposed but only a few have been demonstrated to be effective when applied to large-scale nonlinear programming problems. Also, none has proven to be generally superior.

Fyffe, Hines, and Lee provide a dynamic programming algorithm for solving the system reliability allocation problem [2]. As the number of constraints in a given reliability problem increases, the computation required for solving the problem increases exponentially. In order to overcome these computational difficulties, the authors introduce the Lagrange multiplier to reduce the dimensionality of the problem. To illustrate their computational procedure, the authors use a hypothetical system reliability allocation problem, which consists of fourteen functional units connected in series. While their formulation provides a selection of components, the search space is restricted to consider only solutions where the same component type is used in parallel. Nakagawa and Miyazaki [3] show a more efficient algorithm compared to dynamic programming using the Lagrange multiplier. In their algorithm, the authors use surrogate constraints obtained by combining multiple constraints into one constraint. In order to demonstrate efficiency of the new algorithm, they also solve 33 variations of the Fyffe problem. Of the 33 problems, their N&M algorithm produces optimal solutions for 30 of them. Misra and Sharma [4] present a simple and efficient technique for solving integer programming problems such as the system reliability design problem. The algorithm is based on

function evaluations and a search limited to the boundary of resources.

In the nonlinear programming approach, Hwang, Tillman and Kuo [5] use the generalized Lagrangian function method and the generalized reduced gradient method to solve nonlinear optimization problems for reliability of a complex system. They first maximize complex-system reliability with a tangent cost-function and then minimize the cost with a minimum system reliability. The same authors also present a mixed integer programming approach to solve the reliability problem [6]. They maximize the system reliability as a function of component reliability level and the number of components at each stage.

Using a genetic algorithm (GA) approach, Coit and Smith [7, 8, 9] provide a competitive and robust algorithm to solve the system reliability problem. The authors use a penalty guided algorithm which searches over feasible and infeasible regions to identify a final, feasible optimal, or near optimal, solution. The penalty function is adaptive and responds to the search history. The GA performs very well on two types of problems: redundancy allocation as originally proposed by Fyffe, et al., and randomly generated problems with more complex configurations. For a fixed design configuration and known incremental decreases in component failure rates and their associated costs, Painton and Campbell [10] also use a GA to find a maximum reliability solution to satisfy specific cost constraints. They formulate a flexible algorithm to optimize the 5th percentile of the mean time-between-failure distribution.

1.2 Ant System: Literature Review

Inspired by the collective behavior of a real ant colony, Marco Dorigo first introduced the Ant System (AS) in his Ph.D. thesis (1992), and the study was further continued by Dorrigo, Maniezzo, and Colorni [11, 12]. The characteristics of an artificial ant colony include positive feedback, distributed computation, and the use of a constructive greedy heuristic. Positive feedback accounts for rapid discovery of good solutions, distributed computation avoids premature convergence, and the greedy heuristic helps find acceptable solutions in the early stages of the search process. In order to demonstrate the AS approach, the authors apply this approach to the classical TSP, asymmetric TSP, QAP, and job-shop scheduling. AS shows very good results in each applied area. More recently Dorigo and Gambardella have been working on extended versions of the AS paradigm. Ant Colony System (ACS) is one of the extensions and has been applied to the symmetric and asymmetric TSP with excellent results [13, 14]. The Ant System has also been applied with success to other combinatorial optimization problems such as the vehicle routing problem [15], telecommunications networks [16], scheduling, graph coloring, partitioning, etc.

2. Methodology

2.1 Problem Definition

Notation

R	overall reliability of the series-parallel system
C	cost constraint
W	weight constraint
S	size of search space
s	number of subsystems
i	index for subsystems $(i = 1,...,s)$
a_i	number of available component choices for subsystem i
r_{ij}, c_{ij}, w_{ij}	reliability, cost, and weight of component j available for subsystem i
y_{ij}	quantity of component j used in subsystem i
$\mathbf{y}_i$	$(y_{i1},...,y_{ia_i})$
p_i	$= \sum_{j=1}^{a_i} y_{ij}$, total number of components used in subsystem i
$p_{\max}$	maximum number of components in parallel (user assigned)
k_i	minimum number of components in parallel required for subsystem i to function
$\mathbf{k}$	$(k_1,...,k_s)$
$R_i(\mathbf{y}_i \mid k_i)$	reliability of subsystem i, given k_i
$C_i(\mathbf{y}_i)$	total cost of subsystem i
$W_i(\mathbf{y}_i)$	total weight of subsystem i

Assumptions

- The states of components and the system have only two options - good or bad.
- If the number of good components is less than k_i in a subsystem i, then the subsystem i fails.
- The failure of any subsystem will cause system failure.
- Failures of components are independent events.
- Failed components do not damage the system, and are not repaired.
- The failure rates of components when not in use are the same as when in use (i.e. active redundancy).
- Component reliabilities are known and deterministic.
- The supply of components is unlimited (i.e. off-the-shelf).

The detailed allocation method of system reliability discussed here is designed to select the optimal solution in

the context of a trade-off analysis. Given the overall restrictions on system cost of C and weight of W, the problem is to determine which design alternative to select with the specified level of component reliability, and how many redundant components to use in order to achieve the maximum reliability. This formulation of the redundancy allocation problem for series-parallel system, such as depicted in Fig. 1 leads to the maximization of system reliability R given by the product of subsystem reliabilities

$$R = \prod_{i=1}^{s} R_i(\mathbf{y}_i \mid k_i)$$

subject to the constraints

$$\sum_{i=1}^{s} C_i(\mathbf{y}_i) \leq C, \quad \sum_{i=1}^{s} W_i(\mathbf{y}_i) \leq W$$

$$k_i \leq \sum_{j=1}^{a_i} y_{ij} \leq p_{\max} \quad \forall i = 1,2,...,s$$
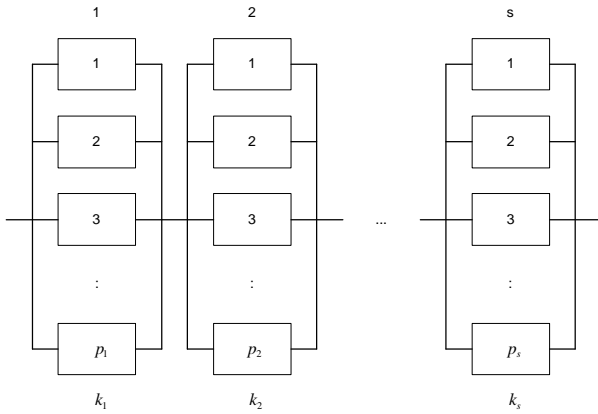

Figure 1. Series-parallel system configuration.

Within this formulation, system weight and cost are often defined as linear functions because it is a reasonable representation of the cumulative component cost and weight. Assuming an upper bound on the maximum number of redundant components in parallel, $p_{\max}$, the number of unique system representations is [17]:

$$S = \prod_{i=1}^{s} [\binom{a_i + p_{\max}}{a_i} - \binom{a_i + k_i - 1}{a_i})]$$

The size of the search space (S) is very large for the series-parallel reliability problem even though the problem size is small or moderate.

2.2 Adaptive Penalty Function for Combinatorial Reliability Design
The penalty function used employs the notion of a "Near-Feasibility Threshold" (NFT) for each constraint [7, 8, 9].

The NFT is the threshold distance from the feasible region that is considered as being close to feasibility. The redundancy allocation problem is formulated with two independent constraints (cost and weight) so the penalty function became a linear summation as follows,

$$R_{kp} = R_k - (R_{all} - R_{feas})((\frac{\Delta w_k}{NFT_w})^g + (\frac{\Delta c_k}{NFT_c})^g).$$ R_{kp} is the penalized objective function value of solution k, R_k is the unpenalized objective function value of solution k; i.e.

$$R_k = \prod_{i=1}^{s} R_i(\mathbf{y}_i \mid k_i) \qquad \text{subject} \qquad \text{to} \qquad k_i \leq \sum_{j=1}^{a_i} y_{ij} \leq p_{\max}$$

$\forall i = 1,2,...,s$, R_{all} represents the unpenalized value of the best solution found, and R_{feas} denotes the value of the best feasible solution found. The exponent g is a preset severity parameter and its value is set to 2 in this paper. NFT_w and NFT_c are the "Near-Feasible Thresholds" for the weight and cost constraints, respectively, and take dynamic formations $NFT = \frac{NFT_0}{1 + \mathsf{I} \times NC}$ where NFT_0 is a starting point for NFT, NC represents the number of iterations, and I denotes a constant which assures that the entire region between NFT_0 and zero is searched. Δw_k and Δc_k represent the magnitude of any constraint violations for the kth solution.

2.3 Ant System Algorithm for the Redundancy Allocation Problem
Each possible solution to the redundancy allocation problem is a collection of p_i parts in parallel ($k_i \leq p_i \leq p_{\max}$) for s different subsystems. The p_i parts can be chosen in any combination from the a_i available components. The a_i components are indexed in descending order in accordance with their reliability; i.e. 1 represents the most reliable component, etc. An index of $a_i + 1$ is assigned to a position where an additional component was not used (left blank). Each of the s subsystems is represented by $p_{\max}$ positions with each component listed according to their reliability index.

The parameters considered in the Ant System Algorithm are shown below:

- a : the relative importance of the trail, $a \geq 0$;
- r : trail persistence, $0 \leq r < 1$;
- Q : a constant related to the magnitude of the trail laid by ants;
- c : initial quantity of pheromone substance for the "blank", $0 \leq c < 1$.

In an Ant System, each ant represents one design of a whole system. In the beginning of the process, an initialization phase takes place during which ants select components in

each subsystem according to the transition probability $P_{ij} = \dfrac{[t_{ij}]^a}{\sum\limits_{l=1}^{a_i}[t_{il}]^a}$ which is based on the trail intensity (t_{ij}).

After all of the components for the ants are selected, the unpenalized reliability R_k and the penalized reliability R_{kp} for the system are calculated. Thereafter, the trail is updated by $t_{ij}(new) = r\,t_{ij}(old) + \sum\limits_{k=1}^{NA}\Delta t_{ij}^k$ where Δt_{ij}^k is the quantity of trail substance laid for combination (i, j) by the kth ant. The magnitude of the new pheromone trails was updated by vaporizing $(1-r)$ percentage of pheromone from previous iteration and accumulating pheromone laid by the ants in current iteration. Also, the best feasible solution found is saved. Starting from the second iteration, a certain amount of mutated ants are generated by following the initial transition probability, and a certain amount of the best feasible ant found are duplicated. The use of mutated ants can help prevent the search from stagnating in a local optimum. The purpose of duplicating the best feasible solution is to enhance the magnitude of trail for "good" combinations. This process continues until the maximum (user-defined) number of cycles, NC_{max}, is reached.

Formally the Any System algorithm is:
1. Initialization
Decide on the number of alternative components in each subsystem (a_i) from the input data set. Update $a_i = a_i + 1$ by considering the "blank" (no component) choice.

For each combination (i, j) where $j = 1,2,...,a_i$, assign an initial trail value $t_{ij} = \dfrac{1-c}{a_i - 1}$ for all alternative components, $t_{ij} = c$ for the "blank" case, and $\Delta t_{ij} = 0$ for all cases.

For $k = 1$ to m (# of ants: each ant represents a system)
  For $i = 1$ to s (# of subsystems)
    For $l = 1$ to p_{max}
    (max # of components in parallel in each subsystem)

    Generate a random number ($rand$) between [0,1]

    Set $count = 0$
    (counter for the # of "non-blank" alternative components used)

Choose a component, including "blanks", with transition probability $P_{ij} = \dfrac{[t_{ij}]^a}{\sum\limits_{l=1}^{a_i}[t_{il}]^a}$.

Repeat the innermost loop until $count \geq k_i$
(Check the feasibility of the minimum number of components in parallel required for subsystem i to function)

Calculate R_k (unpenalized system reliability for each ant)
Update R_{all}, R_{feas} and component selection of R_{feas}

2. Set $NC = 1$ (# of iterations)

Update the dynamic "Near-Feasibility Threshold"
$NFT = \dfrac{NFT_0}{1 + I \times NC}$
Calculate R_{kp} (penalized system reliability for each ant)

For every combination (i, j),
(i = subsystem index, j = component type index.)
  For $k = 1$ to m
    $\Delta t_{ij}^k = Q \times R_{kp}$ if $(i, j) \in$ combinations used by the kth ant and $R_{kp} > 0$;
        0 otherwise

    $\Delta t_{ij} = \Delta t_{ij} + \Delta t_{ij}^k$

  Update trail values $t_{ij}(new) = r\,t_{ij}(old) + \Delta t_{ij}$

  Update the transition probability $P_{ij} = \dfrac{[t_{ij}]^a}{\sum\limits_{l=1}^{a_i}[t_{il}]^a}$ for all possible selections l

  Duplicate R_{feas} and component selection of R_{feas} as t new ants for next population

For $k = 1$ to n
(# of mutated ants: each ant will represent a system by using the initial transition probability)
  For $i = 1$ to s (# of subsystems)
    For $l = 1$ to p_{max}
    (max # of components in parallel in each subsystem)

Set $count = 0$
(counter for the # of "non-blank" alternative components used)

Choose a component, including "blanks", with initial transition probability P_{ij}.

Repeat the innermost loop until $count \geq k_i$
(Check the feasibility of minimum number of components in parallel required for subsystem i to function)

For $k = n+1$ to $m - t$
(# of new ants: each ant will represent a system by the updated transition probability)
    For $i = 1$ to s (# of subsystems)
        For $l = 1$ to p_{max}
(max # of components in parallel in each subsystem)

Set $count = 0$
(counter for the # of "non-blank" alternative components used)

Choose a component, including "blanks", with updated transition probability P_{ij}.

Repeat the innermost loop until $count \geq k_i$
(Check the feasibility of minimum number of components in parallel required for subsystem i to function)

Calculate R_k (unpenalized system reliability for each ant)

Update R_{all}, R_{feas} and component selection of R_{feas}

3. Set $NC = NC + 1$
  For every combination (i, j), set $\Delta \tau_{ij} = 0$.

4. If ($NC < NC_{max}$)   (reach max # of iterations)
  Then
      Goto step2
  Else
  Print largest feasible reliability and component selection
  Stop

3. Test Problems and Results

3.1 Test Problems
The best known problem of a series-parallel system, Fyffe, Hines, and Lee [2], specifies 130 units of system cost, 170 units of system weight and $k_i = 1$; i.e., 1-out-of-n:G subsystems. In the optimal solution, the weight constraint is active and the cost of system is 119. In this paper, we implemented the Ant System on the 33 variations of the Fyffe, Hines, and Lee problem [2] that were devised by Nakagawa and Miyazaki [3]. In the set of problems, $C = 130$ and W is increased incrementally from 159 to 191. In both papers, the approach required is that only identical components can be placed in redundancy. Different component types will be allowed to reside in parallel as was done by Coit and Smith [7, 8, 9]. Assume that the value of k_i is 1 and the value of p_{max} is 8 for all subsystems. Considering component mixing, the search space size is larger than 7.6×10^{33}. The necessary input data for this problem is summarized in Table 1.

3.2 Computational Results
For the decreasing NFT implementation in this paper, two different λ values (0.04, 0.4) were used, NFT_{c0} was set to 100 and NFT_{w0} was set to $\dfrac{W}{1.3}$. The number of ants, NA, was set equal to 1000. An alternative local updating method $\tau_{ij}(new) = \rho \tau_{ij}(old) + (1-\rho) \sum_{k=1}^{NA} \Delta \tau_{ij}^k$ was also tested but not shown to influence the results. In order to show the effect of duplicating the best feasible solution, two different values of t, 1 (i.e. no duplication) and 20, were examined. The default values of the parameters were $c = 0.5$, $a = 1$, $\rho = 0.85$, and $Q = 100$. Three different Q values (1, 100, 10000) were tested but were not shown to influence the algorithm. Three different a values (0, 1, 2) were tested and the results of $a = 1$ were superior. Also, four different c values (0.5, 0.65, 0.7, 0.85), five different ρ values (0.3, 0.5, 0.7, 0.85, 0.9), and three different number of mutated ants (20, 50, 100) were tested. The value of parameters, (c, ρ, λ) equal to (0.5, 0.7, 0.4) with 20 duplicated best feasible ants and 50 mutated ants produced better results. Ten different random number seeds were used for each of the 33 problem variations, and 1500 iterations were used in each run. The computational results of the 33 variations are summarized in Table 2 and also compared with the results from Nakagawa and Miyazaki [3] and from Coit and Smith [8]. The AS results provided good alternatives to the test problems and low variation among the 10 trials.

4. Conclusions

The Ant System is a promising heuristic method for solving combinatorial optimization problems. The efficiency of this algorithm will depend on the selection of parameters, trail update method, and other supplemental devices. In this paper, we show how to apply the AS to the redundancy allocation problem. Unlike the original Ant System, we introduce an elitist strategy and mutation to our AS algorithm. The elitist strategy enhances the magnitude of

the trails of good selections of components. The mutated ants can help explore new search areas.

Table 1. Input data for reliability allocation problems [2]

Sub-system i	Component Alternative											
	1			2			3			4		
	$R1$	$C1$	$W1$	$R2$	$C2$	$W2$	$R3$	$C3$	$W3$	$R4$	$C4$	$W4$
1	0.90	1	3	0.93	1	4	0.91	2	2	0.95	2	5
2	0.95	2	8	0.94	1	10	0.93	1	9	*	*	*
3	0.85	2	7	0.90	3	5	0.87	1	6	0.92	4	4
4	0.83	3	5	0.87	4	6	0.85	5	4	*	*	*
5	0.94	2	4	0.93	2	3	0.95	3	5	*	*	*
6	0.99	3	5	0.98	3	4	0.97	2	5	0.96	2	4
7	0.91	4	7	0.92	4	8	0.94	5	9	*	*	*
8	0.81	3	4	0.90	5	7	0.91	6	6	*	*	*
9	0.97	2	8	0.99	3	9	0.96	4	7	0.91	3	8
10	0.83	4	6	0.85	4	5	0.90	5	6	*	*	*
11	0.94	3	5	0.95	4	6	0.96	5	6	*	*	*
12	0.79	2	4	0.82	3	5	0.85	4	6	0.90	5	7
13	0.98	2	5	0.99	3	5	0.97	2	6	*	*	*
14	0.90	4	6	0.92	4	7	0.95	5	6	0.99	6	9

Bibliography

[1] Frank A. Tillman, Ching-Lai Hwang, Way Kuo, "Optimization Techniques for System Reliability with Redundancy - A Review", *IEEE Transactions on Reliability*, vol. R-26, no. 3, 1977 August, pp. 148-155.

[2] David E. Fyffe, William W. Hines, Nam Kee Lee, "System Reliability Allocation And a Computational Algorithm", *IEEE Transactions on Reliability*, vol. R-17, no. 2, 1968 June, pp. 64-69.

[3] Yuji Nakagawa, Satoshi Miyazaki, "Surrogate Constraints Algorithm for Reliability Optimization Problems with Two Constraints", *IEEE Transactions on Reliability*, vol. R-30, no. 2, 1981 June, pp. 175-180.

[4] Krishna Behari Misra, Usha Sharma, "An Efficient Algorithm to Solve Integer-Programming Problems Arising in System-Reliability Design", *IEEE Transactions on Reliability*, vol. 40, no. 1, 1991 April, pp. 81-91.

[5] Ching-Lai Hwang, Frank A. Tillman, Way Kuo, "Reliability Optimization by Generalized Lagrangian-Function and Reduced-Gradient Methods", *IEEE Transactions on Reliability*, vol. R-28, no. 4, 1979 October, pp. 316-319.

Table 2. Comparison of AS, N & M [3], and C & S [8] performance

No	C	W	N & M	C & S	AS with 10 runs			
			R	R	Max R	Min R	Avg R	Std Dev
1	130	191	0.9864	0.98675	0.98340	0.97850	0.98205	0.00144
2	130	190	0.9854	0.98603	0.97985	0.97771	0.97897	0.00094
3	130	189	0.9850	0.98556	0.98157	0.97943	0.98114	0.00090
4	130	188	0.9847	0.98503	0.98157	0.98157	0.98157	0
5	130	187	0.9840	0.98429	0.98151	0.97921	0.98082	0.00111
6	130	186	0.9831	0.98362	0.98189	0.97713	0.98073	0.00191
7	130	185	0.9829	0.98311	0.97972	0.97568	0.97803	0.00116
8	130	184	0.9822	0.98239	0.97972	0.97972	0.97972	0
9	130	183	0.9815	0.98190	0.97793	0.97386	0.97718	0.00158
10	130	182	0.9815	0.98102	0.97757	0.97537	0.97691	0.00106
11	130	181	0.9800	0.98006	0.97483	0.96875	0.97334	0.00165
12	130	180	0.9796	0.97942	0.97744	0.96903	0.97481	0.00264
13	130	179	0.9792	0.97906	0.97332	0.96958	0.97245	0.00144
14	130	178	0.9772	0.97810	0.97458	0.96995	0.97325	0.00171
15	130	177	0.9772	0.97715	0.97458	0.97458	0.97458	0
16	130	176	0.9764	0.97642	0.96683	0.96624	0.96654	0.00031
17	130	175	0.9744	0.97552	0.96683	0.96683	0.96683	0
18	130	174	0.9744	0.97435	0.96825	0.96269	0.96688	0.00202
19	130	173	0.9723	0.97362	0.97070	0.96825	0.97021	0.00103
20	130	172	0.9720	0.97266	0.97070	0.97070	0.97070	0
21	130	171	0.9700	0.97186	0.96650	0.96336	0.96619	0.00099
22	130	170	0.9700	0.97076	0.96351	0.95909	0.96216	0.00179
23	130	169	0.9675	0.96922	0.96606	0.96290	0.96485	0.00077
24	130	168	0.9666	0.96813	0.96357	0.96061	0.96259	0.00110
25	130	167	0.9656	0.96634	0.96357	0.96357	0.96357	0
26	130	166	0.9646	0.96504	0.96086	0.95809	0.95984	0.00094
27	130	165	0.9621	0.96371	0.95090	0.94160	0.94997	0.00294
28	130	164	0.9609	0.96242	0.95894	0.95260	0.95790	0.00203
29	130	163	0.9602	0.96064	0.95894	0.95894	0.95894	0
30	130	162	0.9589	0.95912	0.95894	0.95894	0.95894	0
31	130	161	0.9565	0.95803	0.94668	0.94380	0.94409	0.00091
32	130	160	0.9546	0.95567	0.95456	0.95191	0.95369	0.00118
33	130	159	0.9546	0.95432	0.95456	0.95456	0.95456	0

[6] Frank A. Tillman, Ching-Lai Hwang, Way Kuo, "Determining Component Reliability and Redundancy for Optimum System Reliability", IEEE Transactions on Reliability, vol. R-26, no. 3, 1977 August, pp. 162-165.

[7] David W. Coit, Alice E. Smith, "Reliability Optimization of Series-Parallel Systems Using a Genetic Algorithm", *IEEE Transactions on Reliability*, vol. 45, no. 2, 1996 June, pp. 254-260.

[8] David W. Coit, Alice E. Smith, "Penalty Guided Genetic Search for Reliability Design Optimization",

Computers and Industrial Engineering, vol. 30, no. 4, 1996, pp. 95-904.

[9] David W. Coit, Alice E. Smith, David M. Tate, "Adaptive Penalty Methods for Genetic Optimization of Constrained Combinatorial Problems", *INFORMS Journal on Computing*, vol. 8, no. 2, 1996 Spring, pp. 173-182.

[10] Laura Painton, James Campbell, "Genetic Algorithms in Optimization of System Reliability", *IEEE Transactions on Reliability*, vol. 44, no. 2, 1995 June, pp. 172-178.

[11] Marco Dorigo, Vittorio Maniezzo, Alberto Colorni, "Positive Feedback as a Search Strategy", *Technical Report No. 91-016,* 1991, Politecnico di Milano, Italy.

[12] Marco Dorigo, Vittorio Maniezzo, Alberto Colorni, "The Ant System: Optimization by a Colony of Cooperating Agents", *IEEE Transactions on Systems, Man, and Cybernetics-Part B,* vol. 26, no. 1, 1996 February, pp 29-41.

[13] Marco Dorigo, Luca M. Gambardella, "Ant Colonies for the Travelling Salesman Problem", *BioSystems,* vol. 43, 1997, pp. 73-81.

[14] Marco Dorigo, Luca M. Gambardella, "Ant Colony System: A Cooperative Learning Approach to the Travelling Salesman Problem", *IEEE Transactions on Evolutionary Computation,* vol. 1, no. 1, 1997 April, pp. 53-66.

[15] Bernd Bullnheimer, Richard F. Hartl, Christine Strauss, "Applying the Ant System to the Vehicle Routing Problem", *2nd Metaheuristics International Conference (MIC-97),* Sophia-Antipolis, France, 1997 July, pp. 21-24.

[16] Gianni Di Caro, Marco Dorigo, "Mobile Agents for Adaptive Routing", *Proceedings of the 31st Hawaii International Conference on System Sciences,* Big Island of Hawaii, January 6-9, 1998, pp. 74-83.

[17] W. Feller, *An Introduction to Probability Theory,* 1968, John Wiley & Sons.