# OPTIMAL RELIABILITY ALLOCATION IN SERIES-PARALLEL SYSTEMS FROM COMPONENTS' DISCRETE COST - RELIABILITY DATA SETS: A NESTED SIMULATED ANNEALING APPROACH

Subba Rao V. Majety[#], Srikanth Venkatasubramanian* and Alice E. Smith[#]
Departments of Industrial[#] and Chemical and Petroleum* Engineering
1048 Benedum Hall
University of Pittsburgh
Pittsburgh, Pennsylvania 15261
aesmith@engrng.pitt.edu

## ABSTRACT

It is generally accepted that a component's cost is an increasing function of its reliability. Most researchers adopt exponentially increasing functions for optimal reliability / redundancy allocation problem formulations. In this paper we address cases where an exact, convex relationship between reliability and cost is not known. Instead, we consider the more realistic assumption of discrete cost-reliability data sets for each component. The resulting integer programming formulations are non-linear and NP hard. We use a nested simulated annealing (SA) approach; one SA for feasibility and one SA for optimality to solve this difficult problem while ensuring feasibility of the final solution. This approach is demonstrated on two example problems with large search spaces.

## KEYWORDS

Reliability, Simulated Annealing, Redundancy Allocation, Design Optimization.

## INTRODUCTION

Many forms of the optimal reliability / redundancy allocation problem have been studied extensively in the past using various optimization techniques (e.g., Moscowitz and Mclean [1], Tillman and Littschwager [2], Sharma and Venkateswaran [3], Misra and Ljubojevic [4], Luus [5], Agarwal et al. [6], Tillman et al. [7, 8], Nakagawa and Nakashima [9], Inagaki et al. [10], Cateanu et al. [11], Hikita et al. [12], Gen et al. [13], Hwang et al. [14], Coit and Smith [15], Painton and Campbell [16]). When minimizing the cost at a specified system reliability is involved, the cost-reliability relationship for each component is of paramount importance. Without appropriate adoption of such a relationship, optimization become erroneous. For example, Tillman et al. [8] assumed an exponential relationship between cost vs. reliability in their formulation. However, if such relationships are non-convex, optimization procedures become more tedious, and more complex. In fact it is frequently noted that the cost of a component can be a non-convex, discontinuous function of its reliability. For example one can see such non-linear relationships between overall reliability vs. cost in Kumar and Malik [17], although the problem they address is not one of reliability / redundancy optimization.

The purpose of this paper is to address reliability allocation for systems where the cost-reliability relationships are non-convex, discrete functions. We consider the approach described here as more general and flexible. From a manufacturer's view, it is easier to provide a cost value for a specified reliability of a component as compared to providing a precise cost-reliability function of a component. Considering the difficulty in dealing with precise mathematical expressions for cost-reliability curves, we consider *discrete* cost-reliability data sets for each component.

In this study we consider series-parallel (SP) systems, although the approach could be generalized to other configurations. We adopt 0-1 integer formulations where each binary variable represents a data point on the cost-reliability curve of a component. Our objective is to find then minimum system cost which meets a minimum system level reliability constraint. In general, the system reliability is a non-linear function of all these variables. It can be easily shown that the problem is NP hard by considering a particular case of SP system, viz. a series system.

Simulated annealing (SA) techniques are adopted to find a good solution to SP systems. Usually in SA, the search for a neighboring solution is done within the feasible region of an optimization problem. In our problem, system reliability has to be computed for every point before checking for its feasibility. Hence it is not so easy to define a feasible neighborhood solution. An interesting feature of our work is that we use two SA procedures nested within the same algorithm where the first SA allows us to screen infeasible solutions probabilistically and the second SA allows us to accept a solution (feasible or infeasible) probabilistically, screened by first SA, even if its cost is more than the previous solution. As the algorithm proceeds, it becomes increasingly difficult to accept infeasible solutions due to the first SA, and also it becomes increasingly difficult to accept a solution that is not an improvement in the second SA.

The organization of this paper is as follows : in the next section we discuss the formulation of the problem with relevant notation. In section 3 we describe the algorithm developed to obtain good solutions for this problem. In section 4 we give example problems that are solved using this approach and discuss the results and related issues.

## NOTATION AND FORMULATION

### Notation

| | |
|---|---|
| $L$ | Number of subsystems in the system. |
| $i$ | Index for the subsystem; $i \in \{1, 2, ..., L\}$. |
| $n_i$ | Number of components in subsystem $i$. |
| $j$ | Index for the $j^{th}$ component in subsystem $i$; $j \in \{1, 2, ..., n_i\}$. |
| $K_{ij}$ | Size of the discrete component set considered; w.r.t. the $j^{th}$ component in subsystem $i$. |
| $\{p_{ijk}\}$ | Set of reliabilities of the $j^{th}$ component in subsystem $i$; $k \in \{1, 2,..., K_{ij}\}$. |
| $\{c_{ijk}\}$ | Set of costs of the $j^{th}$ component in subsystem $i$, associated with $\{p_{ijk}\}$. |
| $\{x_{ijk}\}$ | 0-1 variables associated with $\{p_{ijk}\}$ and $\{c_{ijk}\}$. |
| $f_i$ | Reliability of subsystem $i$. |
| $r_{ij}$ | Reliability of the $j^{th}$ component in subsystem $i$. |
| $R$ | Reliability of the system. |
| $R_s$ | Minimum required system reliability. |

Finally, $\alpha_R$ , $T_R$, , $\alpha_Z$ , $T_Z$ are SA parameters controlling the initial temperatures and the geometric cooling schedules.

### Problem Formulation

General formulation of the problem for any system is as follows

$$Min \sum_{i=1}^{L}\sum_{j=1}^{n_i}\sum_{k=1}^{K_{ij}} c_{ijk}x_{ijk}$$

$$s.t.\ R \geq R_s$$

$$: \quad x_{ijk} \in X$$

$$where$$

$$X = \{ x_{ijk} \mid \textstyle\sum_{k=1}^{K_{ij}} x_{ijk} = 1;\ \forall(i,j);$$
$$r_{ij} = \textstyle\sum_{k=1}^{K_{ij}} x_{ijk}p_{ijk};\ \forall(i,j);\quad x_{ijk} \in \{0,1\}\ \forall(i,j,k)\ \}$$

and $R$ is system reliability expressed as a function of all of its component reliabilities.

### Series Parallel Systems

If the number of components in each subsystem of SP equals one it becomes a simple series system. For a simple series system the non-linear system reliability constraint can be easily linearized by taking logarithms and using the equivalence $r_{1j} \in \{p_{1jk}\} \equiv \ln(r_{1j}) \in \{\ln(p_{1jk})\}$. It can be seen that this formulation is similar to the classic *multiple choice knapsack problem* (MCKP). Even though there are psuedopolynomial algorithms available for such problems as stated by Pisinger [18], the problem is essentially NP hard. For a simple series system, one can adopt a traditional cutting plane approach (Ceria et al. [19]), a branch and bound approach, or one of many available heuristic algorithms for MCKP (Pisinger [18]). Since the series system is a special case of an SP system, the formulation of SP will also be NP hard.

The formulation for an SP system is as follows:

$$Min \sum_{i=1}^{L}\sum_{j=1}^{n_i}\sum_{k=1}^{K_{ij}} c_{ijk} x_{ijk}$$

$$s.t. \prod_{i=1}^{L} f_i \geq R_s$$

$$f_i = 1 - \prod_{i=1}^{n_i}(1 - r_{ij}); \forall i.$$

$$x_{ijk} \in X$$

Unlike in the simple series system, the reliability constraint can not be linearized in this case. As such, the above problem is an integer non-linear programming problem which is difficult to solve directly. Thus we adopt a heuristic SA technique to address this problem, which is global (i.e., not restricted to local optima) and is non-enumerative.

## NESTED SIMULATED ANNEALING ALGORITHM

Many heuristic procedures such as SA (Brusco and Jacobs [20], Kirkpatrick et al. [21]), Genetic Algorithms [15], and Tabu Search (Glover [22, 23]) have been proposed for difficult combinatorial optimization problems. SA is a heuristic algorithm for obtaining good, though not necessarily optimal solutions, to optimization problems. Brusco and Jacobs listed many combinatorial problems for which SA has been successfully applied. Those include school time tabling, multi level lot sizing and cyclic staff-scheduling.

In a traditional SA approach, one starts with a feasible solution and identifies a feasible neighboring solution. If this new solution improves the objective function it is immediately accepted and a move is made to that solution. If not, the new solution is accepted probabilistically based on some annealing schedule. Eglese [24] mentions that the efficiency of an SA algorithm depends on the definition of the feasible neighborhood of the current solution. However, instead of defining a feasible neighborhood, if the problem has a difficult-to-satisfy constraint set, an addition of an exterior penalty function to the objective function may be used. In our problem, we used a nested SA algorithm instead of a penalty function. In principle it is quite similar to the penalty function approach but the difference is that the penalty is applied to the acceptance probability for an infeasible solution rather than to objective function, as is done normally. Our initial experiments with the nested SA always resulted in feasible final solutions (not always achieved with penalty functions) and thus we adopted the feasibility SA instead of regular penalty function approach.

The nested SA algorithm is as follows :

**Step 0** : Generate an initial feasible solution $X$ with reliability $R$ and cost $Z$.
Set initial conditions $T_R$, $T_Z$, *stopping criterion*.
Set *accept = no*.

**Step 1** : If *stopping criterion* is met, *stop*.
Set *accept = no*.

**Step 2** : Generate a neighboring solution $X'$ of $X$ with reliability $R'$ and cost $Z'$.
(i) If $R' < R_s$ , then set *accept = yes* with *probability* $e^{-(R_s - R')/T_R}$; if $R' \geq R_s$ set *accept = yes*.
(ii) If *accept = yes* and $Z' > Z$ , then set $X = X'$ with *probability* $e^{-(Z-Z')/T_Z}$ ,
If *accept = yes* and $Z' \leq Z$ , then set $X = X'$
(iii) Set $T_R = \alpha_R T_R$; $T_Z = \alpha_Z T_Z$ ;

**Step 3** : Return to Step 1.

Notice from the above that the probability of acceptance for an infeasible neighboring solution becomes $p_1$ *times* $p_2$; where $p_1$ is ($e^{-(R_s - R')/T_R}$) and $p_2$ is ($e^{-(Z-Z')/T_Z}$ or 1). Therefore the acceptance of a candidate solution is guaranteed if the solution is feasible and of lower cost than the previous solution. If either or both of these conditions are not true, acceptance is probabilistic depending on the degree of infeasibility, the amount of cost increase and the point on the cooling schedule at which the search currently resides. Probabilistic acceptance for violation of either condition becomes more selective as the search proceeds.

### Initial Feasible Solution

An initial feasible solution can be generated many ways. We adopted the following approach :

(i) Assume all $f_i$ are equal for all subsystems $i$. Thus compute $f_i = (R_s)^{1/L}$.

(ii) Assume all $r_{ij}$ are equal for components $j$ in each subsystem $i$. Thus compute $r_{ij} = f_i^{1/n_i}$ .

(iii) Look into the cost / reliability sets of each component and assign a $p_{ijk}$ if $p_{ijk-1} < r_{ij} < p_{ijk}$, and assign the corresponding cost $c_{ijk}$ to a component ($i$, $j$).

**Neighboring Solution**

It is very difficult to define a precise neighborhood for some combinatorial problems. In this problem the precise and comprehensive definition of neighborhood is also complicated. One could fix a reliability of each subsystem first and then work out components' reliability; or one could arbitrarily pick some components and randomly increase or decrease from the current reliability to the next value in their reliability sets and so on. While the neighborhood definition itself is complicated, assuring feasibility of the minimum system reliability level for the neighboring solution adds to this complexity. The feasibility of a neighboring solution can be ensured but not without some computational effort; which could be very cumbersome. Only searching the feasible region of a constrained problem also often results in inefficient search and suboptimal final solutions. Therefore, we are willing to accept an infeasible neighboring solution, with some probability as described in the algorithm. The nested annealing process makes the acceptance of such infeasible solutions more unlikely as the search progresses.

The neighborhood is defined as below :

(i) We define *k_index* (*i, j*) for each component at the current solution as the index set such that *k_index* (*i, j*) = *k* if $p_{ijk}$ is assigned to component (*i, j*). One may note that $\{p_{ijk}\}$ are ordered by *k* .

(ii) Randomly select some components in the system and randomly increase or decrease *k_index* (*i, j*) by 1 for selected components only. If *k_index* (*i, j*) is already at the lowest value (highest value) for a selected component then we increase (decrease) its value. The neighboring solution obtained in this fashion may not be feasible, but one can easily see that it guarantees a non-zero probability to reach any solution from a given solution.

We adopted the standard geometric cooling schedule for both the feasibility SA as well as the objective SA with $\alpha_R$ and $\alpha_Z$ both = 0.99, and $T_R$ for example 1 = 1000, $T_R$ for example 2 = 5000, $T_Z$ for example 1 = 10000 and $T_Z$ for example 2 = 25000. The stopping criterion used was the total number of solutions considered, which was set to 20000 for example 1 and 30000 for example 2. The algorithm was tested on two problems and the results are summarized in the next section. The algorithm was coded in Borland C++ 4.0v, on a GATEWAY2000/p5-90(pentium) personal computer.

## EXAMPLES AND RESULTS

### Example 1

We chose an SP system with nine components for example 1. For ease of handling we fixed the same reliability data set for all components. For each component, we assigned randomly generated cost values to corresponding reliability values in the data set as shown in Table 1. Notice that for *k* = 1, all components have very low reliability and zero cost. If any component is assigned *k* = 1 in the optimal solution that component is deemed redundant and hence can be removed from the system, that is, it is a "blank." We assured an increasing trend of cost w.r.t. reliability while assigning costs. However, a convex relationship is not guaranteed (as desired) due to randomness. The components are then randomly assigned to three subsystems of an SP system. The final SP system for the example problem has 3, 4 and 2 components in its three subsystems, respectively. The total search space is equal to $12^9$.

From our experience in experimenting with the nested SA, the best final solutions are obtained when the algorithm visited mostly infeasible solutions. For example (see Table 3), for the best solution observed across 30 independent runs; the ratio of infeasible to feasible (IFR) solutions is 6.52. When we restricted this ratio below 1.5 in the algorithm, the best feasible solution cost from those runs was quite suboptimal (above 600). The nested SA was run with no restrictions on IFR, and statistics are presented in Table 3 for 30 different independent runs. The best solution across all runs is within 6.65% of the global optimum solution obtained by enumeration. The search space considered by the SA (20000 solutions) was an extremely small fraction of the total of $12^9$.

### Example 2

We chose example 2 to demonstrate that the model suggested in this paper addresses the redundancy issue as well. For this purpose, we added two more components to the above system in subsystem 1 (see Table 2), enlarging the search space to $12^{11}$, but not altering the optimal solution. This time, we generated higher costs purposefully for the newer components.

As in example 1, the best solutions are obtained when the algorithm visited mostly infeasible solutions. For the best solution observed across 30 independent runs of example 2 the ratio of infeasible to feasible (IFR) solutions is 10.11. The nested SA was run again with no restrictions on IFR, and statistics are presented in Table 3 for 30 different independent runs. This time the best solution across all runs is within 7.82% of global solution with the SA considering only a tiny fraction of the search space (30000 solutions of $12^{11}$). It is interesting to note that in all the 30 different runs; the best solution always contained the lowest reliability for the two additional high cost components. Thus the algorithm clearly indicated that these components are redundant components to the system and are “blanks.”

## CONCLUSIONS

We formulated a SP reliability optimization problem as a mixed integer non-linear programming problem based on non-convex discrete cost-reliability data sets. We adopted a nested SA algorithm as the solution procedure to this NP hard combinatorial problem. Initial results are encouraging and the solution procedure clearly identified near optimal configurations of redundant components in the system, while maintaining feasibility of the final solution. Note that the nested SA does not require the extensive tuning and parameter setting of many penalty function approaches. Only a relatively straightforward geometric cooling schedule needs to be specified.

We did not consider other constraints such as weight or volume, however it would be easy to extend the formulation of the problem to handle these additional constraints. While applying the nested SA, one can include more SA procedures for the feasibility of these additional constraints. We believe that it is worthwhile to investigate similar nested SA algorithms for other combinatorial problems.

## REFERENCES

1. Moskowitz, F. and Mclean, J. B., Some reliability aspects of system design, *IRE Transactions on Reliability and Quality Control*, v-8, 1956, 7-35.
2. Tillman, F. A. and Littschwager, J. M., Integer programming formulation of constrained reliability problems, *Management Science*, v-13, 1967, 887-899.
3. Sharma, J. and Venkateswaran, K. V., A direct method for maximizing the system reliability, *IEEE Transactions on Reliability*, v-20, 1971, 256-259.
4. Misra, K. B. and Ljubojevic, M. D., Optimal reliability design of a system : a new look, *IEEE Transactions on Reliability*, v-22, 1973, 255-258.
5. Luus, R., Optimization of system reliability by a new nonlinear integer programming procedure, *IEEE Transactions on Reliability*, v -24, 1975, 14-16.
6. Agarwal, K. K., Gupta, J. S. and Misra, K. B., A new heuristic criterion for solving a redundancy optimization problem, *IEEE Transactions on Reliability*, v -24, 1975, 86-87.
7. Tillman, F. A., Hwang, C. L., Fan, L. T. and Balbale, S.A., Systems reliability subject to multiple nonlinear constraints, *IEEE Transactions on Reliability*, v -17, 1968, 153-157.
8. Tillman, F. A., Hwang, C. L. and Kuo, W., Determining component reliability and redundancy for optimal system reliability, *IEEE Transactions on Reliability*, v -26, 1977, 162-165.
9. Nakagawa, Y. and Nakashima, K., A heuristic method for determining reliability allocation, *IEEE Transactions on Reliability*, v -26, 1977, 156-161.
10. Inagaki, T., Inoue, K. and Akashi, H., Interactive optimization of system reliability under multiple objectives, *IEEE Transactions on Reliability*, v -27, 1978, 264-267.
11. Cateanu, V. M., Popentiu, F. and Gheorgiu, M., A reliability optimization computer algorithm for complex systems, *Microelectronics Reliability*, v-26, 1986, 1019-1023.
12. Hikita, M., Nakagawa, Y., Nakashima, K. and Narihisa, H., Reliability optimization of systems by a surrogate-constraints algorithm, *IEEE Transactions on Reliability*, v -41, 1992, 473-480.
13. Gen, M., Ida, K., Tsujimura, Y. and Kim, C. E., Large scale 0-1 fuzzy goal programming and its applications to reliability optimization problem, *Computers and Industrial Engineering*, v-24, 1993, 539-549.
14. Hwang, C. L., Lee, H. B., Tillman, F. A. and Lie, C. H., Nonlinear integer goal programming applied to optimal system reliability, *IEEE Transactions on Reliability*, v -33, 1984, 431-438.
15. Coit, D. W. and Smith, A. E., Reliability optimization of series-parallel systems using a genetic algorithm, *IEEE Transactions on Reliability*, 1996, in press.
16. Painton, L. and Campbell, J., Genetic algorithms in optimization of system reliability, *IEEE Transactions on Reliability*, v-44, 1995, 172-178.
17. Kumar, A. and Malik, K., Voting mechanisms in distributed systems, *IEEE Transactions on Reliability*, v-40, 1991, 593-600.
18. Pisinger, D., A minimal algorithm for the multiple-choice knapsack problem, DIKU, University of Copenhagen, Denmark, Report 94/25, 1994.
19. Ceria, S., Cornuejols, G. and Dawande, M., Combined gomory cuts, *Proceedings of the 4th International Conference on Integer Programming and Combinatorial Optimization*, May 29-31, 1995, Copenhagen, Denmark.
20. Brusco, M. J. and Jacobs, L. W., A simulated annealing approach to the cyclic staff-scheduling problem, *Naval Research Logistics*, v-40, 1993, 69-84.

21. Kirkpatrick, S., Gelatt, C. D. and Vecchi, M. P., Optimization by simulated annealing, *Science*, v-220, 1983, 671-680.
22. Glover, F., Tabu search, *ORSA Journal on Computing*, v-1, 1989, 190-206.
23. Glover, F., Tabu search - Part II, *ORSA Journal on Computing*, v-2, 1989, 4-32.
24. Eglese, R. W., Simulated annealing : A tool for operational research, *European Journal of Operational Research*, v-46, 1990, 271-281.

## BIOGRAPHICAL SKETCHES

**Subba Rao V. Majety** is a Ph.D. candidate in Industrial Engineering at the University of Pittsburgh. He has a M.A.Sc. in Industrial Engineering from University of Windsor, Canada. His research interests include reliability optimization, application of OR techniques and heuristic optimization methods.

**Srikanth Venkatasubramanian** is a Ph.D. candidate in Chemical and Petroleum Engineering at the University of Pittsburgh. He has a B.E. (Hons.) in Chemical Engineering from Birla Institute of Technology and Science in India. His research is the development of AI based control and diagnostics for powder and bulk solids handling and processing.

**Alice E. Smith** is Assistant Professor of Industrial Engineering and Bicentennial Board of Visitors Faculty Fellow at the University of Pittsburgh. Her research in modeling and optimization of manufacturing processes and engineering design has been funded by Lockheed Martin Corp., ABB Daimler Benz Transportation, the Ben Franklin Technology Center of Western Pennsylvania and the National Science Foundation, from which she was awarded a CAREER grant in 1995. She is an associate editor of *INFORMS Journal on Computing* and *Engineering Design and Automation.*

**Table 1 : Cost and Reliability Data for Example 1 : { $c_{ijk}$ } Values ( $R_s$ = 0.85 )**

| *i* | *j* | RELIABILITY VALUES | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | k=1 | k=2 | k=3 | k=4 | k=5 | k=6 | k=7 | k=8 | k=9 | k=10 | k=11 | k=12 |
| | | 0.001 | 0.5 | 0.55 | 0.6 | 0.65 | 0.7 | 0.75 | 0.8 | 0.85 | 0.9 | 0.95 | 0.99 |
| 1 | 1 | 0.0 | 4.05 | 16.30 | 40.40 | 67.35 | 95.70 | 135.75 | 186.05 | 251.05 | 339.80 | 440.45 | 597.70 |
| 1 | 2 | 0.0 | 3.65 | 17.75 | 36.00 | 59.35 | 78.50 | 169.60 | 224.45 | 303.80 | 391.95 | 505.30 | 654.45 |
| 1 | 3 | 0.0 | 9.10 | 22.35 | 44.45 | 71.55 | 105.10 | 148.85 | 198.10 | 276.75 | 374.25 | 496.80 | 633.65 |
| 2 | 1 | 0.0 | 4.35 | 14.10 | 29.15 | 50.45 | 78.20 | 117.55 | 170.90 | 248.55 | 347.90 | 463.75 | 609.40 |
| 2 | 2 | 0.0 | 3.15 | 10.80 | 31.95 | 52.0 | 183.25 | 222.10 | 278.80 | 350.30 | 434.15 | 539.30 | 699.15 |
| 2 | 3 | 0.0 | 7.80 | 22.85 | 43.85 | 70.80 | 101.45 | 143.70 | 202.05 | 276.70 | 370.20 | 495.15 | 628.50 |
| 2 | 4 | 0.0 | 8.75 | 18.80 | 42.80 | 72.05 | 106.25 | 151.20 | 210.95 | 289.95 | 370.00 | 482.75 | 636.60 |
| 3 | 1 | 0.0 | 5.45 | 16.45 | 36.45 | 60.70 | 191.2 | 230.75 | 282.0 | 354.00 | 449.50 | 572.75 | 703.30 |
| 3 | 2 | 0.0 | 2.05 | 7.67 | 23.87 | 101.90 | 128.81 | 164.35 | 207.00 | 271.25 | 362.80 | 480.95 | 623.40 |

**Table 2 : Cost Data for Additional Components in Example 2 : { $c_{ijk}$ } Values ( $R_s$ = 0.85)**

| *i* | *j* | RELIABILITY VALUES | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | k=1 | k=2 | k=3 | k=4 | k=5 | k=6 | k=7 | k=8 | k=9 | k=10 | k=11 | k=12 |
| | | 0.00 | 0.5 | 0.55 | 0.6 | 0.65 | 0.7 | 0.75 | 0.8 | 0.85 | 0.9 | 0.95 | 0.99 |
| 1 | 4 | 0.0 | 231.65 | 272.75 | 456.00 | 689.35 | 820.50 | 989.60 | 1354.45 | 2043.80 | 3491.95 | 4605.30 | 6754.45 |
| 1 | 5 | 0.0 | 219.10 | 232.35 | 364.45 | 591.55 | 735.10 | 978.85 | 1148.10 | 1676.75 | 2474.25 | 3696.80 | 5733.65 |

**Table 3 : Summary of Results for Examples 1 and 2 with Nested SA over 30 Runs of Each**

| Problem | Max. Cost | Mean Cost | Min. Cost | No. of Iterations | Feasible Solutions | Best Configuration | | | Reliability of Best | CPU Time (sec/run) |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 662.05 | 572.72 | 533.90 | 20000 | 2660 | 4 -5-4 | 2-5-3-4 | 5-8 | 0.850266 | 2.54 |
| 2 | 726.15 | 599.47 | 541.10 | 30000 | 2699 | 7-4-3-1-1 | 2-4-3-3 | 5-8 | 0.850922 | 5.70 |
| Optimal | - | - | 500.60 | - | - | 3-6-5 | 4-3-2-3 | 5-8 | 0.850172 | ≅6 hours |

Where the optimal solution was obtained by enumeration.