Minimizing Cost of Redundant Sensor-Systems with Non-Monotone and Monotone Search Algorithms

R. R. Brooks; California State University Monterey Bay, Seaside
S. S. Iyengar; Louisiana State University, Baton Rouge
Suresh Rai; Louisiana State University, Baton Rouge

Key words: Dependability, Cost, Simulated annealing, Tabu Search, Heuristic Method

SUMMARY & CONCLUSIONS

This paper considers distributed multisensor applications, and finds redundant configurations that minimize system weight or cost while insuring system dependability. Given choices among different component types, fulfilling the system's operational requirements but having different dependability parameters and per item cost, two heuristics, one non-monotone (tabu search) and one monotone (simulated annealing), are used to find configurations that minimize the chosen cost metric. The search is limited to a surface delimiting the solution space region fulfilling system dependability requirements. Experimental results are presented with cost savings of 20% compared with the least expensive system consisting of only one component type. A test case compares results from the two methods with an exhaustive search to verify that the heuristics provide reasonable solutions.

1. *INTRODUCTION*

Redundant systems achieve fault tolerance by duplication of components. Feasibility requires attention be paid to both reliability bounds and cost. The Byzantine Generals Problem concerns making unanimous decisions in the presence of arbitrary errors in less than one third of the components (Ref. 13). Ref. 2 provides a full discussion of this topic. Sensor fusion systems make decisions based on data from multiple sensors and glean the best interpretation from noise-corrupted data of limited resolution. Figure 1 illustrates a one dimensional problem, where algorithms in Refs. 9,16 tolerate failures of up to one half of the sensors. Both Byzantine Generals Problem and sensor fusion are examples of fault masking. The information theoretical basis of fault masking is explored in Ref. 11. Brooks and Iyengar explain the relationship between sensor fusion and the Byzantine Generals problems in Refs. 4,5. Note that sensor fusion and Byzantine agreement aid in designing reliable systems which function correctly when more than half (one-dimensional sensor fusion) or more than two-thirds (Byzantine Agreement) of the individual components function correctly.

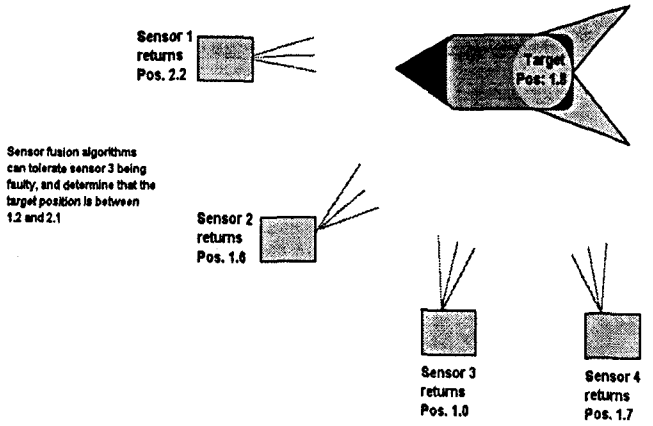

Figure 1 These sensors function correctly if the data is within 0.5 of the real value.

1.1 *Motivation*

Our motivation comes from the sensor integration paradigm, which increases the ability of systems to interact with their environment by combining independent sensor readings into logical representations (Refs. 8,15). Sensor integration of highly redundant systems offers these advantages: 1) Multiple inaccurate sensors can cost less than a few accurate sensors. 2) Sensor reliability may increase (Ref. 1). 3) Sensor efficiency and performance can be enhanced. 4) Self-calibration can be attained (Ref. 24). Highly redundant sensors are used in key areas. Defense related applications include missile defense, global positioning, autonomous land vehicles, pilot assistance, and command and control (Refs. 15,17). Intelligent manufacturing requires distributed systems with independent sensors (Ref. 8). Systems use distributed sensors for applications like Desert Storm (Ref. 21), and controlling smuggling (Ref. 10). These designs depend on the "best possible trade-off at least cost" (Ref. 17).

An open question is sensor selection to improve reliability and resolve resource allocation conflicts (Ref. 24). An approach to sensor selection using system cost and sensor accuracy is given in (Ref. 1). Our paper presents a more general approach using dependability instead of accuracy. We consider redundancy among heterogeneous components. The

0-7803-3783-2/97/$5.00 © 1997 IEEE

use of off-the-shelf components to reduces system complexity and cost, particularly for sensors (Refs. 14,25). A draft standard for sensor interfaces has been proposed by the National Institute of Standards and Technology (Ref. 18). Acceptance of this standard will allow heterogeneous sensor redundancy that does not effect system complexity.

1.2 *Nomenclature and Notation List*

The following variables are used in this report:

J - Total number of component types under consideration

N - Total number of components in a given system which satisfies dependability constraints.

N_i - Number of components of type i needed to create a system which satisfies dependability constraints when only components of type i are used

c_i - Unit cost of component type i

DC - Difference in cost between two system configurations

t - Temperature variable used by simulated annealing.

x - System configuration vector $(x_1,x_2,...,x_i,....,x_J)$

x_i - Number of components of type i in a given system which satisfies the dependability constraints.

$r(t)$ - Reliability function giving the probability that the component fails after mission time t

a - Steady state component availability.

$f(x)$ - Fitness function giving the relative merit of system configuration x

In this report we assume:

1) An adequate reliability or availability statistic is known for each component type.

2) Component failures are statistically independent and dependability is between 0.5 and 1.

3) A cost metric c_i is known for each component type.

1.3 *Problem Statement and Report Layout*

We characterize components by dependability statistics and unit cost. Our method can use statistics based on several existing models. The dependability statistics used depend on the exact problem. If the design is concerned with mechanical failure, mean-time-to-failure and mean-time-to-repair are adequate. If transient or intermittent errors are considered, statistics must be available for fault arrival and duration. For tractability and consistency with reliability literature our example uses exponential distributions for component failure and repair. Equations are presented for systems where over 50% of all components must be functional, this can be changed to any other percentage by replacing $N/2$ as necessary.

Dependability is a generic term for either *reliability*, the probability that the system is working at a given mission time, or *availability*, the percentage of time the system is functional. When a system with strict dependability requirements uses Byzantine agreement or sensor fusion, a choice is made between component types which may be used redundantly to mask errors. Given a choice between various modules, each with different dependability parameters and per item cost, we attempt to obtain the configuration which meets the dependability requirements with the lowest system cost. Cost may be dollar amounts or weight. The dual problem, maximizing dependability within cost bounds, can be solved by making minor modifications.

The paper is organized as follows: Section 2 presents the mathematics required for this problem. Section 3 gives two heuristic search algorithms for finding approximate answers. We discuss results illustrating the quality of the heuristics in Section 4. Final discussion is in Section 5.

2. *DEPENDABILITY CONSTRAINTS*

Figure 2 shows a Markov chain model of our process (for reliability evaluation, the repair rate m is zero) where N identical components fail independently which can be solved to obtain the reliability or availability parameter. Alternatively, a combinatorial approach can derive identical results. Assume each component has an identical probability of success $r(t)$. Let $q(t) = 1 - r(t)$. The assumption of statistical independence allows Bernoulli's law to find the probability of i out of N components working at time t as:

$$\binom{N}{i} [r(t)]^{i} [q(t)]^{N-i} \qquad (1)$$

The reliability of the system is the sum of the probabilities for the states with i equal to N to $\lfloor N/2 \rfloor+1$ since the system is functional only when more than $N/2$ components are operational. We describe a model with two types of components, where the total number of components N is equal to the number of components of type 1 (x_1) plus the number of components of type 2 (x_2), and then a generalized version is considered.

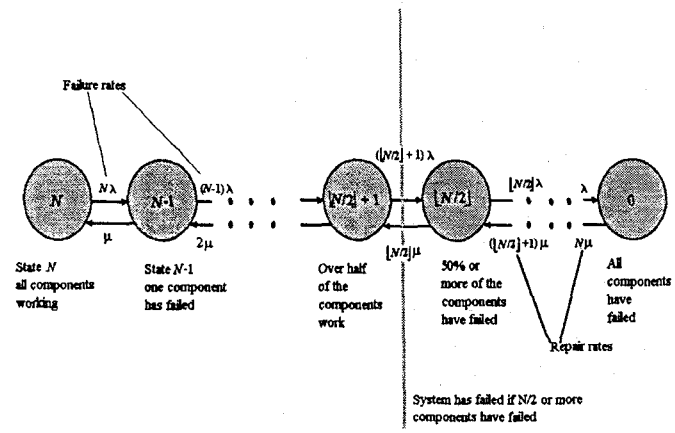

Figure 2 Markov chain model of a system which tolerates up to 50% component failure.

Assume cases where no components of type 1 have failed, one component of type 1 has failed, *etc.*, up to the case where all x_1 components of type 1 have failed. The cases are disjoint and the sum of their probabilities is one. They partition the space giving a reliability expression at time t in terms of $r_1(t)$ component type 1 reliability, and $r_2(t)$ component type 2 reliability.

$$R(t)=\sum_{k=0}^{x_1}\left[\binom{x_1}{k}\,[r_1(t)]^k[1-r_1(t)]^{x_1-k}\sum_{m=\lfloor N/2\rfloor+1-k}^{x_2}\left[\binom{x_2}{m}\,[r_2(t)]^m[1-r_2(t)]^{x_2-m}\right]\right] \quad (2)$$

Equation (2) can be extended to more than two types of components. A combination of J different types of components requires J levels of summations in the format of (2). Ref. 3 explains methods for performing these calculations efficiently.

The extension of equation (2) to J dimensions determines if a system fulfills dependability requirements. We use this to find the configuration that fulfills the dependability criteria with minimum cost. We consider each combination of J types of components as a point in a discrete J-dimensional space, described by a J-dimensional vector $(x_1,x_2,\ldots,x_J)$. Each position in the vector corresponds to the number of components of a given type in the configuration. If the choice is made among three types, the combination of 2 components of type 1, 25 components of type 2, and none of type 3 corresponds to point (2, 25, 0). Equation (2) determines which points in the J-dimensional space fulfill the dependability requirements. Since each component type has a known cost it is possible to determine the system cost. If type i has cost c_i, the cost of combination (2,25,0) is $2*c_1 + 25*c_2 + 0*c_3$. We must minimize:

$$f(x) = \sum_{i=0}^{J} c_i * x_i \quad (3)$$

This is a combinatorial optimization problem which can not be solved by known mathematical programming techniques such as integer programming. These techniques are inappropriate because equation (2) which defines whether or not a given combination fulfills dependability requirements is non-linear (Ref. 20). We formulate the problem as a search for the optimal point in a J-dimensional discrete solution space where J is the number of types under consideration. The region with valid solutions is known as the feasible set in the solution space (Ref. 23). We determine a surface which divides the points containing combinations in the feasible set from the rest of the J-dimensional space. This surface restricts our search for the optimal configuration to a small portion of all configurations.

Lemma 1. An optimal answer must lie on the surface dividing the J-dimensional problem space into two regions: one region containing points which either satisfy the dependability requirements or contain a subset which does, and one region containing points which do not satisfy the dependability requirements.

Proof. Three distinct types of points exist: points beneath the surface in the region which does not satisfy reliability constraints, points on the surface dividing points which satisfy reliability constraints from those which do not, and points above the surface in the region which contains combinations that satisfy reliability constraints. Points below the surface can be dismissed trivially since they do not satisfy the dependability constraints. Any point K above the surface satisfies dependability constraints or contains a subset which does, but can be dismissed since there is at least one other point L which satisfies the dependability constraints with at least one component of at least one type fewer. Point L has a lower cost than point K, since all costs are positive. Thus the minimum cost point being sought must be located on the surface defining the set of points which satisfy the reliability requirements. •

3 *NON-MONOTONE AND MONOTONE HEURISTIC SEARCH ALGORITHMS*

Many search methods used to navigate solution spaces containing local minima belong to the classes of non-monotone and monotone search algorithms (Refs. 6,7). Both methods use a fitness function $f(x)$ to determine the relative merit of possible solutions. Local minima can be the result of the feasible set being defined by a non-convex surface, or the fitness function being non-linear. Optimization problems containing local minima are more difficult than linear problems and can not be solved by linear programming.

The majority of search heuristics start at a point in the parameter space and move to neighboring points whose value of $f(x)$ is inferior to the current value. Both non-monotone and monotone methods occasionally move to points in the search space where the value of $f(x)$, the fitness function, is superior to the current value. With monotone methods the ability to do this is dependent on a strictly decreasing parameter, for non-monotone methods it is not.

Monotone methods include simulated annealing (see section 3.2) and threshold acceptance. Monotone methods avoid local minima by occasionally moving to a neighboring point whose value of $f(x)$ is superior to the current value. The amount of increase that will be accepted by the method can be either probabilistic (simulated annealing) or deterministic (threshold acceptance) and is dependent on a parameter whose value decreases as the search progresses (Ref.s 6,7).

3.1 *Tabu Search*

Tabu search is non-monotone in that it disqualifies a number of moves due to the history of moves made by the algorithm (Ref. 6). We use Tabu search since it is the most widely implemented heuristic of this class. Tabu search modifies an existing heuristic by keeping a list of the nodes in the search space visited most recently by the search. These points become "tabu" in that they are not revisited while they are on the list. This allows an algorithm to climb out of local

minima. Our implementation uses a tabu list that is practically infinite.

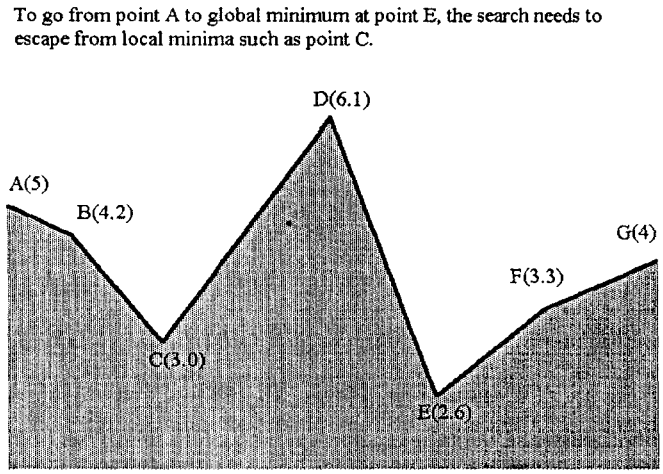

Figure 3 One dimensional search space with local minima.

Figure 3 illustrates the problem. A greedy heuristic starts at A, moves to B and then C due to the decreasing values. All neighbors of C have larger values. Point C is a local minima and a greedy heuristic would return C (3.0) as the minimum. This is incorrect as the minimum value is at E (2.6). Tabu search puts points visited on a list and forbids movement to these points. At C the tabu list is {A,B,C} and the only neighbor not in the tabu list is D. From D, tabu search moves directly to the global minimum E.

The tabu search used here relies on a "greedy" heuristic and starts with the lowest cost configuration of only one component type. The algorithm evaluates the cost of each neighboring configuration using the fitness function. The search moves to the configuration with the minimum value for the fitness function $f(x)$. When a configuration is visited it is placed on the tabu list, should the search return to its neighborhood later, the fitness function is set to a large value. As each configuration is visited, the value of the fitness function is compared to the smallest value found up to that point. If the value is smaller, the configuration becomes the best fit found. Figure 4 presents pseudocode for this tabu search.

Algorithm: tabu_search
Inputs: J, d_i for $1 \le i \le J$, c_i for $1 \le i \le J$, and D.
Outputs: Vector L of length J with the minimum cost dependable configuration.
Procedure:

Step 1: Compute N_i, $1 \le i \le J$. Note, N_i is the number of components of type i needed to meet requirement D when no components of another type are used.
Step 2: Sort component types in increasing order of $N_i * c_i$.
$x_1 = N_1$ $\quad x_i = 0,\ 2 \le i \le J$
Low_Conf $= (x_1, x_2, \ldots, x_J)$ *Current_Conf* $= (x_1, x_2, \ldots, x_J)$
For $I := 1$ to N do /* N is determined experimentally */
Step 3: Compute $f(x)$ for all $2J$ neighbors of *Current_Conf*.
A neighbor is a configuration which is made by adding or subtracting 1 component from any x_i ($2 \le i \le J$). A negative number of components is not allowed. If any neighbor is on the tabu list set the value of $f(x)$ for that neighbor to a prohibitively large value.
Step 4: Set *Current_Conf* to the neighbor with the smallest $f(x)$ value.
Step 5. If f(*Current_Conf*) $< f$(*Low_Conf*) then *Low_Conf* $= x$
Step 6. Append *Current_Conf* to tabu list.
end for loop.
Step 7. return(*Low_Conf*)

Figure 4 Pseudocode for cost minimization using tabu search.

x_1 refers to the number of components of type one in the configuration. Type 1 always refers to the component which can be used alone to create the lowest cost configuration. This is the *lowest cost solo configuration.* The value of x_1 is the smallest number of components of type 1 needed with the values of $x_2, x_3, \ldots, x_J$ to create a configuration which fulfills dependability constraints. The fitness function $f(x)$ refers to equation (3). Since no clear stopping criteria exists, this study compared the results from a given number of iterations of the tabu search with the results obtained by performing a simulated annealing which ran until completion.

3.2 *Simulated Annealing*

Simulated annealing attempts to find optimal answers in a manner analogous to the formation of crystals in cooling solids. A material heated beyond a certain point will become fluid, if the fluid is cooled slowly the material will form crystals and revert to a minimal energy state. Refer to Ref. 12 for a full description of simulated annealing and a discussion of its scientific basis.

The strategy of the algorithm is again based on a *fitness function* comparing the relative merit of various points in the problem space. We use the same solution space, same starting point and fitness function with tabu search and simulated annealing. From the algorithm's current position a neighboring point is chosen at random. The cost difference between the new point and the current point is calculated. This difference is used together with the current system temperature to calculate the probability of the new position being accepted. This probability is given by the distribution $e^{-\Delta C/\tau}$. The process continues with the same temperature τ for either a given number of iterations, or until a given number of positions have been occupied, at which time the value τ is decreased. The temperature decreases until no transitions are possible, so the system remains frozen in one position. This occurs only when ΔC is positive for all neighboring points, therefore the position must be a local minimum and may be the global minimum (Ref. 19).

The simulated annealing method used is based on the algorithm given in (Ref.s 12,19). The algorithm has been modified so that the problem space and fitness function are appropriate. A cooling schedule has been found which allows

the algorithm to converge to a reasonable solution. Several possible *cooling schedules* exist. A cooling schedule is defined by the initial temperature, the number of iterations performed at each temperature, the number of position modifications allowed at a given temperature and the rate of decrease of the temperature. The answers found by the algorithm are directly dependent on the cooling schedule. Note, there does not exist a definite rule for defining the schedule (Ref.s 12,19).

Algorithm: simulated_annealing
Inputs: J, d_i for $1 \le i \le J$, c_i for $1 \le i \le J$, and D.
Outputs: Vector L of length J with the minimum cost dependable configuration.
Procedure:
Step 1: Compute N_i, $1 \le i \le J$. Note, N_i is the number of components of type i needed to meet requirement D when no components of another type are used.
Step 2: Sort component types in increasing order of $N_i * c_i$.
Step 3: $CC = (N_1, 0, ...,0)$ /* Initial position */
$\tau = 1.0$ /* Initial temperature */
Step 4: $CC\_mod = 1$ $step4\_iter = 0$
While (CC_mod not = 0) and
($step4\_iter$ < maximum number for step 4) do
begin
$CC\_mod = 0$ $inner\_loop\_iter = 0$
While(CC_mod < maximum number of transitions) and
($inner\_loop\_iter$ < maximum number for inner loop)
do begin
x_1 = number of components of type 1 needed to fulfill the dependability constraint for CC
$CC = CC$ with first position x_1
new_CC = random modification of CC
x_1 = number of components of type 1 needed to fulfill the dependability constraint for new_CC
$new\_CC = new\_CC$ with first position x_1
ΔC = cost(CC) - cost(new_CC)
if($\Delta C < 0$) then
begin
$CC = new\_CC$
$CC\_mod = CC\_mod + 1$
end
else following Boltzmann distribution of ΔC and τ
do begin
$CC = new\_CC$ $CC\_mod = CC\_mod + 1$
end
$inner\_loop\_iter = inner\_loop\_iter + 1$
end
$\tau = 0.9 * \tau$ $step4\_iter = step4\_iter + 1$
end
Step 5: Output CC as the minimal cost configuration .

Figure 5 Pseudo-code for cost minimization with simulated annealing.

The cooling schedule used in this application started with a temperature of 1.0 which decreased at a rate of 10%. The total number of iterations at a given temperature was limited to 100 * J, and the maximum number of positions visited at a given temperature was 10 * J.

The cooling schedule is important since it determines the rate of convergence of the algorithm and the quality of the results. New configurations are generated randomly from the current configuration. In choosing a new configuration each position in the vector had a 40% chance of being modified. Those positions chosen for modification had a 25% chance of being incremented, a 25% chance of being decremented, and a 50% chance of staying the same.

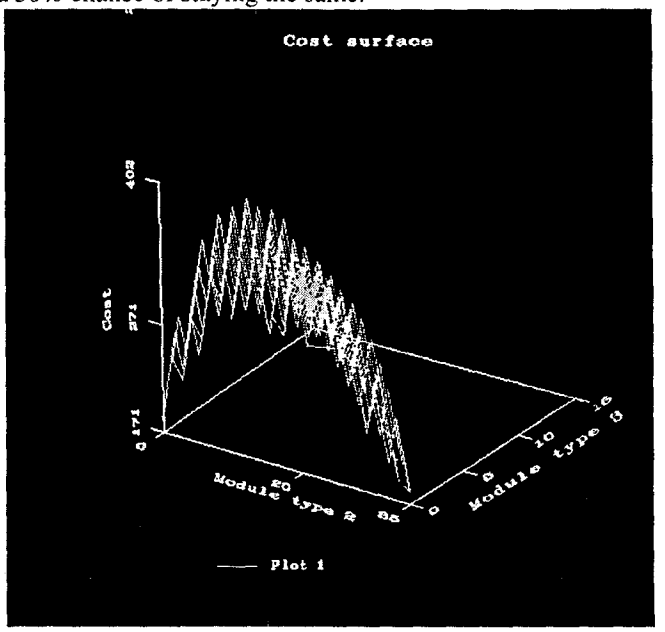

Figure 6 Cost versus components of types 2 and 3.

4. *EXPERIMENTAL RESULTS*

Figure 6 shows the shape of the search space found by using an exhaustive search algorithm on a sample problem consisting of three component types. Note the jagged nature of the search space and the large number of local minima present. This illustrates the non-linear nature of the problem and the need for using heuristics which are resistant to local minima.

Example 1:

	C. 1	C. 2	C. 3	C. 4	C. 5	C. 6	C. 7	C. 8
Failure	0.03	0.05	0.05	0.01	0.08	0.03	0.04	0.01
Repair	0.45	0.95	0.3	0.05	0.45	0.97	0.3	0.1
Cost	22.06	25.00	12.25	10.00	9.50	36.70	15.50	15.10
Solo	5	5	9	11	11	3	7	7
Cost	110.30	125.00	110.25	110.00	104.50	110.10	108.50	105.70
Min.	0	1	0	0	0	0	0	4
Cost	85.40							
Tabu	0	1	0	0	0	0	0	4
Cost	85.40							
SA	0	1	0	0	0	0	0	4
Cost	85.40							

Example 1 is a test case of multiple dimensions used to test the tabu search and simulated annealing approaches to this problem. It has eight dimensions. The number of dimensions was kept relatively small to allow the use of an exhaustive search algorithm to verify the global minimum. Tabu search and simulated annealing both succeeded in finding the global minimum. Tabu search also found the optimal answers but suffers from lacking a clear stopping criteria. While these results are positive, it should be noted that both heuristics can not be guaranteed to provide the globally optimal answer.

5. *DISCUSSION*

The methods described in this paper are useful for finding system configurations for high reliability systems made up of individual components and relying on fault masking. They use equation (2) to verify that the system fulfills dependability constraints when component failures are statistically independent. An example has been given to illustrate how this methodology can be put into practice and result in cost savings.

Note that example 1 has provided savings of approximately 20% compared to the lowest cost solo configuration. This is a sizable improvement, whether cost is defined as dollar amounts or component weight. The dual problem of the problem studied here, maximizing system reliability within fixed cost or weight bounds, is equally important and can be solved by switching the cost function and constraint functions proposed in this article. In our tests, simulated annealing performed remarkably well and appears to be the methodology best suited to solving the problem. A reasonable approach would be to use both approaches to verify the results found.

It should be noted, however, that tabu search and simulated annealing both have drawbacks. Tabu search has no clear stopping criteria. Both methods are relatively insensitive to the presence of local minima in the search space. With simulated annealing this insensitivity is partially obtained by the creative application of nondeterminism. This nondeterminism also means that the quality of the answers found by the algorithms will vary from case to case. It is also impossible to know how long either algorithm will need to find the global minima, or if it will ever find the global minima.

ACKNOWLEDGMENT

This work was supported in part by the Office of Naval Research grant N 00014-94-1-0343. Portions of this research has appeared in the Ph. D. dissertation in Ref. 3.

REFERENCES

1. J. G. Balchen and F. Dessen "Structural Solution of Highly Redundant Sensing in Robotics Systems" in *Highly Redundant Sensing in Robotics Systems* (ed.s Tou and Balchen) NATO Advanced Science Institutes Series vol. F-58, Springer Verlag, pp. 263-275, 1990.

2. M. Barborak, M. Malek and A. Dahbura, "The Consensus Problem in Fault Tolerant Computing" *ACM Computing Surveys* 25, 2(June), pp. 171-220, 1993.

3. R. R. Brooks, *Sensor Fusion Algorithms: Calibration and Cost Minimization*, Ph. D. Dissertation, Louisiana State University, June 1996.

4. R. R. Brooks and S. S. Iyengar, "Methods of Approximate Agreement for Multisensor Fusion", *Signal Processing, Sensor Fusion and Target Recognition IV*, eds. I. Kadar and V. Libby, SPIE, Bellingham, Wa., vol. 2484, Proceedings of SPIE International Symposium on Aerospace /Defense Sensing and Dual-use Photonics, Orlando, Fla. April 1995, pp. 37-44, 1995

5. R. R. Brooks and S. S. Iyengar, "Approximate Agreement for Distributed Computing by Masking Data Conflicts", Accepted for publication *IEEE Computer*, 1995.

6. F. Glover, "Tabu Thresholding: Improved Search by Nonmonotonic Techniques", *ORSA Journal on Computing*, vol. 7, No. 4(Fall), pp. 426-442, 1995.

7. T. C. Hu, A. B. Kahng and C. A. Tsao, "Old Bachelor Acceptance: A New Class of Non-Monotone Threshold Accepting Methods", *ORSA Journal on Computing*, vol. 7, No. 4(Fall), pp. 417-425, 1995.

8. S. S. Iyengar, L. Prasad, and H. Min, *Advances in Distributed Sensor Integration: Applications and Theory*, Prentice Hall, 1995.

9. D. N. Jayasimha, "Fault Tolerance in a Multisensor Environment." Technical Report, Department of Computer Science Ohio State University, Fall 1993.

10. I. Kadar "Adaptive Sensor Fusion" in *Signal Processing, Sensor Fusion, and Target Recognition IV* (eds. Kadar and Libby), SPIE proceedings vol. 2484, pp. 75-82, April 1995.

11. T. Krol, "(N,K) Concept Fault Tolerance" *IEEE transactions on Computers* Vol. C-35, No 4(April), 339-349, 1986.

12. P. J. M. van Laarhoven and E. H. L. Aarts, *Simulated Annealing: Theory and Applications*, D. Reidel Publishing Co. , Dordrecht, 1987.

13. L. Lamport, R. Shostak, and M. Pease "The Byzantine Generals Problem", *ACM Trans. Program. Lang. Syst.*, 4, 3(July), pp. 382-401, 1982.

14. Y. Levendel "Fault Tolerance Cost Effectiveness" in *Hardware and Software Architectures for Fault Tolerance: Experience and Perspectives* (eds. Banâtre and Lee), Springer Verlag, pp. 15-20, 1994.

15. R. Luo and M. Kay "Data Fusion and Sensor Integration: State-of-the-art 1990s" *Data Fusion in Robotics and Machine Intelligence* Abidi and Gonzales ed.s, Academic Press, Boston, pp. 7-136, 1992.

16. K. Marzullo, "Tolerating failures of continuous-valued sensors." *ACM Transactions on Computer Systems*, 8 (4):284-304, 1990.

17. F. Maurin "Keynote Address" in *Fault Tolerant Design Concepts for Highly Integrated Flight Critical*

Guidance and Control Systems NATO Advisory Group for Aerospace Research and Development Conference Proceedings No. 456 AGARD-CP-456, pp. KE-1-KE-4, 1990.

18. J. L. Michaloski, P.G. Backes, and R. Lumin "Integration of Sensor Feedback and Teleoperation into an Open Standard" in *Sensor Fusion and Networked Robotics VIII* (eds. Schenker and McKee), SPIE Proceedings vol. 2589, pp. 206-217, October, 1995.

19. W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in Fortran*, pp. 436-448, Cambridge University Press, 1986.

20. Roseaux, *Exercises et Problemes Resolus de Recherches Operationelle tomes 1-3*, Masson, Paris, 1987.

21. R. K. Saha "On Bayesian Attribute Fusion Using Offboard ID Sources" in *Signal Processing, Sensor Fusion, and Target Recognition IV* (eds. Kadar and Libby), SPIE proceedings vol. 2484, pp. 14-24, April 1995.

23. G. Strang, *Linear Algebra and Its Applications*, Academic Press, New York, 1976.

24. J. T. Tou "A Knowledge-Based System for Redundant and Multi-Sensing in Intelligent Robots" in *Highly Redundant Sensing in Robotics Systems* (ed.s Tou and Balchen) NATO Advanced Science Institutes Series vol. F-58, Springer Verlag, pp.3-14, 1990.

25. J. Wahrburg "Control Concepts for Industrial Robots Equipped with Multiple and Redundant Sensors" in *Highly Redundant Sensing in Robotics Systems* (ed.s Tou and Balchen) NATO Advanced Science Institutes Series vol. F-58, Springer Verlag, pp.277-291, 1990.

BIOGRAPHIES

Richard R. Brooks, *PhD*
California State University Monterey Bay
100 Campus Center
Seaside, California 93995-8001 USA

Richard R. Brooks (corresponding author, and presenting author) is Assistant Professor of Applied Computing at California State University Monterey Bay. He has a Ph.D. in Computer Science (1996) from Louisiana State University, and a bachelors degree in Mathematical Sciences (1979) from Johns Hopkins. Dr. Brooks also studied operations research and computer science at the Conservatoire National des Arts et Metiers (CNAM) in Paris. His work experience includes projects with Goddard Spaceflight Center, Radio Free Europe/ Radio Liberty Munich and the French stock exchange authority. As a consultant for the World Bank he helped implement their network in Africa, Eastern Europe and Central Asia. Dr. Brooks speaks French and German fluently. He is a member of ACM and INFORMS.

S. S. Iyengar, *PhD*
Department of Computer Science
Louisiana State University
Baton Rouge, Louisiana 70803-4020 USA

S. S. Iyengar (Fellow IEEE) is the Chairman of the Computer Science Department and Professor of Computer Science at Louisiana State University. He has been involved with research in high-performance algorithms and data structures since receiving his PhD in 1974, and has directed over 22 PhD dissertations at LSU. He has served as a principal investigator on research projects supported by the Office of Naval Research, the National Aeronautics and Space Administration, the National Science Foundation, California Institute of Technology's Jet Propulsion Laboratory, the Department of Navy-NORDA, the Department of Energy, LEQFS - Board of Regents and U.S. Army Office. His publications include several books and over 200 publications (including 105 archival journal papers). He is a series editor for *Neuro Computing of Complex Systems*, and an area editor for the *Journal of Computer Science and Information*. He has served as a guest editor for the *IEEE Transactions on Software Engineering*, the *IEEE Transactions on System, Man and Cybernetics* and the *IEEE Transactions on Knowledge and Data Engineering*. He is a fellow of the IEEE for his research contributions in the area of Algorithms and Data Structures in Image Processing.

Suresh Rai, *PhD*
Department of Electrical and Computer Engineering
Louisiana State University
Baton Rouge, Louisiana 70803-4020 USA

Dr. Rai is working with the Department of Electrical and Computer Engineering at Louisiana State University, Baton Rouge. Dr. Rai has taught and researched in the area of reliability engineering, fault diagnostics, and parallel and distributed processing. He is a co-author of the book *Waveshaping and Digital Circuits*, and tutorial texts *Distributed Computing Network Reliability* and *Advances in Distributed System Reliability*. He has guest edited a special issue of *IEEE Transactions on Reliability* on the topic *Reliability of Parallel and Distributed Computing Networks*. Dr. Rai has worked as a committee member for IPCCC '91 (Phoenix) conference. Currently, he is working as an Associate Editor for *IEEE Transactions on Reliability*. Dr. Rai received his B. E. degree from Benaras Hindu University in 1972, his M. E. from University of Roorkes in 1974, and his Ph.D. from Kurukshetra University in 1980. He joined the Regional Engineering College at Kurukshetra in 1974 and worked there for six years. After a brief stay at MMM Engineering College, Gorakhpur, he transferred to University of Roorkee in 1981 as an Associate Professor. He also worked for two years at the School of Engineering at North Carolina State University, Raleigh. Dr. Rai is a senior member of the IEEE, and member of the ACM.