Minimizing cost of redundant sensor-systems with an artificial immune algorithm

Conference Paper · February 2003

DOI: 10.1109/ICIF.2003.177413 · Source: IEEE Xplore

2 authors, including:

Q. M. Jonathan Wu
University of Windsor

Minimizing Cost of Redundant Sensor-Systems with an Artificial Immune Algorithm

Q. M. Jonathan Wu
National Research Council Canada
3250 East Mall, Vancouver, B. C.
Canada V6T 1W5
Jonathan.Wu@nrc.ca

Guiliang Yin
Electrical Engineering Department
Yashan University, 066004, China
guiliang.yin@nrc.ca

Abstract – *Using human immune system mechanisms, an artificial immune algorithm used for minimizing cost of redundant multi-sensor systems is presented in this paper. The search surface for finding the minimal cost of the sensor system while insuring system dependability has many local minima. The heuristic methods must be used to solve the problem. The artificial immune algorithm uses hyper-mutation to search local areas and receptor editing to escape from local minima. It is more suitable than simulated annealing and genetic algorithms for solving this problem. The results produced by the method are compared with results of simulated annealing and genetic algorithm.*

Keywords: Sensor fusion, dependability, artificial immune algorithm, optimum.

1 Introduction

Multi-sensor fusion and integration refers to the synergistic combination of sensory data from multiple sensors to provide more reliable and accurate information. The potential advantages of multi-sensors fusion and integration are redundancy, complement-arity, timeliness, and cost of the information. Redundant multi-sensor systems achieve fault tolerance by multiple inaccurate sensors. Feasibility requires attention be paid to both dependability bounds and cost. Success in designing redundant systems depends on making the best possible trade-off at least cost.

In [1]-[4], dependability measure methods of redundant systems are given in detail. After deriving the dependability expressions, an algorithm called exhaustive search guaranteed to find the minimal cost configuration using the item costs of each sensor is presented. But this algorithm is computationally expensive; it is just suitable for small-scale problems. For lager-scale problems, Tabu search, simulated annealing (SA) and genetic algorithms (GAs) are applied for finding near-optimal combination.

Tabu search is often used as an alternative to simulated annealing and it has no clear stopping criteria. Tabu search, simulated annealing and genetic algorithms are relatively insensitive to the presence of local minima in the search space. With simulated annealing and genetic algorithms this insensitivity is partially obtained by the creative application of non-determinism. This non-determinism also means that the quality of the answers found by the algorithm will vary from case to case. GAs are sensitive to the reproduction strategy chosen, including mutation rates and how elements are chosen for crossover. SA is sensitive to the cooling schedule, which includes the initial temperature and the rate of decrease of temperature. The quality of the answers found and the amount of time needed to find reasonable answers are directly dependent on the reproduction strategy of a GA and the cooling schedule of an SA approach. Both the reproduction strategy and cooling schedule must be found through a process of trial and error. For neither is there a guarantee that a particular strategy nor schedule will be appropriate for all cases encountered.

The immune system, with its cell diversity and variety of information processing mechanisms, is a cognitive system of complexity comparable to the brain. Interest in studying the immune system has been increasing over the last few years. Several optimal methods using artificial immune mechanism are presented recently [5-11]. A comparison of optimization performances between immune algorithm and genetic algorithms is given in reference [5] and immune algorithm shows a better result than genetic algorithm for a problem having many local optima.

In this paper, we present an artificial immune algorithm (IA) for the problem, based on the way the immune system's T-cell-dependent responses. We extend the idea to adapt the problem of minimizing cost of redundant sensor systems.

2 Dependability Measure and Optimization Model

Redundant sensor-systems achieve fault tolerance by duplication of components. It increases the ability of systems to interact with their environment by combining independent sensor readings into logical representations. 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; 3) Sensor efficiency and performance can be enhanced; 4) Self-calibration can be attained. But the feasibility of the systems requires attention be paid to both reliability bounds and cost.

The problem we have to solved is: for a redundant sensor system, given a choice between various modules, each with different dependability parameters and per item costs, which combination of modules meets the reliability requirements with the lowest system cost? In [1]-[4], the dependability measures and optimization model are presented in detail. Here we just give the results.

Assume that a system contains N identical sensors and the sensor failure is statistically independent. If each sensor has an identical probability of functioning $r(t)$ (if a sensor has a constant failure rate λ, the reliability for that sensor will be $e^{-\lambda t}$) at time t and a probability of being faulty $q(t)$ where $q(t)=1-r(t)$. The probability of i out of N sensors working at time t as:

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

The reliability for the system is the summation of the terms with i varying from N to $N/2+1$.

This approach is easy to apply when more than one type of sensor is used. Equation (2) is the reliability of a system with two sensor types. N_1 (N_2) is the number of sensors of type 1 (2). $r_1(t)$ and $r_2(t)$ are the reliabilities of sensor type 1 and type 2.

$$R(t)=\sum_{k=0}^{N_1}\left[\binom{N_1}{k} r_1^k (1-r_1)^{N_1-k} \times \sum_{m=\frac{N}{2}+1-k}^{N_2}\left(\binom{N_2}{m} r_2^m (1-r_2)^{N_2-m}\right)\right] \tag{2}$$

This can be extended to more than two component types. Evaluating a combination of J different types of sensors requires J levels of summations in the format of Eq. (2).

Given J different component types that meet a system's requirements. Finding the combination of components that meets dependability requirements with the lowest total cost, requires considering each combination of the J component types as a point in a discrete J-dimensional space. This point is described by the J-dimensional vector (Q_1, Q_2, ..., Q_J), where Q_i corresponds to the number of components of type i included in the system. Since each component has a known per item cost, if component type i has cost c_i , the cost of this type of components in the system is c_i*Q_i. The question remains as to how to find the combination that minimizes the fitness function:

$$\sum_{i=0}^{J} c_i Q_i \tag{3}$$

This is a combinatorial optimization problem. Unfortunately it cannot be solved by techniques such as linear programming or integer programming. These techniques are inappropriate since Eq. 2, which defines dependability requirements, is nonlinear. Another approach is needed.

Figure 1 [1] shows the shape of the search space found by using an exhaustive search algorithm on a sample problem consisting of three component types. The surface is represented by a series of jagged lines. The jaggedness of the lines is caused by two factors: the problem space is discrete, and tolerating failures of less than 50% of the components means that adding two components to increases system reliability is necessary. Adding one component increases the number of items that may fail without increasing the number of failures that may be tolerated. This jaggedness indicates that the search space has many local minima and search methods that depend only on information in the neighborhood of a point will be unsuited to solving this problem. Although the GAs and SA obtain good results for this problem [1], immune algorithm shows a better result than genetic algorithm for a problem having many local optima [5].

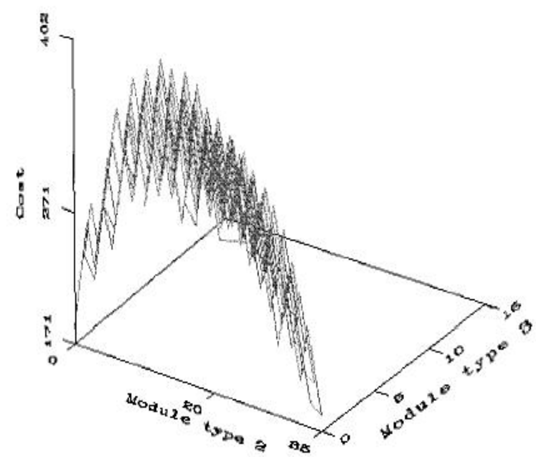

Fig. 1 Cost of resulting system versus number of components of types 2 and 3 needed to fulfill constraints

3 Immune Algorithm

The immune system is our basic defense system against bacteria, viruses and other disease-causing organisms. Our body maintains a large number of immune cells-called lymphocytes, which circulate throughout the body. There are mainly two types of lymphocytes, namely T cells and B cells. These two types of lymphocytes play different roles in the immune response, though they may act together and control or affect one another's function. When an antigen invades the body, only a few of these immune cells can recognize the invader's peptides. This recognition stimulates proliferation and differentiation of the cells that produce matching clones. This process, called clonal expansion, generates a large population of antibody-producing cells that are specific to the antigen. The clonal expansion of immune cells result in destroying or neutralizing the antigen. It also retains some of these cells in immunological memory, so that any subsequent exposure to a similar antigen leads to rapid immune response. These mechanisms are dramatic and complex. By using these mechanisms, immune algorithm shows a good performance as an optimization algorithm.

There are two mechanisms of natural immune response that are usually used in solving optimal problems. They are hyper-mutation and receptor editing [7]. Antibody's present in a memory response have, on average, a higher affinity than those of the early primary response. This phenomenon, which is restricted to T-cell-dependent responses, is referred to as the maturation of the immune response. This maturation requires the antigen-binding sites of the antibody molecules to be structurally different from those present in the primary response. Random changes are introduced into the genes responsible for the antigen–antibody interactions and occasionally one such change will lead to an increase in the affinity of the antibody. These higher affinity variants are then selected to enter the pool of memory cells. Not only the repertoire is diversified through a hyper-mutation mechanism, but also mechanisms must exist such that rare B cells with high affinity mutant receptors can be selected to dominate the response. Those cells with low affinity or self-reactive receptors must be efficiently eliminated, become anergic (with no function), or be edited. The hyper-mutation guides to local optima, and receptor editing offers the ability to escape from local optima on an affinity landscape. This guarantees the immune algorithm can get global optimum.

The solution space to the minimizing cost of redundant sensor system consists of component configurations. For this reason, the antibodies used by the immune algorithm consist of a vector that describes a possible system configuration. Position i of the vector, where i is between 1 and J, is an integer ranging from 1 to N_i giving the number of components of type i in the system. The number of elements of type 1 is calculated by the program and set to the smallest number of components of type 1 needed to fulfill the system dependability requirements. An antibody's affinity corresponds to the evaluation of the objective function. The immune algorithm for minimizing the cost of redundant system is described below:

Step 1: Compute values N_i for all of the J component types under consideration. These are the number of components of type i needed to meet the dependability requirement D when no components of any other type are used;

Step 2: Sort component types in increasing order of the value N_i*c_i;

Step 3: Randomly initialize a population of anti-bodies (feasible solutions to the problem);

Step 4: While stopping criterion is not met do

Step 4.1: Determine the fitness of each antibody and normalize the vector of fitness;

Step 4.2: Generate a number Nc of clones for each antibody;

Step 4.3: Mutate each clone proportionally to the fitness of its parent antibody, but keep the parent antibody;

Step 4.4: Determine the fitness of all individuals of the population;

Step 4.5: For each clone, select the antibody with highest fitness and calculate the average fitness of the selected population;

Step 4.6: If the average error of the population is not significantly different from the previous iteration, then continue. Else, return to step 4.1;

Step 4.7: Determine the affinity of all antibodies. Suppress all but the highest fitness of those antibodies whose affinities are less than the suppression threshold σ_s and determine the number of antibodies, named memory antibodies, after suppression;

Step 4.8: Introduce a percentage d% of randomly generated antibodies and return to step 4;

Step 5: End while.

From step 4.1 to 4.5, a population of antibodies is optimized locally through affinity proportional mutation (exploitation of the fitness landscape). From 4.6 to 4.8, when this population reaches a stable state (measured via the stabilization of its average fitness), the antibodies interact with each other, and some of the similar antibodies are eliminated to avoid redundancy. Also, a number of randomly generated antibodies are added to the current population (exploration of the fitness landscape) and process of local optimization restart.

4 Experiment Results

For comparison with the genetic algorithms and simulated annealing, we use the same experiment parameters that reference [1] uses to test IA approach to this problem. Tables 1 and 2 show the results of these experiments. IA succeeded in finding the global minimum in both cases. GA did not find the global minimum in experiment 1. SA also found the global minimum in both cases, but SA is not guaranteed to provide the globally optimal answer.

Figure 2 shows the run times of the four approaches for problems consisting of seven, eight and nine components. The graph shows the number of second required versus the number of components considered. SA was the fastest approach in our tests. IA was a little bit slower than GA. But it found the correct answers and has the stopping criterion.

Table 1 IA test result of eight dimensions
Minimum cost: 96.62   Availability constraint: 99.99999995%

Iterations	Comp.1	Comp.2	Comp.3	Comp.4	Comp.5	Comp.6	Comp.7	Comp.8	Current_Cost
1	0	4	0	0	0	0	5	14	175.84
20	0	0	6	0	3	0	0	17	163.36
40	5	1	0	0	0	0	0	5	137.72
60	0	0	0	7	4	0	1	0	122.20
80	0	0	2	0	7	0	0	3	100.52
100	0	4	0	0	4	1	0	0	99.14
115	0	7	0	0	1	0	0	1	96.62

Table 2 IA test result of eleven dimensions
Minimum cost: 100.80   Availability constraint: 99.99999995%

Iterations	C.1	C.2	C.3	C.4	C.5	C.6	C.7	C.8	C.9	C.10	C.11	Current_Cost
1	0	8	3	0	2	0	0	0	0	0	0	190.00
25	0	8	1	0	0	3	0	0	0	0	1	152.00
50	0	1	0	0	0	2	0	0	3	0	6	144.10
75	3	0	0	0	0	0	0	0	1	2	6	136.30
100	0	0	0	0	1	0	0	2	1	3	7	131.10
125	0	4	0	9	0	0	0	3	0	0	0	109.01
150	0	4	0	9	0	0	0	3	0	0	0	109.01
175	0	0	0	9	0	1	0	0	1	0	4	108.71
200	0	0	0	9	0	2	4	0	0	0	0	103.01
225	0	0	0	0	1	0	7	0	0	4	0	101.20
241	0	0	0	2	0	0	12	0	0	1	0	100.80

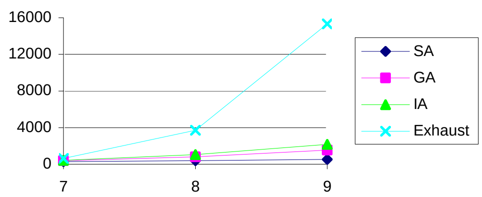

Fig. 2 Run times of exhaustive search SA, GA and IA

5 Conclusion

This paper presents an artificial immune algorithm for minimizing cost of redundant multi-sensor systems. It approaches optimization by hypermutation of the antibody, and utilizes receptor editing to escape from local optima on an affinity landscape. The artificial immune algorithm shows good results in minimizing cost of redundant multi-sensor system. Although it is more computational than GA, IA succeeded in finding better results than GA.

References

[1] R. R. Brooks, S. S. Iyengar, Multi-Sensor Fusion: Fundamentals and Applications with Software, p, Upper Saddle River, N.J.:Prentice Hall, 1998.
[2] R. R. Brooks, S. S. Iyengar, Suresh Rai, Comparison of genetic algorithms and simulated annealing for cost minimization in a multi-sensor system, Optical Engineering, Vol. 37, No.2, pp. 505-516, Feb. 1998.
[3] R. R. Brools, S. S. Iyengar, Suresh Rai, Minimizing Cost of Redundant Sensor-Systems with Non-Monotone and Monotone Search Algorithms, 1997 Proceedings Annual Reliability and Maintainability Symposium, pp.307–313.
[4] R. R. Brools, S. S. Iyengar, Maximizing Multi-Sensor System Dependability, Proceedings of the 1996 IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems, pp. 1-8.
[5] Jang-Sung Chun, Hyun-Kyo Jung and Song-Yop Hahn, A Study on Comparison of Optimization Performances between Immune Algorithm and other Heuristic Algorithm, IEEE Transactions on Magnetics, Vol. 34, No. 5, Sept. 1998, pp. 2972-2975.
[6] Leandro N. de Castro and Jon Timmis, An artificial immune network for multimodal function optimization, IEEE Proceedings of the 2002 Congress on Evolutionary Computation, 2002, Vol. 1, pp. 699-704.

[7] Leanfro N. de Castro and Fernando J. Zuben, Learning and Optimization Using the Clonal Selection Principle, IEEE Transaction on Evolutionary Computation, Vol. 6, No. 3, June 2002, pp. 239-251.
[8] Alessio Gaspar and Philippe Collard, From GAs to artificial immune systems: improving adaptation in time dependent optimization, Proceedings of the 1999 Congress on Evolutionary Computation, 1999, Vol. 3, pp.1859 - 1866
[9] Alessio Gaspar and Philippe Collard, Two models of immunization for time dependent optimization, 2000 IEEE International Conference on Systems, Man, and Cybernetics, Vol. 1, 2000, pp. 113-118.
[10] V. A. Huznetsov, G. D. Knott, A. V. Ivshina, Artificial immune system based on syndromes-response approach: recognition of the patterns of immune response and prognosis of therapy outcome, 1998 IEEE International Conference on Systems, Man, and Cybernetics, 1998, Vol. 4, 3804-3809.
[11] Lei Wang and Licheng Jia, A novel genetic algorithm based on immunity, ISCAS 2000 IEEE International Symposium on Circuits and Systems, 2000, pp. 385-388.