ESTIMATION OF SOFTWARE RELIABILITY BY SEQUENTIAL TESTING WITH SIMULATED ANNEALING OF MEAN FIELD APPROXIMATION

Nidhi Gupta and Manu Pratap Singh*

Department of Computer Science, Institute of Basic Science
Khandari, Dr. B. R. A. University, AGRA, India
manu_p_singh@hotmail.com

*Corresponding Author

(Received: July 29, 2004 – Accepted: November 2, 2006)

Abstract Various problems of combinatorial optimization and permutation can be solved with neural network optimization. The problem of estimating the software reliability can be solved with the optimization of failed components to its minimum value. Various solutions of the problem of estimating the software reliability have been given. These solutions are exact and heuristic, but all the exact approaches are of considerable theoretical interest. In this paper, we propose the simulated annealing technique of mean field approximation for finding the possible minimum number of failed components in the sequential testing. These minimum numbers of failed components are depending upon the selection of time intervals or slots. The selection of time intervals or slots satisfies all the necessary constraints of the problem. The new energy function with the mean field approximation is also proposed. The constraint parameter for the annealing schedule is also dynamically defined that is changed with the selection of a time interval or slot on each iteration of the processing. The algorithm of the whole process shows that this approach can generate the optimal solution.

Key Words Software Reliability, Mean Field Approximation, Simulated Annealing, Optimization

چکیده مسائل مختلف بهینه سازی ترکیبی و ترتیبی را می‌توان با بهینه سازی توسط شبکه های عصبی حل کرد. مساله تخمین قابلیت اعتماد می‌تواند با بهینه سازی اجزاء معیوب تا مقدار مینیمم آن حل شود. راه های مختلفی برای تخمین قابلیت اعتماد نرم افزار داده شده است. این پاسخ ها دقیق و ابتکاری هستند اما تمام روش های دقیق به صورت نظری مورد علاقه فراوان می‌باشند. در این مقاله روش شبیه سازی سرد شدن فولاد مربوط به تقریب میدان متوسط برای بدست آوردن حداقل تعداد از اجزاء معیوب در آزمون های ترتیبی ارائه می‌شود. این حداقل تعداد از اجزاء معیوب به انتخاب فاصله زمانی بستگی دارد. انتخاب فاصله زمانی (یا شیار زمانی) کلیه محدودیت های لازم مساله را برآورده می‌سازد. یک تابع جدید انرژی برای تقریب میدان متوسط نیز ارائه شده است. پارامتر محدودیت برای برنامه "سرد شدن" نیز به صورت پویا تعریف شده به طوری که با انتخاب فاصله زمانی (یا شیار زمانی) با هر تکرار از فرآیند تغییر می‌کند. الگوریتم تمام فرآیند نشان می‌دهد که این شیوه می‌تواند پاسخ بهینه را ایجاد کند.

1. INTRODUCTION

Most of the traditional problems of combinatorial permutation can be solved with the help of the Artificial Neural Network [1]. There are various applications and uses of a neural network [2]. One of the most successful applications of the neural network is solving optimization problems [3]. There are many situations where a problem may be formulated as minimization or maximization of the cost function or objective function subject to constraints. It is possible to map such problems onto a feedback network, where the units and connection strengths are identified by comparing the cost function of the problem with the energy function of the network expressed in terms of the states values of the units and the connection strength. The software reliability can be defined in statistical terms as “the probability of fault free operation of a system or component to perform its required functions under stated conditions for a specified period of time” [4]. Software reliability represents a user (or customer) oriented view of software quality. It relates directly to operation rather than design of a program, and hence it is dynamic rather than static. For this reason software reliability is interested in failures occurring and not faults in a program. Failures mean a function of the software that does not meet user requirements .The reliability of any software can be estimated using the failure data of the system, but prediction of exact failures in software is complicated. We have various statistical methods [5] for predicting failures but research indicates that these models are not appropriate for prediction of failure. It may be a better idea to use the optimization technique of ANN for determining the failures and use it for estimating the software reliability.

Various solutions for such types of problems have been proposed in which the problems can be formulated as the optimization of the cost or function with the satisfaction of certain constraints. The same method can also be applied to optimize the reliability estimation, where the objective is to minimize the number of failed components in sequential testing [6-7]. The traditional method to solve such problems was gradient decent approaches of hill climbing and a stochastic simulated annealing [8].

In this paper, we propose the simulated annealing technique of mean field approximation for optimizing the possible minimum number of failed components in the given time duration. The selection of time intervals or slots satisfies all the necessary constraints. Energy function for any Hopfield type feedback neural network represents the stored input pattern in the form of the number of failed components. This energy function also satisfies all the necessary constraints imposed in the problem. A global constraint in the form of the number of failed components in the particular slot or time interval (that is being selected randomly) can be selected for the Annealing schedule. The constraints can be reduced as per annealing schedule and the corresponding the new energy function is estimated. It can be seen that the possible minimum energy function will represent the minimum number of failed components in the sequential testing.

2. SIMULATED ANNEALING OF MEAN FIELD APPROXIMATION FOR OPTIMIZING THE NUMBER OF FAILED COMPONENTS OF THE SOFTWARE

To optimize the maximum or minimum number of failed components in the testing period we can consider the optimization method of simulated annealing. The optimization is a process in which a function can be formulated as minimization or maximization of some cost function or objective function subject to certain constraints. The solution of the problem is generally defined in two terms, one is global optimal solutions and other is local the optimal solution. Global optimal solution is used when there is no other feasible solution with better objective-function values. A local optimal solution is when there is no other feasible solution in the vicinity with better objective function values. In the testing of the different software components, the objective is to minimize the number of failed components in the given time duration (T) under certain conditions. We divide the total time duration in to different small time interval and note all the number of failed components at the end of that slot. Therefore, we can consider the different time intervals for the execution of software. For example we are noting the number of each failed component after one hour, in other words we can note the number of failed components after 2 hour and so on. Now, the question is which time interval should be selected so that the number of failed components could be minimized with the constraints that each slot should come only once in the testing period. The Hopfield memory can be used to solve this problem. In this process the characteristic of interest is the rapid minimization of the energy function. To use the Hopfield memory for the application, we map the problem onto the Hopfield type network architecture. The first term is to develop a representation of the problem of the solution that fits in an architecture having a single array of the processing elements (PE). We develop it by allowing a set of N PEs to represent the N possible positions for a given slot in the sequence of the testing. The weight matrix format can be found from the number of failed components in that particular slot. The output will be labeled as Tx_i, where the 'X', subscript refers to the slot and the 'i', subscript refers to the position in the testing. To formulate the connection weight matrix, the energy function must be constructed so that it satisfies the following criteria:

Energy minima must favor states that have each slot only once in the sequential testing.
Energy minima must favor states that have each position in the testing only once.
Energy minima must favor states with the minimum total number of failed components.

Denoting the state of a processing unit of the Hopfield network as $Tx_i = 1$ indicates that the slot X is to be completed at the ith stage of testing, the energy function can be written as;

$$E = \frac{A}{2}\sum_{X=1}^{N}\sum_{i=1}^{N}\sum_{j=1,j\neq 1}^{N} T_{X_i}T_{X_j} +$$

$$\frac{B}{2}\sum_{i=1}^{N}\sum_{X=1}^{N}\sum_{Y=1,Y\neq X}^{N} T_{X_i}T_{Y_i} +$$

$$\frac{C}{2}\sum_{X=1}^{N}\sum_{Y=1,Y\neq X}^{N}\sum_{i=1}^{N} f_{xy}T_{X_i}(T_{Y,i+1}+T_{Y,i-1}) \tag{2.1}$$

where A, B and C are the positive constants and denote the relative importance given to the constraints.

The mean field approximation algorithm [9-10] also proposed a solution for this type of optimization. The following energy function with a mean field approximation can be proposed:

$$E_{ST} = \frac{f_{max}}{2}\sum_{i}^{N}\sum_{j\neq i}^{N}\sum_{X}^{N} T_{X_i}T_{X_j} +$$

$$\frac{1}{2}\sum_{X}^{N}\sum_{Y\neq X}^{N}\sum_{i}^{N} f_{XY}T_{X_i}\left(T_{Y,i}+T_{Y,i-1}\right) \tag{2.2}$$

Where f_{max} is real constant, which is slightly larger than the largest number of failed components between the time intervals in the given sequential testing. There fore, in Equation 2.2, there are only two terms, the first term is regarding feasibility, which inhibits two slots from being in the same testing time. The second summation term is used for the minimization of the number of failed components. The term f_{max} is used for the balancing of the summation terms.

The output Tx_i of a neuron (X, i) is interpreted as the probability of finding slot X in testing duration at position i. The mean field for a neuron (X, i) is defined according to the energy function given in Equation 2.2 as;

$$E_{x_i} = f_{max}\sum_{Y\neq X}^{N} T_{X_i} + \sum_{Y\neq X}^{N} f_{XY}(T_{Y,i+1}+T_{Y,i-1}) \tag{2.3}$$

Initially all the neurons are arranged to the average value and the constraint parameter F with the f_{max}. The weights of the interconnections are initialized with small random numbers. As per the Hopfield model, the stable state of the any ith neuron can be define as

$$T_i\,(t+1) = F'\,[\,\sum_{j\neq i} W_{ij}T_j(t)\,] \tag{2.4}$$

This stable state may lead to a state corresponding to a local minimum of the energy function. In order to reach the global minimum, i. e. the possible minimum number of failed components in the sequential testing, by passing the local minima, we use the concept of stochastic updating of the unit in the activation dynamics of the network. In stochastic updating, the state of units is updated using the probabilistic updating, which is controlled by the annealing schedule constraint parameter ($F = f_{max}$) and the probability distributions of the states at thermal equilibrium is given as;

$$P(\,T_{X_i}) = \frac{1}{Z}e^{-E_{X_i}/F} \tag{2.5}$$

Where Z is the partition function.

Initially, the arbitrary time slot is selected and the energy function of the network is constructed, as given in Equation 2.2. The annealing schedule assigned with the maximum number of failed components i. e. to its higher value. So at a higher F, many states are likely to be visited, irrespective of the energies of those states. Now, the value of the constraint parameter F is gradually reduced as per the annealing schedule, and the output value of the states perturb. This perturbation continues until the network settles in to a stable state or equilibrium state. So, the network estimates the energy function for this state and compares it with the previous energy function by computing ΔE. If $\Delta E \leq 0$, we accept the solution with the highest probability i.e. 1, otherwise we accept it with the probability given in the Equation 2.5. On each iteration of this process, the solution is accepted with the probability 1, and then the constraint parameter F is assigned with f_{max} of the newly constructed energy function. Then every time on the acceptable solution with higher probability, the constraints parameter changes with the newly found maximum number of failures. This newly found maximum number of failed components will be less then the previously found maximum number of failed components. So the simulated annealing process will continue with the new value of the constraint parameter. This process continues until the final value of the schedule is obtained. At this state the units of the network represent the state of equilibrium, which will represent the minimum energy function for the network. Thus the minimum energy function will represent the possible minimum number of failed components in the sequential testing.

In order to speed up the process of the simulated annealing, the mean field approximation is used, in which the stochastic updating of the binary units is replaced by deterministic analog states. Thus the fluctuating activation value of each unit is replaced with its average value. The Equation 2.4 can be express with this method as:

$$< T_i(t+1) > = F'[< \sum_{j \neq i} W_{ij} T_j(t) >] = F' \sum_{j \neq i} W_{ij} < T_j(t) > \quad (2.6)$$

and from the stochastic updating with thermal equilibrium we have;

$$< T_i(t+1) > = \tanh \left[\frac{1}{F} \sum_{j \neq i}^{N} W_{ij} < T_j(t) > \right] \quad (2.7)$$

This equation is solved iteratively starting with some arbitrary values $< T_i(0) >$ initially. The stable state in the mean field approximation is a result of minimization of an effective energy and defined as the function of F:

$$< T_i > = \tanh \left[-\frac{1}{F} \frac{\partial E(< T_i >)}{\partial < T_i >} \right] \quad (2.8)$$

Where the effective energy E ($< T_i >$) is the expression for energy function of the Hopfield model using averages for the state variables.

Now the constraint parameter F decreases as per the annealing schedule, the state of the network perturbs with the stochastic asynchronous updating of the processing elements. So, as the output value perturbs, the neurons produce the updated states. The states updating continues until the Equation 2.6 satisfied.

Hence, for stable conditions the energy function of the annealing schedule can be expressed as;

$$E_{ST} = \frac{f_{max}^{new}}{2} \sum_{i}^{N} \sum_{j \neq i}^{N} \sum_{X}^{N} < T_{X_i} >< T_{X_j} > +$$

$$\frac{1}{2} \sum_{X}^{N} \sum_{Y \neq X}^{N} \sum_{i}^{N} f_{XY} < T_{X_i} > \left(< T_{Y,i} + T_{Y,i-1} >\right) \quad (2.9)$$

The energy difference ΔE is computed as;

$$\Delta E = E_{ST}^{new} - E_{ST}^{old} = \frac{f_{max}^{new}}{2} \sum_{i}^{N} \sum_{j \neq i}^{N} < T_{X_i}^{new} >$$

$$< T_{Xj}^{new} > - \frac{f_{max}^{old}}{2} \sum_{i}^{N} \sum_{j \neq i}^{N} < T_{X_i}^{old} >< T_{X_j}^{old} > +$$

$$\frac{1}{2} \sum_{X}^{N} \sum_{Y \neq X}^{N} \sum_{i}^{N} (f_{XY}^{new} - f_{XY}^{old})(< T_{X_i}^{new} - T_{X_i}^{old} >)$$

$$\left[\left(< T_{Y,i+1} + T_{Y,i-1} >\right)^{new} - \left(< T_{Y,i+1} + T_{Y,i-1} >\right)^{old}\right]$$

$$(2.10)$$

If the energy difference $\Delta E \leq 0$, the probability of accepting the solution becomes 1 otherwise it is

$P_{mp} \alpha e^{-\Delta E/F}$ On each iteration with the highest probability of accepting the solution, the constraint parameter of the Annealing Schedule is defined as;

$$F = f_{max}^{new} \tag{2.11}$$

On each iteration, the constraint parameter is changed to its minimum value with respect to the previous value. Hence, each time the network determines the stable condition with the new value of energy function E and constraints parameter F (when PE < = 0). The neurons produce the stable state that represents the position of the slot with a minimum number of failed components with respect to the previous position. So, at stable condition with $\Delta E \leq 0$ the state of the neurons can be defined from the Equation 2.5

$$< Tx_i > = \tanh [- \frac{1}{f_{max}^{new}}$$

$$\frac{\partial [\frac{f_{max}^{new}}{2} \sum_{i}^{N} \sum_{j \neq i}^{N} \sum_{X}^{N} < Tx_i > < Tx_j > + \frac{1}{2} \sum_{X}^{N} \sum_{Y \neq X}^{N} \sum_{i}^{N} f_{XY} < T_{X_i} > (< T_{Y,i+1} + T_{Y,i-1} >)]}{\partial < T_{X_i} >}] \tag{2.12}$$

The Mean field for the neuron (X, i), where output $< T_{Xi} >$ can be interpreted as the probability of finding the slot X in the ith testing position as defined from the Equation 2.3 as;

$$E_{X_i} = f_{max}^{new} \sum_{Y \neq X}^{N} < T_{X_i} > + \sum_{Y \neq X}^{N} f_{XY} (< T_{Y,i+1} + T_{Y,i-1} >) \tag{2.13}$$

Thus, the entire process continues until a fixed point is found for the every value of constraint parameter F. In the process of selecting a fixed point, change in the energy is computed and the probability of accepting the point in the minimum number of failed components can be found. The field for the selected points will also be computed. This entire process will continue for every schedule of the constraints parameter F, until F reaches the final value.

The algorithm of the entire process can be proposed as:

Initialize the weights and bias with a small random number. The value of F is initialized with any large random number.

Store the input pattern in the form of the number of failed components $(f_1, f_2, ---, f_n)$ that are observed during the different time slots i.e. $(\Delta t_1, \Delta t_2, ---, \Delta t_n)$ of sequential testing for the different modules of the software. The number of failures $(f_1, f_2, ---, f_n)$ is stored with the stable state of the network as:

$$T_i(t+1) = F' [\sum_{j \neq i} W_{ij} T_j(t)]$$

Do until the fixed point is found.

Randomly select a slot say X.

Compute the energy function as;

$$E_{ST} = \frac{f_{max}}{2} \sum_{i}^{N} \sum_{j \neq i}^{N} \sum_{X}^{N} < T_{X_i} >< T_{X_j} > +$$

$$\frac{1}{2} \sum_{X}^{N} \sum_{Y \neq X}^{N} \sum_{i}^{N} f_{XY} < T_{X_i} > \left(< T_{Y,i} + T_{Y,i-1} > \right)$$

Compute the state of the unit at equilibrium;

$$< Tx_i >= \tanh[- \frac{1}{f_{max}} \frac{\partial E_{ST}}{\partial < T_{X_i} >}]$$

Also calculate the ΔE using Equation 2.7:

if $\Delta E \leq 0$ accept with $P_{< T_{X_i} >}^{(accept)} = 1$ and set F = f_{max} else $P_{< T_{X_i} >}^{(accept)} = \exp(\Delta E / F)$ Compute the Mean field as

$$Ex_i = f_{max} \sum_{Y \neq X}^{N} < T_{X_i} > + f_{XY} (< T_{Y,i+1} + T_{Y,i-1} >)$$

(The average of the output values of accepted fixed point i. e. neurons).

If F reaches the final value stop the processing otherwise decrease the F according to the

TABLE 1. Number of Failures in the Different Time Intervals for the Sequential Testing.

Time (hrs.)	Failed components	Time (hrs.)	Failed components	Time (hrs.)	Failed components	Time (hrs.)	Failed components	Time (hrs.)	Failed components
0-1	130	0-2	170	0-3	150	0-4	240	0-5	300
1-2	83	2-4	158	3-6	163	4-8	248	5-10	350
2-3	75	4-6	132	6-9	148	8-12	230	10-15	200
3-4	68	6-8	145	9-12	130	12-16	172	15-20	150
4-5	62	8-10	100	12-15	121	16-20	110		
5-6	56	10-12	73	15-18	146				
6-7	51	12-14	60	18-20	140				
7-8	46	14-16	16						
8-9	41	16-18	18						
9-10	37	18-20	20						
10-11	34								
11-12	31								
12-13	28								
13-14	64								
14-15	76								
15-16	62								
16-17	40								
17-18	12								
18-19	3								
19-20	1								

annealing schedule and repeats Step 2.

3. SIMULATION RESULTS

In this paper, for estimating the software reliability we have applied the technique of sequential testing with simulated annealing of mean field approximation. The pattern of failure can be obtained from life - test results, i.e. by testing a fairly large number of components until failure occurs, and observing the failure rate characteristics as a function of time. For optimization of software reliability we consider a series of tests conducted under certain stipulated conditions on 1,000 software components.

The total time duration of the tests is 20 hours. The number of components that will fail during the defined time interval is noted. The results obtained are tabulated as shown in Table 1.

This table lists the total number of failed components at the end of 1hr, 2 hr, 3hr, etc. in the second column. In the same manner, the total number of failed components at the end of 2hr, 4hr, etc. are listed. The same pattern can be seen in the following columns. Now after some iteration this number of failures will be stored in different energy minima's with the time interval (as Figure 1 shows) according to this energy function as given in Equation 2.2.

Initially the arbitrary slot is selected and the energy function is constructed as given in Equation 2.2. The annealing schedule is assigned with the maximum number of failures i.e. F. At the higher F, many states are likely to be seen. Therefore, the value of the constraints parameter F is gradually reduced as per the annealing schedule, the output value of the state perturbs. This perturbation continues until the network settles to a stable state or an equilibrium state.

At the stable state after minimizing the constraint parameter, Table 2 can be constructed. Table 2 shows the results that the number of

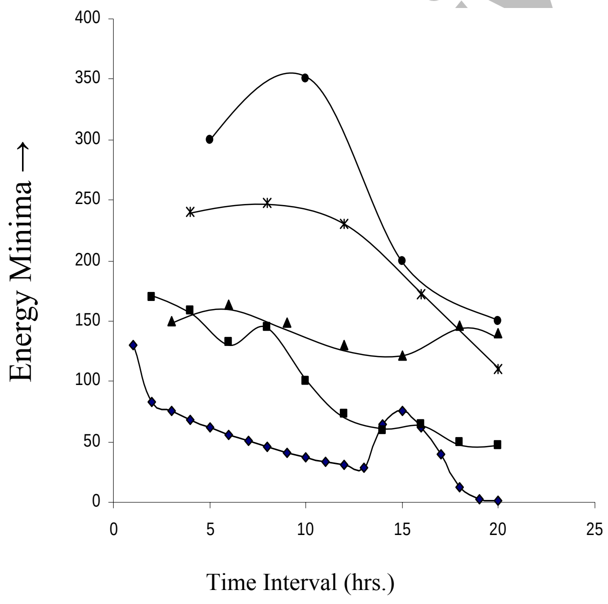

Figure 1. Number of failures at different time interval at different energy minima's.

TABLE 2. Selection of Time Interval with the Minimum Number of Failures.

Time interval	Number of Failures
0-1	130
0-2	
0-3	
0-4	
0-5	
1-2	83
2-3	75
2-4	
3-4	68
3-6	
4-5	62
4-6	
4-8	
5-6	56
5-10	
6-7	51
6-8	
6-9	
7-8	46
8-9	41
8-10	
8-12	
9-10	37
9-12	
10-11	34
10-12	
10-15	
11-12	31
12-13	28
12-14	
12-15	
12-16	
13-14	64
14-15	
14-16	65
16-17	40
16-18	
16-20	
17-18	12
18-19	3
18-20	
19-20	1

failures of 130 gradually reduces from its higher values of 350. This trend continues in the other rows for the selection of the next slots. Now the energy function will change as given in Equation 2.10 and the state of the neurons can be defined in Equation 2.5.

This entire process will continue for every schedule of the constraint parameter F, until F reaches the final state. After completing the entire iterations for the allowed minimum limit of F, Table 3 can be constructed.

Table 3 shows that, the number of failed components has been reduced. Initially we have 1,000 failures and after applying the annealing schedule in the sequential testing, 927 cumulative numbers of failures remain. The results of the fourth column shows that this technique of optimization reduced 73 failed components.

Figure 2 shows the optimized number of failed components with different time intervals in the energy landscape. This table also shows the failed component density, failed component intensity, and the reliability after getting the failures in each time interval. The table shows that the reliability of software components increases.

4. CONCLUSIONS

In the process of estimating the software reliability we are optimizing the failed software components to their minimum value. Using the simulated annealing technique of MFA we reduce the failures in a very systematic way. For this we divide the total time duration in the small time intervals or slots and note the failed components at the end of each slot.

The selection of time interval or slot depends upon certain constraints. The main contribution of our work is to design or formulations of necessary constraints under which the slot is selected and determining the optimized number of failed components of the software. To accomplish the task of optimizing the number of failed components, we select the slots in the sequential testing. The selection of slots satisfies all the necessary constraints imposed in the problem.

The Hopfield energy function represents the stored failed components in different time intervals in the sequential testing. The constraint parameter F of the annealing schedule will change on each iteration of the process with the fmax (number of failures), which is slightly larger than the number of failures between slots that have been selected for the sequence.

This newly found maximum number of failed components will be less then the previously found maximum number of failed components. So the simulated annealing process will continue with the new value of the constraint parameter. This process continues until the final value of the schedule is obtained.

At this state the units of the network represent the state of the equilibrium, which represents the minimum energy state for the network. Thus the minimum energy function will represent the possible minimum number of failed components in the sequential testing. Thus, energy iteration will be scheduled with the optimized value of F.

The reliability of the software is easily estimated with the optimized minimum number of failures determined by the network. The results show that after applying the sequential testing with MFA we can optimize the number of failures up to a minimum value. These minimum numbers of failures increase the software's reliability. More experiments and analytical investigation are still required for increasing the efficiency and speed of the solution.

8. REFERENCES

1. Hassoun, M. H., “Fundamentals of Artificial Neural Networks”, MIT Press, Cambridge, MA, (1995).
2. Wasserman, P. D., “Advanced Method in Neural Computing”, Van Nostrand Reinhold, New York, NY, (1993).
3. Fogel, D. B., “An Introduction to Simulated Evolutionary Optimization”, ***IEEE Trans. Neural Network***, 5, (1994), 3-14.
4. Fogel, D. B., “System Identification through Simulated Evolution: A Machine Learning Approach to Modeling”, Ginn press, Nedham, MA, (1991).

TABLE 3. Optimized Value of the Number of Failures, Cumulative Number of Failures, Failure Rate, Failure Density and the Reliability.

Time (hrs.)	Number of failed components f	Cumulative number of failures F	No. of survivors	Failure density	Failure rate	Reliability
0		0	1000			1
	130			0.13	0.139	
1		130	870			0.87
	83			0.083	0.101	
2		213	787			0.787
	75			0.075	0.1	
3		288	712			0.712
	68			0.068	0.1	
4		356	644			0.644
	62			0.062	0.101	
5		418	582			0.582
	56			0.056	0.101	
6		474	526			0.526
	51			0.051	0.101	
7		525	475			0.475
	46			0.046	0.101	
8		571	429			0.429
	41			0.041	0.1	
9		612	388			0.388
	37			0.031	0.1	
10		649	351			0.351
	34			0.034	0.101	
11		683	317			0.317
	31			0.031	0.103	
12		714	286			0.286
	28			0.028	0.103	
13		742	258			0.258
	64			0.064	0.283	
14		806	194			0.194
	65			0.065	0.04	
16		871	129			0.129
	40			0.04	0.368	
17		911	89			0.089
	12			0.012	0.144	
18		923	77			0.077
	3			0.003	0.039	
19		926	74			0.074
	1			0.001	0.013	
20		927	73			0.073

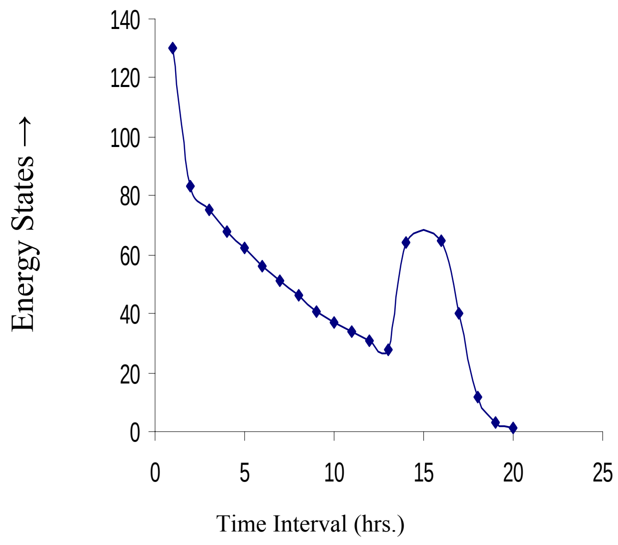

Figure 2. The energy states of the optimized number of failed components.

5. Musa, J. D., Iannino, A. and Okumoto, K., "Software Reliability: Measurement, Prediction, Application, Professional Edition", Software Engineering Series, McGraw - Hill, New York, NY, (1990).
6. Karunanithi, K., Whitley, D. and Malaiya, Y., "Using Neural Networks in Reliability Prediction", ***IEEE Software***, 9-4, (1992), 53-59.
7. Traintafyllos, G., Vassiliadis, S. and Kobrosly, W., "On the Prediction of Computer Implementation Faults Via static Error Prediction Models", ***Journal of Systems and Software***, 28-2, (1995), 129-142.
8. Sharma, N. K., Mangal, M. and Singh, M. P., "Solution of Traveling Salesman Problem with Simulated Annealing of Mean Field Approximation Neural Network", ***Proceeding of the Second International Conference on Mathematical and Computational Models***, Coimbatore Allied Publishers, (2003), 297-304.
9. Srinath, L. S., "Reliability Engineering", Affiliated East West Press Pvt. Ltd., (1991).
10. Waterman, R. E. and Hayatt L. E., "Testing - When do I Stop?", (Invited), ***International Testing and Evaluation Conference***, Washington, DC, USA, (1994).