Arid Zone Journal of Engineering, Technology and Environment. August, 2010; Vol.7, 24 -33

Print ISSN: 1596-2490, Electronic ISSN: 2545-5818
www.azojete.com.ng

K-NODE SET RELIABILITY OPTIMIZATION OF A DISTRIBUTED COMPUTING SYSTEM USING PARTICLE SWARM ALGORITHM

Bakare, G.A.,[1] Chiroma, I.N.[2] and Ibrahim, A.A.[3]

Abstract

A discrete Particle Swarm algorithm is proposed to solve a typical combinatorial optimization problem: *K*-Node Set Reliability (*KNR*) optimization of a distributed computing system (*DCS*) which is a well-known *NP*-hard problem is presented in this paper. The reliability of a subset of network nodes of a *DCS* is determined such that the reliability is maximized and specified capacity constraint is satisfied. The proposed algorithm is demonstrated on an 8 node 11 link *DCS* topology. The test results show that the proposed algorithm can achieve good solution quality and convergence characteristics.

1. Introduction

Advances in computer networks and low cost computing elements have led to increasing interest in Distributed Computing System (*DCS*). Among the numerous merits of using a *DCS* include more effective resource sharing, better fault tolerance, and higher reliability. In designing such systems, reliability must be considered, which largely relies on the topological layout of the communication links (Jan *et al*., 1993; Lin, 1994). Almost all of the optimization problems relevant to distributed computing system design are *NP*-complete. That is, for most problems, there is no known algorithm that could guarantee finding the global optimum in a polynomial amount of time. Several *DCS* reliability measures have been developed and one of these distributed system reliability measures, *K*-Node reliability (*KNR*), is adopted in Chen *et al*. (1994). Reliability optimization is the design of distributed computing systems, where *KNR* is viewed as the probability that all k (a subset of the processing elements) nodes in the *DCS* can be run successfully. They presented a heuristic algorithm for maximizing reliability by the node select problem to obtain an optimal design *DCS*. Their work did not consider an exhaustive method because it is too time consuming. Instead, they applied the exact algorithm, which examine k-node set reliability optimization with a capacity constraint to find an optimal solution of k-node reliability. The heuristic algorithm work well for large *DCS* problems as it largely avoids unnecessary generation of spanning trees. They regarded the *DCS* as a weighted graph, in which the weight of each node represents its capacity and generate k-node disjoint terms using a k-tree disjoint reduction method to obtain the *KNR*. This reduces the disjoint terms and hence reduces the computation time.

In Chen (1993), K-node set reliability optimization with capacity constraint for a *DCS* was examined using exact method. This method can obtain an optimal solution but cannot effectively reduce the problem space. Moreover, exact method spends more execution time

[1] Department of Electrical Engineering Programme, Abubakar Tafawa Balewa University, Bauchi. Nigeria.
e-mail: bakare_03@yahoo.com

[2] University Computer Centre, Abubakar Tafawa Balewa University, Bauchi. Nigeria.
e-mail: inchiroma@yahoo.com
[3] Department of Electrical & Electronics Engineering, University of Maiduguri, Maiduguri. Nigeria.
e-mail:ibrahim_ali@yahoo.com

with a large *DCS*. In fact, most distributed system problems are large and an increase in the number of nodes causes the exact method execution time to grow exponentially. Occasionally, therefore an application requiring an efficient algorithm with an approximate solution is highly attractive. In many cases, complex heuristics have to be developed to achieve satisfying results. Previous approaches have either been enumerative-based, which are applicable only for small *DCS* sizes (Aggarwal et al., 1982), or heuristic-based, which can only be applied to larger networks, but do not guarantee optimality. Therefore other heuristic approaches are required. There exist a number of researches, which deal with such approaches as discussed in Pierre *et al*. (1998). Genetic algorithm (*GA*) is one of such heuristics that have been found to be a very good computational tool for problem of this nature. They can be applied to search large, multimodal, complex problem spaces (Holland, 1975; Bakare, 2001). In Davis *et al*. (1987), a genetic algorithm to determine the link capacities of a network, initially generated by the software called DESIGNET was proposed. Chiroma (2005) applied *GA* to the optimization of *K*-node set reliability of a distributed computing system. Good results in terms of solution quality and convergence characteristics were obtained. Cheng (1998) employed *GA* approach in the backbone network design under the constraint: minimal total link cost and 1-FT (fault-tolerant to 1 link-failure).

Kennedy and Eberhart proposed a new evolutionary computation technique called particle swarm optimization (*PSO*) in 1995. This method was developed through the simulation of a simplified social system and has been found to be robust in solving continuous non-linear optimization problems (Gudise *et al*. 2003; Abido 2002; Kennedy *et al*., 1997; Zhang, 2004). Particle swarm algorithms differ from other evolutionary algorithms most importantly in both metaphorical explanations and how they work. The individuals (particles) persist over time, influencing one another's search of the problem space, unlike genetic algorithm where the weakest chromosomes are immediately discarded. The particles in *PSO* (similar to chromosomes in GA) are known to have fast convergence to local / global optimum position (s) over a small number of iterations.

In this paper, an innovative particle swarm algorithm, called discrete *PSO* for K-node set reliability optimization of a distributed computing system is presented. The feasibility of the proposed algorithm is demonstrated on a typical *DCS* topology obtained from Cheng (1998). Simulation shows that the proposed algorithm can achieve good results in terms of solution quality and convergence characteristics.

2. Problem formulation

The KNR problem is characterized as follows:
Given the topology of an undirected *DCS*, the reliability of each communication link, the capacity of each node and a possible set of data files, and assume that each node is perfectly reliable and each link is either in the working state or failed state. The problem can be mathematically stated thus:

Maximize $R\ (G_k)$ (*1*)
Subject to: $\sum c\ (v_i) \geq C_{\ constraint}$ (*2*)

Where G_k is the graph G with set k of nodes specified,
$k \geq 2$, $R(G_k)$ is the reliability of the k-node set, v_i is an *ith* node which represents the *ith* processing element, $c(vi)$ is the capacity of the *ith* node and $C_{constraint}$ is the total capacity constraint in the *DCS* .

3. Applied software techniques

3.1 Overview of Particle Swarm Optimization Technique

Particle swarm optimization (*PSO*) is an evolutionary computation technique that was originally developed in 1995 by Kennedy and Eberhart. It has been developed through simulation of simplified social models and has been found to be robust for solving non-linear, non-differentiability multiple optimal and multi-objective problems. The features of the technique are as follows:

- It is based on a simple concept and has high quality solution with stable convergence.
- The method is based on the researches about swarms such as fish schooling and bird flocking.
- It was originally developed for non-linear optimization with continuous variables; however it is easily expanded to treat problems with discrete variables. Therefore it is applicable to K-node reliability optimization of a distributing computing system.

PSO is an evolutionary technique that does not implement survival of the fittest. Unlike other evolutionary algorithms where an evolutionary operator is manipulated, each individual in the swarm flies in the search space with a velocity which is dynamically adjusted according to its own flying experience and its companions flying experience. The system initially has a population of random solutions. Each potential solution, called a particle, is given a random velocity and is flown through the problem space. The particles have memory and each particle keeps track of its previous best position, called the *pbest* and its corresponding fitness. There exist a number of *pbest* for the respective particles in the swarm and the particle with greatest fitness is called the global best (*gbest*) of the swarm. The basic concept of the *PSO* method lies in accelerating each particle towards its *pbest* and *gbest* locations, with random weight acceleration at each time step. The modified velocity of each particle can be computed using the current velocity and the distance from *pbest* and *gbest* as given by:

$$V_{id}^{k+1} = W \times V_{id}^{k} + c_1 \times rand_1 \times \left(pbest_{id} - x_{id}^{k}\right) + c_2 \times rand_2 \times \left(gbest_{id} - x_{id}^{k}\right) \quad (3)$$

In the discrete version of the *PSO*, the trajectories are changed in the probability that a coordinate will take on a binary value (0 or 1). Therefore, the positions are computed using:

$$\begin{aligned} &if \quad (rand < S(V_{id}^{k+1})) \quad \text{then } x_{id}^{k+1} = 1 \\ &\qquad\qquad\qquad\quad \text{else } x_{id}^{k+1} = 0 \end{aligned} \quad (4)$$

Where:
$S(V)$ is a sigmoid limiting transformation function given by:

$$S(V) = 1/\left(1 + e^{-V}\right) \quad (5)$$

rand , $rand_1$ and $rand_2$: random numbers between 0 and 1
V^{k}_{id} : current velocity of individual *i* at iteration *k*

$V^{min}_{id} \leq V^{k}_{id} \leq V^{max}_{id}$
V^{k+1}_{id} : modified velocity of individual *i*
x^{k}_{id} : current position of individual *i* at iteration *k*
$pbest_{id}$: *pbest* of individual *i*
$gbest_{id}$: *gbest* of the group
c_1 and c_2: the weighting of the stochastic acceleration that pulls each particle towards *pbest* and *gbest* .
W : inertia weight factor that controls the exploitation and exploration of the search space by dynamically adjusting the velocity. It is computed using:

$$W = W^{max} - \frac{W^{max} - W^{min}}{gen^{max}} \times iter \qquad (6)$$

gen^{max} : maximum generation
iter : current iteration number.

3.2 Overview of Genetic Algorithms Based Technique

Genetic Algorithms (*GAs*), first proposed by John Holland in the 1960's, are numerical optimization algorithms based on principles inspired from the genetic and evolution mechanisms observed in natural systems and populations of living beings (Bakare, 2001). The robust behaviour, which is the distinguishing feature of *GAs* with respect to other optimization methods, implies that *GAs* must differ in the following fundamental ways:

- It searches from a population of candidates and do not process only one single solution; thus, they are resistant to being trapped in local optima.
- *GAs* use probabilistic transition rules and not deterministic rules.
- *GAs* exploit only the payoff information of the objective function to guide their search towards the global optimum. They do not depend on any additional information like the existence of derivatives.

One of the disadvantages of *GA* search is the large number of function evaluations, with the resulting undesirably long execution time. Binary encoding *GAs* deals with binary strings, where the number of bits of each string simulates the genes of an individual chromosome, and the number of individuals constitutes a population. Each parameter set is encoded into a series of a fixed length of string symbols, usually from the binary bits, which are then concatenated into a complete string called chromosome. Sub-strings of specified length are extracted successively from the concatenated string and are then decoded and mapped into the value in the corresponding search space. Generally, *GAs* implementation comprises three different phases: initial population generation, fitness evaluation and genetic operations (Bakare, 2001).

4. Implementation of the proposed technique

Particle swarm K-node reliability optimization is developed as follows:

4.1 Representation of Individual String

Before using the binary *PSO* algorithm to solve the *KNR* combinatorial optimization problem, the representation of a particle must be defined. A particle is also called an individual. The problem involves a capacity constraint. The network topology is fixed. The size of every node is also fixed. Binary coding scheme is therefore employed. The length of a chromosome is equal to the number of network nodes of the form:

$$x_n, x_{n-1},.. x_{i-1},\ldots x_2, x_1 \qquad (7)$$

Each bit indicates whether the node is selected for the k-node set,
where $x_i = 1$ if v_i is selected; otherwise $x_i = 0$.

4.2 Initialization

The *DCS* parameter such as the number of nodes (n), the number of links (e), each link's reliability (a_{ij}) and unreliability (b_{ij}), each node's capacity, $c(vi)$, capacity constraint ($C_{constraint}$), etc are entered into the FORTRAN program. The *PSO* parameters such as particle size (ps), minimum and maximum inertia weights, W^{min} and W^{max}, the limit of velocity change V^{min} and V^{max}, maximum generation (gen^{max}), acceleration constants c_1 and c_2, etc are also entered.

For each node v_i, if the degree of v_i is $d(v_i)$ and if the links $e_{i,k1}$, $e_{i,k2}$, $e_{i,k3}$, …, $e_{i,kd(vi)}$ are adjacent to v_i, then its weight can be computed as in Yeh (2001) using:

$$w(v_i) = a_{i,k_1} + b_{i,k_1} * (a_{i,k_2} + (b_{i,k_2} * (a_{i,k_3} + (..(b_{i,k_{d(v_i)}}))..) \qquad (8)$$

Where $i, k1, k2, \ldots, kd(vi) \in \{1, n\}$. The heaviest node is then determined.

Initial population of chromosome Xi =[xn, .., x1] $\in$ X are randomly generated with the heaviest node included. The node capacity that includes this chromosome is computed. If the sum of the capacity of this chromosome satisfies the capacity constraint, then it is appended to the population, otherwise it is discarded.

4.3 Evaluation function

The objective function of chromosome is then computed using Yeh *et al*. (2001):

$$f = \left(ratioprdl_k + ratio\deg_k\right) / \sqrt{|k|+2} \qquad (9)$$

Where

$$ratioprdl_k = \left(\sum_{v_i \varepsilon G_k} \sum_{v_j \varepsilon G_k} a_{i,j} \right) \Big/ k(k-1)$$

$$ratio\deg_k = \left(\sum_{v_i \varepsilon G_k} d(v_i) \right) \Big/ (n-1)k$$

$$2 \le |k| \le n$$

The fitness values that determine which chromosomes are to be carried onto the next generation are computed from the objective function equation.

4.3 Implementation of the PSO for KNR

The main steps involved in the searching procedures of the proposed *PSO* method are given by the following:

Step 1: Read the *DCS* data and the *PSO* parameters. Compute the weight of each node using Equation (8) and select the heaviest node.

Step 2: Generate randomly initial population of particles with random positions and velocities and the heaviest node included. The node capacity that includes this chromosome is computed. If the sum of the capacity of this chromosome satisfies the capacity constraint, then it is appended to the population, otherwise it is discarded.

Step 3: Compute the fitness value of the initial particles in the swarm using the objective function (Equation 9). Set the initial *pbest* to current position of each particle and the initial best evaluated values among the swarm is set to *gbest* .

Step 4: Update the generation count.

Step 5: Update the velocities and the positions according to Equations (3) and (4) respectively.

Step 6: Compute the fitness value of the new particles in the swarm using Equation (9). Update the *pbest* with new positions if the particle's present fitness is better than the previous one. Also update the *gbest* with the best in the population swarm.

Step 7: Repeat steps 4 to 6 until the preset convergence criterion: maximum number of generations is fulfilled.

Step 8: Compute the reliability $R\ (G_K)$ of the optimal K-node set result in the chromosome of the population using Monte Carlo technique (Lin *et al.* 1994) and output the K-node set corresponding to the $R\ (G_K)$.

5. Results and discussion

The procedure described above was implemented using the FORTRAN language and the developed software program was executed on a 450 MHz Pentium III PC. To illustrate the effectiveness of the proposed method, a distributed computing system topology obtained from Chen *et al.* (1993) was considered.

The topology of the distributed computing system with eight (8) nodes, and eleven (11) links, was considered as a case study where $c\ (v_i)$ represents the capacity of nodes v_i, and a_{ij} represents the reliability of the link e_{ij}.

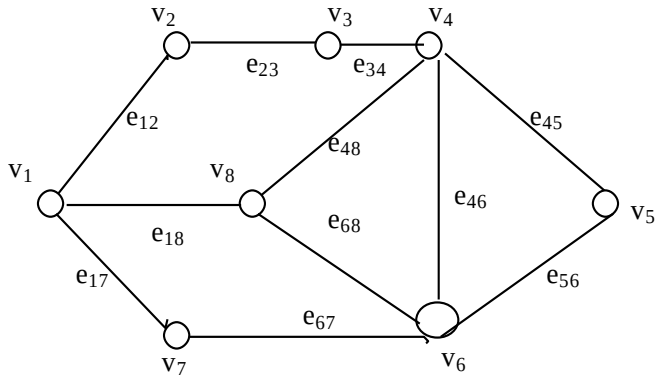

Figure 1: A *DCS* with 8 nodes and 11 links

Benchmark input data for *DCS* are:
c $(v_1) = 39$, $c(v_2) = 45$, $c(v_3) = 38$, $c(v_4) = 53$, $c(v_5) = 47$, $c\ (v_6) = 49$, $c(v_7) = 51$, $c(v_8) = 41$.
$a_{12}= 0.89$, $a_{17}= 0.81$, $a_{18}= 0.93$, $a_{23}= 0.85$, $a_{34}= 0.91$, $a_{45}= 0.82$, $a_{46}= 0.83$, $a_{48}= 0.96$, $a_{56}= 0.87$, $a_{67}= 0.84$, $a_{68}= 0.88$. *Cconstraint* ≥ 100.

The proposed approach was then applied to this problem and the optimum parameter setting for the *PSO* is shown in Table I.

Table I: Optimum parameter setting for *PSO*

C_1	2
C_2	2
V^{max}	2
V^{min}	-2
gen^{max}	50
Particle size (PS)	20
W^{max}	0.9
W^{min}	0.4

The maximum fitness value of 0.701 was achieved and the chromosomes are 0001010 (from left to right). The k-node set in the population chromosome is $\{v_4, v_6\}$, i.e. nodes 4 and 6. The algorithm then computes the reliability and output the k-node set, which is the highest reliability k-node set, and R ($\{v_4, v_6\}$) = 0.9974. It can therefore be seen that the same solution quality was achieved by the *PSO* when compared with *GA* approach (Chiroma, 2005) and the exact method (Chen *et al*., 1993). The convergence characteristic of *PSO* is as shown in Figure 2. The convergence characteristics of both *GA* and *PSO* methods are comparatively shown in Figure 3. As can be seen from this figure, both the *PSO* and *GA* have rapid convergence characteristics. *GA* achieved the maximum fitness earlier than the *PSO*: second and third generations respectively.

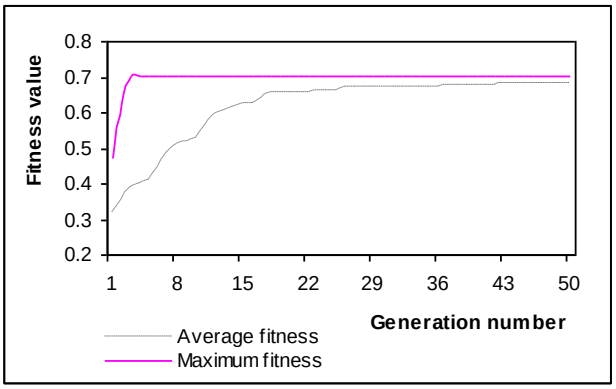

Figure 2: Convergence characteristics of *PSO*

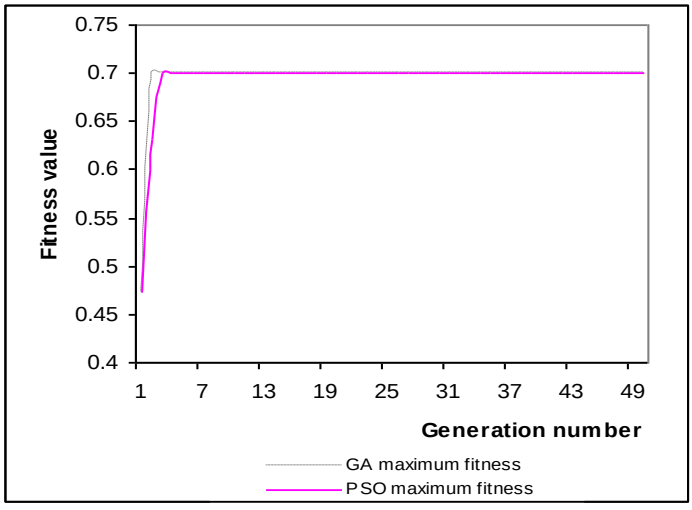

Figure 3: Comparison of convergence characteristics of *PSO* & *GA*

This shows that *PSO* has good convergence characteristics and is less prone to being trapped into local optimal. Finally, *PSO* has less parameter settings when compared to GA.

Comparing the complexity of the proposed method with that of exact method, the complexity of the exact method is O (2*e* x 2*n*), where *e* denotes the number of edges (links) and *n* represents the number of nodes (Chen *et al*., 1994). With our proposed algorithm, however, in the worst case, the complexity of evaluating the weight of each node is O(*e*), that of selecting the heaviest node is O(*n*), and that of computing the reliability of the k- node set is O(*m2*), where *m* represents the number of paths of the selected *k*-node set (Aziz, 1982). Therefore, the complexity of the proposed algorithm is O (*n*ps*ng+m2)*, where *ps* is the population size, and *ng* is the total number of generation. Although the conventional techniques can yield an optimal solution, they cannot effectively reduce the number of reliability computations. Some applications, especially *NP*-complete problems, often require an efficient algorithm for computing the reliability. Under this circumstance deriving the optimal reliability may not be feasible. So an efficient algorithm, even when it yields only approximate reliability, is preferred.

4. Conclusions

In this paper, comparison is made between the particle swarm optimization and genetic algorithm based *K*-node set reliability optimization. The efficiency and accuracy of the proposed algorithm has been verified by implementing the procedure using FORTRAN language program on a Pentium III microcomputer. Verification on a typical *DCS* topology revealed that *PSO* achieved the same good solution quality as compared with *GA* and exact method. *PSO* exhibits good convergence characteristics and is less prone to being caught at the local optimal. The proposed algorithm can efficiently obtain the optimal k-node set reliability with a capacity constraint.

References

Abido M. A. (2002). Optimal design of power system stabilizers using *PSO*. *IEEE Transactions on Energy and Conversion*. Vol 17, No. **3**, pp. 406-413.

Aggarwal, K.K., Chopra, Y.C. and Bajwa J.S. (1982). Topological layout of links for optimizing the overall reliability in a computer communication system. *Microelectronics and Reliability* **22**, pp.347-351

Aziz M. A., (1982). Path set enumeration of directed graphs by the set theoretic method. *Microelectronics and Reliability*, Vol. 5, No. **37**, pp. 809-814.

Bakare, G. A. (2001). *Removal of Overloads and Voltage Problem in Electric Power Systems using Genetic Algorithm/Expert Systems Approaches*. Ph.D. Dissertation published by Shaker Verlag, Germany.

Chen R. S., Chen, D. J. and Yel, Y. S. (1994). Reliability optimization in the design of distributed computing systems. Proceedings of Int. Conference on Computing and Information Institute of Computer Science and Engineering of National Chiao Tung University, Hsinchu, Taiwan. R.O.C., pp.422-439.

Chen R. S., Chen, D. J. and Yel, Y. S. (1993). Reliability optimization of distributed computing systems subject to capacity constraint. *International journal of Computers & Mathematics with Applications.* 29(**3**), pp. 93-99.

Cheng S. T. (1998) Topological optimization of a reliable communications network. *IEEE Trans. on Reliability*. Vol.47, No.**3**, pp. 225-233.

Chiroma I. N. (2005) *A genetic algorithm- based efficient optimization of K-node set reliability for distributed computing system.* MSc. Thesis (Computer Science), Abubakar Tafawa Balewa University, Bauchi, Nigeria.

Davis, L. and Coombs, S. (1987). Optimizing network link sizes with genetic algorithms in modelling and simulation methodology. Knowledge System's Paradegins. Amsterdam, The Netherlands, Elsevier, pp.22-28.

Gudise V. G. and Venayamooorthy G. K. (2003) Comparison of *PSO* and back propagation of training algorithms for neural networks", IEEE Swarm Intelligence Symposium.

Holland, J. H (1975). Adaptation in natural and artificial systems. University of Michigan Press.

Jan, R.H., Hwang, F.J. and Cheng, S.T. (1993). Topological optimization of a communication network subject to a reliability constraint. *IEEE Trans on Reliability*. Vol. 42, N0.**1**, pp.63-70.

Kennedy J. and Eberhart R. (1997). A discrete binary version of the particle swarm optimization". Proceedings of IEEE International Conference on Neural Networks, Perth, Australia,Vol. 4, pp. 4104-4108.

Lin, M. S. (1994). *The reliability analysis on distributed computing systems*. PhD Dissertation, Institute of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu, Taiwan, R.O.C., pp.64-75.

Pierre, S. and Legault, G. (1988). A genetic algorithm for designing distributed computer network topologies. *IEEE Transactions on Systems, Man and Cybernetics*. Part B: Cybernetics, Vol.28, No.**2**.

Yeh Y. S., Chiu C. C. and Chen R. S. (2001). A genetic algorithm for *K*-node set optimization with capacity constraint of a distributed system. Proceedings National Science Council ROC (A), Vol. 25, No. 1, pp. 27-34.

Yeh M. S., Lin J. S. and Yeh W. C. (1994). A new Monte Carlo method for estimating network reliability. Proceedings of the 16th International Conference on Computers and Industrial Engineering, pp. 723-726.

Zhang W. and Liu Y. (2004). Reactive power optimization based on *PSO* in a practical power system". IEEE, PES Meeting.