2010 Second International conference on Computing, Communication and Networking Technologies

Optimization Of Disk Replacement As A Large Domain Objective Function

Sandeep Patil, Abhinay Nagpal,Suyash Jape, Bhushan Jain
rsandeep@in.ibm.com,abhinaynagpal@in.ibm.com, suyash.jape@in.ibm.com, bhujain1@in.ibm.com

Abstract- *There are several motivations for trying to find hardware infrastructure which can be replaced. Most administrators believe in owning a diversified portfolio of disks, tapes to replace older ones, a strategy that has been widely accepted and practiced. When attempting to create an efficient portfolio of disks to be replaced there are numerous factors to consider set of fitness heuristics over a population of disks, the goal here is to find a portfolio that has a high probability of failure and at the same time will ensure savings in power consumption and carbon emission. Given the fundamental financial and price information of power consumed ,cooling required, we attempt to use GA (Genetic Algorithm) and a hybrid PSO-EO (Particle Swarm Optimization) algorithm to identify disks that are likely to outperform the existing ones by preventing failure and having excess returns and compare these two approaches.*

I. INTRODUCTION

The pressure is on businesses all around the world to be more environmentally conscious and reduce their carbon footprints. Rising energy prices and drastic increases in the density and power requirements of server and storage solutions over the past few years have driven IT managers to consider strategies for reducing power consumption in the data center. According to the Department of Energy, the average cost of power in the United States has increased 25% since 2000 to 8.74 cents per kilowatt. From a technology perspective, increased server, storage and connectivity density provides significant 'power budget' challenges to data center managers. HVAC systems draw even more electricity as they work harder to remove increased amounts of excess heat. If the power and cooling systems in the data center are near capacity, expensive upgrades may become unavoidable. With the advent & ongoing success of data cloud computing, the disk becomes even more critical and the data centers and associated storages need to take a closer look at the methods and tools they use for their disk replacement Strategy. Generally, there exist two types of preventive maintenance schemes, i.e. condition based and time based preventive maintenance [3]. For condition based preventive maintenance, the action taken after each inspection is dependent on the state of the system. It could be no action, or minimal maintenance to recover the system to the previous stage of degradation, or major maintenance to bring the system to as good as new state. For time based preventive maintenance, the preventive maintenance is carried out at pre-determined time intervals to bring the system to as good as new state [7].

978-1-4244-6589-7/10/$26.00 ©2010 IEEE

In literature a number of different methods have been applied in order to predict disk replacement. Sim and Endrenyi introduced the multi-stage exponential device failure model in [5], in which the idea of minimal preventive maintenance was introduced. The minimal preventive maintenance is de ned as the preventive maintenance activity with limited effort and effect [4]. For deterioration failures modeled as several stages of exponential distributions, minimal maintenance restores the system to the previous deterioration stage. Corresponding to the minimal preventive maintenance, the idea of major maintenance is de ned as the maintenance operation by which the device is restored 'as good as new' status.

With condition based preventive maintenance, the maintenance action taken after each inspection is dependent on the state of the system. There could be no action, or minimal maintenance to recover the system to the previous failure stage, or major maintenance to bring the system to as good as new state. Hosseini et al. [2] introduced the threshold-type policy for the maintenance action. No maintenance action is taken if the device deterioration stage is found to be smaller than the minimal maintenance threshold; minimal maintenance is performed if the device deterioration stage is found to be between the minimal maintenance threshold and the major maintenance threshold; and major maintenance is carried out if the device deterioration stage is larger than the major maintenance threshold. This model was captured by a stochastic Petri net, and its optimal inspection interval to maximize the system availability is presented. Closed-form results for such threshold-type condition-based maintenance problems are reported in [1].

Technical analysts, known as chartists, attempt to predict the failure by tracing patterns that come from the study of charts which describe historic data of the such

failure. Fundamental analysts study the intrinsic Mean Time to Failure (MTTF) of a disk. In Traditional Time Series forecasting an attempt to create linear prediction models to trace patterns in historic data takes place. These linear models are divided in two categories: the univariate and the multivariate regression models, depending on whether they use one of more variables to approximate the disk failure time series.

However, these approaches have some drawbacks in solving the disk replacement problem. For example, fuzzy approach [5-6] usually lacks learning ability, while neural network approach [7] has over fitting problem and is often easy to trap into local minima. In order to overcome these shortcomings, Genetic Algorithm (GA) is used to perform this task. Some related typical literature can be referred to [5-7] for more details. The main aim of this study is to select some valuable disks using GA and to test the efficiency of the GA for disk replacement. The rest of the study is organized as follows. Section 2 describes the selection process based on the genetic algorithm in detail. Section 3 presents a simulation experiment. . Section 4 compares GA and a hybrid PSO-EO algorithm for solving the disk selection problem and Section 5 concludes.

II. GA-BASED DISK SELECTION AND REPLACEMENT PROCESS

GA imitates the natural selection process in biological evolution with selection, crossover and mutation [11-15] and the sequence of the different operations of a genetic algorithm is shown Fig. 1. That is, GA is procedures modeled after genetics and evolution. Genetics provide the chromosomal representation to encode the solution space of the problem while evolutionary procedures are designed to efficiently search for attractive solutions to large and complex problem. Usually, GA is based on the survival-of-the-fittest fashion [8-9] by gradually manipulating the potential problem solutions to obtain the more superior solutions in population. GA is intrinsically parallel and inclined to determine the global optimum. Since GA can generate many offspring in a complete loop, it can explore the search space in a multi-direction way. GA has been proven to be effective at escaping local optima and discovering the global optimum through genetic operations in some problems. With crossover, there is a transfer of information between successful solutions, which means offspring can benefit from what parents have learned, and parental schemata can be mixed and combined so as to reproduce next generations with strengths of both their parents. Therefore, GA has a higher probability of finding the global optimal solution in a relatively short time compared with other classical heuristic search algorithms [11, 13-15]. Genetic Algorithms are useful when:

• The search space is large, multidimensional, and complex or has a high level of uncertainty.

• Domain knowledge is limited or narrowing the search space is difficult

• No mathematical analysis is available.

•Traditional search methods fail or have a poor performance.

Our aim is to identify the quality of each disk and using GA so that administrators can choose some good ones for replacement. Here we use disk ranking to determine the quality of disk. The disks with a high rank are regarded as bad quality disk. In this study, some indicators of the disk are employed to determine and identify the quality of each disk. That is, the power consumption and failure susceptible rate indicators of the disks are used as input variables while a score is given to rate the disks. The output variable is disk ranking. Throughout the study, five important indices

1) Average Power consumed by each disk.

2) Average Heat Dissipated by each disk.

3) Average Cooling Power required by each disk

4) Average Carbon Credits consumed by each disk.

5) Average life remaining as per hours used and compared to disks MTRF as per manufacturer are utilized in this study.

i) INPUT VARIABLES

The first step is to input the Mean Time to failure of a disk as supplied by disk vendor.

The next step is to calculate the environmental costs for the old and new disk configurations.

Floor space and electricity prices can vary widely across geographic regions of the country. For this example, we estimated the floor space cost to be \$20/ft^2 per month and the electricity cost to be \$0.10/kwh.

Also, since the disk configuration involved multiple expansion units, requiring eight racks of disk space, for the purpose of these calculations we stacked the disks on sixteen 41" x 25.5" racks

Computer room cost per square foot – \$20/ft^2

Cost of electricity – \$0.10 kwh

Electricity is required to both power and cool the hardware. To calculate the yearly electrical cost (yec) of running the two systems, add the power (p kwh) and cooling (c kwh) requirements for each device and the number of hours in a year are 8760 and the cost per kwh is \$0.10. nu represents the number of units.

$$\boxed{yec = (p + c) * nu * 8760 * \$0.10} \quad (1)$$

The total environmental cost (tec) would be a sum of the yearly footprint cost and the yearly electrical costs where the yearly footprint cost is 12 times the footprint cost (f).

$$\boxed{tec = f * 12 + yec} \quad (2)$$

Model of the average hard disk power consumption for typical operations of a user can be described by the following formula:

$$P_{typ} = (Idle * 90\% + Write * 2.5\% + \mathrm{Re}ad * 7.5\%)/100\% \quad (3)$$

Digits (multipliers for these power values) denote percentage of the HDD mode duration (we take maximum power consumption values for reading and writing, which correspond to the beginning zones of a disk; Seekmode is actually metered here through reading and writing). This model is based on the assumption that read/write HDD operations make up 10% of the total time for the typical desktop usage.

The average power consumption during intensive hard disk operations (for example, defragmenting disks, scanning the surface, copying files, checking files for viruses in the background, etc) can be defined by the following formula:

$$P_{\max} = (Write + Seek + \mathrm{Re}ad * 3)/5 \quad (4)$$

When the input variables are determined, we can use GA to distinguish and identify the quality of each disk, as illustrated in Fig. 1. The detailed procedure is illustrated as follows.

ii) CHROMOSOME REPRESENTATION AND EVOLUTION

First of all, a population, which consists of a given number of chromosomes, is initially created by randomly assigning "1" and "0" to all genes. In the case of disk ranking, a gene contains only a single bit string for the status of input variable. In this study, the initial population of the GA is generated by encoding five input variables. For the testing case we design 11 statuses representing different qualities in terms of different interval, varying from 0 (Extremely poor) to 10 (very good). Other input variables are encoded similarly. That is, the binary string of a gene consists of four single bits.

The subsequent work is to evaluate the chromosomes generated by previous operation by a fitness function, while the design of the fitness function is a crucial point in using GA, which determines what a GA should optimize [11]. Since the output is some estimated disk ranking of designated testing companies, some actual disk ranking should be defined in advance for designing fitness function. Here we use annual price return (APR) to rank the disk and the APR is represented as:

$$ARP_n = \frac{ASP_n - ASP_{n-1}}{ASP_{n-1}} \quad (5)$$

Where APRn is the annual price return for year n, ASPn is the annual disk price for year n. usually; the disks with a high annual price return are regarded as good disks for replacement. With the value of APR evaluated for each of the N disks, they will be assigned for a ranking r ranged from 1 and N, where 1 is the highest value of the APR while N is the lowest.

Thus, the fitness function can be designed to maximize the APR and minimize the root mean square error (RMSE) of the difference between the indicator derived ranking and the next year's actual ranking of all the disks for a particular chromosome, representing by:

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(R_{derived} - R_{actual}\right)^2} \quad (6)$$

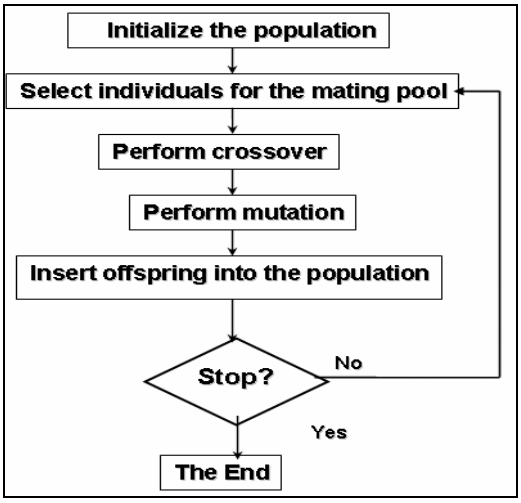

Figure 1. Overview genetic algorithm

```
BEGIN
 Generate initial population;
 Compute fitness of each individual;
 REPEAT /* New generation /*
  FOR population_size / 2 DO
   Select two parents from old generation;
    /* biased to the fitter ones */
   Recombine parents for two offspring;
   Compute fitness of offspring;
   gene <- DECODE(chromosome)
   data <- INPUT Parameters
   problem <- PREPAREPROBLEMS(data)
   //generate all solutions
   solutions <- APR (phenotype)  //Choose
   the best solution with //maximum
   returns
   best-fitness-value <- FITNESS-FN(solutions)
   Insert offspring in new generation
  END FOR
 UNTIL population has converged
END
```

Figure 2. Disk selection algorithm

After evolving the fitness of the population, the best chromosomes with the highest fitness value are selected by means of the roulette wheel [12-14].
Thereby, the chromosomes are allocated space on a roulette wheel proportional to their fitness and thus the fittest chromosomes are more likely selected. In the following crossover step, offspring chromosomes are created by some crossover techniques. A one-point crossover technique is employed, which randomly selects a crossover point within the chromosome. Then two parent chromosomes are interchanged at this point to produce two new offspring. The mutation prevents the GA from converging too quickly in a small area of the search space [11]. Finally, the final generation will be judged. If yes, then the optimized results are obtained. If no, then the evaluation and reproduction steps are repeated until a certain number of generations, until a defined fitness or until a convergence criterion of the population are reached. In the ideal case, all chromosomes of the last generation have the same genes representing the optimal solution [12]. Through the process of GA optimization, the disks are ranked, administrators can select the top n disks to construct a portfolio for replacement

The basic procedure of GA for this problem is similar as described earlier, but a suitable chromosome representation is needed to encode its solution space and an appropriate fitness function should be designed. In order to apply the model, the values of the expected return E(Ri) and covariance σ_{ij} for all i and j should be determined.

Solution for asset allocation for disk should be a composition of the disk quantity to be held so as to minimize the risk on a given level of expected return which will get the optimal solution. Thus the chromosome can be designed as follows: each of the disk weight (w) is a composite of eight bits, representing the value from 0 to 255, thus the normalized weight (x) of each disk can be calculated with following equation

$$x_i = \frac{w_i}{\sum_{i=1}^{n} w_i} \qquad (7)$$

The fitness function is another important issue in genetic algorithms for solving the problem. In the disk culling optimization, the fitness function must make a rational tradeoff between minimizing risk and maximizing return.

III. EXPERIMENT ANALYSIS

The sample data used in this study is from a span of over 3 years. For simulation, 350 disks were randomly selected. First of all, the disk information as the input variables is fed into the GA to obtain the derived disk ranking. This output is compared with the actual disk ranking in terms of APR. In the process of GA optimization, the RMSE between the derived and the actual ranking of each disk is calculated and served as the evaluation function of the GA process. The best chromosome obtained is used to rank the disks and the top n disks are chosen for the replacement [13]. For experiment purpose, the top 10 and 20 disks are chosen for testing according to the ranking of disk quality using GA. The top disks selected by GA can construct an optimal set of disks to be replaced. For convenience, equally weighted portfolios are built for comparison purpose. It is observed that as the number of generations increases the algorithm converges to a better solution. The Hamming wall problem has been overcome with the use of gray-codes.

In order to evaluate the usefulness of the GA optimization, we compared the net accumulated return generated by the selected disk from GA with a benchmark. The benchmark return is determined by an equally weighted portfolio of all the disks available in the experiment. Fig. 3 reveals the results for different sets.

From Fig. 3, we can find that the net accumulated return of the equally weighted portfolio formed by the disks selected by GA is significantly outperformed the benchmark. The larger the number of disks in the portfolio is, the more flexible for the set to make the best composition to avoid risk. However, selecting bad disks is the prerequisite of obtaining a good set of disks to be replaced

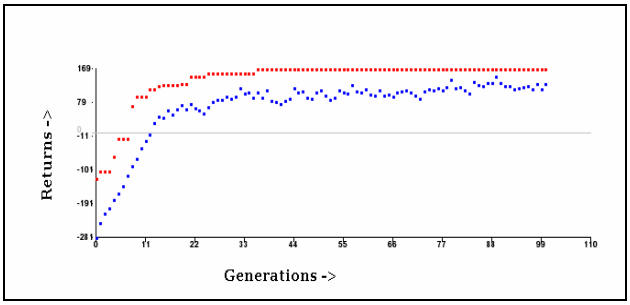

Figure 3. Potential Savings outperform conventional SMART tool based savings

Optimal Size:10

Alleles:[IntegerGene(0,6)=0, IntegerGene (0,6)=1, IntegerGene(0,6)=2, IntegerGene(0,6)=3, IntegerGene(0,6)=4, IntegerGene (0,6)=5, IntegerGene(0,6)=6], Application data:null

```
Fitness value [307.8713731063958]
```

Sample chromosome Size:10,

Alleles:[IntegerGene(0,6)=0, IntegerGene(0,6)=2, IntegerGene(0,6)=1, IntegerGene(0,6)=4, IntegerGene(0,6)=5,IntegerGene(0,6)=3, IntegerGene(0,6)=6], Application data:null

```
Fitness value [1479.769464870246]
```

Figure 4 Observed Output

IV. ALTERNATIVE APPROACH –HYBRID PSO ALGORITHM

PSO has strong global-search ability, while EO has strong local-search ability. In this work, we propose a novel hybrid PSO-EO algorithm which combines the merits of both PSO and EO. This hybrid approach makes full use of the exploration ability of PSO and the exploitation ability of EO and offsets the weaknesses of each other. Consequently, through introducing EO to PSO, the proposed approach is capable of escaping from a local optimum. However, if EO is introduced to PSO each iteration, the computational cost will increase sharply and at the same time the fast convergence ability of PSO may be weakened. In order to perfectly integrate PSO with EO, EO is introduced to PSO every "I_n" iterations. Therefore, the hybrid PSO-EO approach is able to keep fast convergence in most of the time under the help of PSO, and to escape from a local optimum with the aid of EO. The value of parameter "I_n" is predefined by the user.

4.1 Hybrid PSO-EO Algorithm

For maximization problem the hybrid PSO-EO algorithm works as Figure 5. It is important to note that, in order to find out the worst component (or so-called decision variable), each component of a solution should be assigned a fitness value in the EO procedure

4.2 Mutation Operator

Since there is merely mutation operator in EO, the mutation plays a key role in EO search. In this work, we present a mutation method based on mixing Gaussian mutation and Cauchy mutation and choose it as the mutation operator in EO. The mechanisms of Gaussian and Cauchy mutation operations have been studied by Yao et al. [15]. They pointed out that Cauchy mutation is better at coarse-grained search while Gaussian mutation is better at fine-grained search. Inspired by the idea proposed by Yao et al. [15], we present a new mutation method based on mixing Gaussian mutation and Cauchy mutation. It must be indicated that, unlike the method in [15], this mutation doesn't compare the anticipated outcomes between Gaussian mutation and Cauchy mutation due to the characteristics of EO. In the hybrid GC mutation, the Cauchy mutation is first adopted. It means that the large step size will be taken first at each mutation. If the new generated variable after mutation goes beyond the range of variable, the Cauchy mutation will be used repeatedly for some times (T C), until the new generated offspring falls into the range. Otherwise, Gaussian mutation will be carried out repeatedly for another some times (T G), until the offspring satisfies the requirement. That is, the step size will become smaller than before. If the new generated variable after mutation still goes beyond the range of variable, then we choose the upper or lower bound of the decision variable as the new generated variable.

1. Initialize a swarm of particles with random positions and velocities on dimensions. N D Set iteration=0.
2. Evaluate the objective function value for each particle, and update P_i (i=1… N) and Pg
3. Update the velocity and position of each particle using Eq.1 and Eq.2, respectively.
4. Evaluate the objective function value for each particle, and update P_i (i=1… N) and Pg
5. If the (iteration mod I_n) =0, then EO procedure is introduced. Otherwise, continue the next step.
6. If the terminal condition is satisfied, go to the next step; otherwise, set iteration= iteration + 1, and go to Step 3.
7. Output the optimal solution and the optimal objective function value.

Figure 5 PSO-EO Algorithm

Thus, our approach combines the advantages of coarse-grained search and fine-grained search. The above analyses show that the hybrid GC mutation is very simple yet effective. Unlike some switching algorithms which have to decide when to switch between different mutations during search, the hybrid GC mutation does not need to make such decisions.

In this work, the Gaussian mutation performs with the following representation:

$$x_k' = x_k + N_k(0,1) \tag{8}$$

where xk and xk' denote the k-th decision variables before mutation and after mutation, respectively, Nk(0; 1) denotes the Gaussian random number with mean zero and standard deviation one and is generated anew for k-th decision variable. The Cauchy mutation performs as follows:

$$x_k' = x_k + \delta_k \tag{9}$$

where δ_k denotes the Cauchy random variable with the scale parameter equal to one and is generated anew for the k-th decision variable.

In the hybrid GC mutation, the values of parameters T C and T G are set by the user beforehand. The value of T C decides the coarse-grained searching time, while the value of T G has an effect on the fine-grained searching time. Therefore, both values of the two parameters can not be

large because it will prolong the search process and hence increase the computational overhead.

V. RESULTS

The modeled system's functioning on the two alternative approaches has shown the following comparisons:

Algo	Runtime	Success (%)	Worst	Mean	Best
PSO -EO	0.177	77	7	8.5	10
GA	2.086	65	5	7.5	10

Table 1: Comparison in Disk Portfolio Selection

VI. CONCLUSIONS

This paper presents a method to solve the Disk Portfolio Selection problem. Since we get all information in all the databases the input parameters are obtained by normal queries. We present two solutions to combine evolutionary algorithms with planning techniques to get optimal plans. We find the PSO-EO performs better than Genetic Algorithm. The portfolios constructed are evaluated by the fitness function to select better combinations. The results show it is a workable solution. In the future, we will work on efficiency improvement. A more complete calculation would also include:

• The cost of maintenance after the warranty period expires. Warranty periods differ among vendors and across products.

• The cost to replace disk and tape after several years (if that was the typical buying pattern of the IT organization). Some IT organizations replace disk every two to three years and tape drives every five to seven years.

• The cost of phasing in equipment purchases. It is not necessary to purchase all equipment in the first year. Not all of the disk controllers or tape drives and cartridges would need to be purchased the first year.

• The cost associated with backing up data to local or remote tape or disk. This exercise assumes that only one copy of backup data exists and that additional backup copies are not stored at remote locations for disaster-recovery purposes.

• The additional cost of mirroring disk (RAID-1). Mirroring disk doubles the cost of the disk solution. Though other forms of RAID protection, such as RAID-5, can provide protection against drive failures and require some additional disk capacity without doubling the cost of storage.

VII. REFERENCES

[1] Chen D-Y, Trivedi KS. Closed-form analytical results for condition based maintenance. Reliab Eng Syst Saf 2002;(76):43–51.

[2] Hosseini MM, Kerr RM, Randall RB. An inspection model with minimal and major maintenance for a system with deterioration and Poisson failures. IEEE Trans Reliab 2000;49(1):88–98.

[3] Legat V, Zaludova AH, Cervenka V, Jurca V. Contribution to optimization of preventive maintenance. Reliab Eng Syst Saf 1996; 51:259–66.

[4] Sheu S-H, Yeh RH, Lin Y-B, Juang M-G. A Bayesian approach to an adaptive preventive maintenance model. Reliab Eng Syst Saf 2001;71: 33–44.

[5] Sim SH, Endrenyi J. Optimal preventive maintenance with repair. IEEE Trans Reliab 1988;37(1):92–6.

[6] Tijms HC. Stochastic modelling and analysis: a computational approach. New York: Wiley; 1986.

[7] Vaurio JK. On time-dependent availability and maintenance optimization of standby units under various maintenance policies. Reliab Eng Syst Saf 1997;56:79–89.

[8] Holland, J. H.: Genetic Algorithms. Scientific American 267 (1992) pp. 66-72

[9] Goldberg, D.E.: Genetic Algorithm in Search, Optimization, and Machine Learning. Addison-Wesley, Reading, MA (1989)

[10] Russell, S., and Norvig, P., "Artificial Intelligence: A Modern Approach", second edition, Prentice Hall.

[11] Mitchell, Melanie, (1996), *An Introduction to Genetic Algorithms*, MIT Press, Cambridge, MA.

[12] Fogel, David B. (2000). *Evolutionary Computation: Towards a New Philosophy of Machine Intelligence*. New York: IEEE Press, 140.

[13] Wright, A.H.; et al. (2003). "Implicit Parallelism". *Proceedings of the Genetic and Evolutionary Computation Conference*.

[14] Pinheiro et al; "Failure Trends in a Large Disk Drive Population" Proceedings of the 5th USENIX Conference on File and Storage Technologies (FAST'07), February 2007

[15] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster. IEEE Transactions on Evolutionary Computation 1999.