A Double-Objective Genetic Algorithm for Parity Declustering Optimization in Networked RAID

Xiaoguang Liu, Gang Wang, Jing Liu

Department of Computer Science, Nankai University, Tianjin, 300071, China
{liuxg74, wgzwp}@hotmail.com,jingliu@nankai.edu.cn

Abstract. RAID, as a popular technology to improve the performance and reliability of storage system, has been used widely in computer industry. Recently, the technique of designing data layout in order to fit the requirements of networked storage is becoming a new challenge in this field. In this paper, we present a double-objective Genetic Algorithm for parity declustering optimization in networked RAID with a modified NSGA, we also take *Distributed recovery workload* and *Distributed parity* as two objects to find optimal data layout for parity declustering in networked RAID.

1 Introduction

Since RAID (Redundant Array of Independent Disks) [1] was invited in 1980s, it is becoming the most important technology in storage systems. Especially, RAID 5 has been treated as the most reliable storage standard. However, a shortcoming of RAID 5 is that its performance falls obviously in degrade and reconstruction mode. To solve this problem, parity declustering was introduced by Muntz and Lui[2]. Through parity stripe distribution and Reconstruction Load Balance, parity declustering can improve the performance and reduce the time cost of reconstruction. Holland and Gibson also defined six standards to estimate ideal data layout[3]. It includes *Single failure correcting, Distributed recovery workload, Distributed parity, Efficient mapping, Large write optimization* and *Maximal parallelism*. Following the six standards, Alvarez proved that building an ideal data layout was difficult in most cases[4]. So the problem has been converted to how to find a data layout which is as close to ideal data layout as possible. Many data layouts have been studied, such as BIBD, PRIME,PDDL and RELPR[2-5]. All of them emphasized only parts of the six standards for different applications. However, since ideal data layouts are required by the six standards, the multiobjective optimization algorithm is obviously a nature selection to find ideal data layouts.

One way to solve multiobjective problems is transforming the original problem into a single objective problem by weighting the objectives with a weight vector. But the solution obtained in this way depends on the weight vector used in the process. Genetic algorithm works with a population, so we expect that it can find the Pareto optimal front to our problem.

Many Pareto-based multiobjective GAs are developed in recent years. VEGA(vector evaluated genetic algorithm) performs the selection operation for each objective respectively. The pareto-based ranking GA was proposed by Fonseca and Fleming[6]. An individual's rank equals the number of other individuals in the population by which it is dominated. NPGA (niched pareto genetic algorithm) uses the concept of pareto dominance and tournament selection in solving multiobjective optimization problems[7].NSGA (Non-dominated Sorting Genetic Algorithm) was first implemented by Srinivas and Deb[8]. While it follows the standard genetic algorithm for parent selection and offspring generation, it determines the fitness of the individual using the concept of parato dominance also. To improve the performance, NSGA-2 was proposed in 2000[9], and its source codes also can be download.

In this paper, we use NSGA to find a better data layout for networked RAID systems. Because *Distributed recovery workload* and *Distributed parity* take more weights on the networked RAID systems, we set them as the two objects of our algorithm. To compare the performances of the solution, an experiment based simulated annealing was done simultaneously. The results show that the double-objective Genetic Algorithm can produce better data layout for networked storage. To our knowledge, this work is firstly applied the multiobjective genetic algorithm to solve parity declustering optimization problem.

2 The Double-objective Genetic Algorithm

2.1 The Design of Algorithm

Among the six standards of ideal data layout, the second, *Distributed recovery workload*, and the third, *Distributed parity*, take the largest influence on the performance of networked storage. So we set these two standards as the objects of NSGA. According to the choice of double-objective optimization function，the partial relation on the target space can be converted to pareto dominance on the decision space. After enough iterations, the partial relation can be converted to pareto dominance on the decision space under the control of the double-objective function. At last, we can get a set of pareto dominated solutions. The optimum data layout can be selected from this set.

2.2 The Detail of Algorithm

2.2.1 Objective Functions

In this section, the detail of objective functions is presented.

(1) Weight

During initial period, we set the weight under the following rules,

Rule 1: To all local connected disk, the value of weight, e_{ij}, is 1;

Rule 2: If there is one controller node in the networked storage system, and all storage management job run only on the controller (supposing the number of controller is 0), then we have,

$$e_{ij} = \begin{cases} 1, & (j \bmod m) = 0 \\ e, & (j \bmod m) \neq 0 \end{cases} \tag{1}$$

Rule 3: To the distributed storage system, such as petal, the storage management job can run on more than one node. If disk *i* and disk *j* belong to the same node, then e_{ij} is 1,else e_{ij} is e. Here, *m* is the number of disks.

$$e_{ij} = \begin{cases} 1, & (j \bmod m) = (i \bmod m) \\ e, & (j \bmod m) \neq (i \bmod m) \end{cases} \tag{2}$$

(2) The function of Reconstruction Load

$$H_{WEIGHTED}(L) = \sum (\mathrm{X}_{\mathrm{ij}} \bullet e_{ij})^2 \tag{3}$$

Here, X_{ij} means the number of stripes read from disk *j* while disk *i* broke down, e_{ij} means the cost of the system which reads a stripe unit from disk *j* while disk *i* failed. In order to improve the performance, we should minimize the value of function H.

(3) The function of Parity overhead

$$P_{WEIGHTED}(L) = \sum \mathrm{P}_{\mathrm{i}}^{2} \tag{4}$$

Here, P_i is the number of parity units on disk *i*. Obviously, we should also minimize function P.

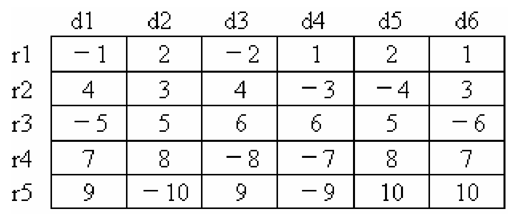
2.2.2 Parameters of Algorithm

The data layout is used as the chromosome in the algorithm. Every data layout is presented as a $r \times v$ matrix. Here, r is the number of lines in the data layout, v is the number of the disks, k is the length of the stripe. Simply, we only consider the situation that v can be divided exactly by k in this paper. The absolute value of the elements in the matrix is the sequence number of the stripes, and the elements which value less than zero are parity unites in stripes. An example(r=5, v=6，k=3) is shown in figure 1.

The Pareto dominate relation on decision space can be described as follows,

$$\begin{aligned} \text{A Dominat B} \Leftrightarrow ((H(A) \le H(B)) \,\&\&(P(A) \le P(B))) \\ \&\&((H(A) < H(B)) \,||\, (P(A) < P(B))) \end{aligned} \tag{5}$$

In order to avoid illegal data layout produced during iterations, we make some restrictions on the intercross and mutation regulars. For example, all lines must be interchanged between two data layouts in intercross, and mutation only interchanges two elements in the same line.



	d1	d2	d3	d4	d5	d6
r1	−1	2	−2	1	2	1
r2	4	3	4	−3	−4	3
r3	−5	5	6	6	5	−6
r4	7	8	−8	−7	8	7
r5	9	−10	9	−9	10	10

Fig. 1. An example of data layout

2.2.3 The Algorithm

The algorithm can be described as follows,

a. Initially, N data layouts are given randomly.

b. The functions of Reconstruction Load (H) and Parity overhead (P) are computed.

c. According to their pareto relations, all the data layouts are divided into m sets.

d. According to the regulars in niche, we set all sets with the shared fitness values in turn.

e. N better data layouts are selected from all sets.

f. The parents are selected by roulette, and N new data layouts are produced after intercross and mutation.

g. The functions of Reconstruction Load (H) and Parity overhead (P) are calculated again. If the difference between the actual value and ideal value is less than the threshold which is defined initially, then the program ends, else turns to step b.

Here, the population size in the first iteration is N, and it is 2N in the others. However, the population, which used for selection, intercross and mutation, is still N in each generation. Specially, these N individuals are the better ones in the 2N data layouts.

2.2.4 Simulated Annealing Algorithm

In order to compare the performance of the double-objective Genetic Algorithm, we also implement a simulated annealing algorithm for the same problem. The simulated annealing algorithm can be described as follows,

a. Initialization. The default values of parameter are set. Such as *length*, the maximal length of Markov chain, T_0, the temperature in initial state, *ngel*, the maximal number of data layouts which have same value, *gen*, the maximal number of iteration.

b. If the number of iteration exceeds *gen*, end the program, else turns to step c.

c. If the number of same data layouts exceeds *ngel*, turns to step g.

d. If the length of Markov chain exceeds *length*, turns to step g.

e. Generate new data layout, and calculate its value of reconstruction function.

f. The length of Markov chain adds 1. If the probability estimation can be accepted, then the new data layout is legal and turns to step d. Else, the number of same data layouts adds 1 and turns to step c.

g. According to *Distributed parity*, if the new data layout is the best solution until now, then records it. The number of iteration adds 1 and turns to step b.

In our experiments, T_0 is 0.5, *length* is 1000, *ngel* is 10 and *gen* is 10.

3 Experimental Results

There are two groups of test in the experiment. One is the optimum data layout in local disk array. The other is the result in distributed storage system. Simultaneously, the best results in this experiment are compared with the best results in theory.

Table 1. The optimum data layout in local disk array (v = 12,k = 6,N = 50)

Lines	ideal X_{ij}	Actual X_{ij}	Difference	Ideal P_i	Actual P_i	Differ-ence
117	53.18	55	3.309%	19.5	20	2.25%
308	140	142	1.428%	51.333	52	1.346%
1121	509.54	512	0.482%	186.833	187	0.358%
4873	2215	2219	0.18%	812.167	813	0.103%

According to the conclusion in reference 10, any X_{ij} has the same value in local disk arrays[10], and the value should be equal to $r \times (k-1)/(v-1)$. Besides, any P_i also has the same value, it equals to r/k. As shown in table 1, X_{ij} and P_i in actual test have nearly reached the best results in theory.

Table 2. The optimum data layout in distributed storage system(r = 117, k = 10, v = 40)

Nodes	H[a]	H[b]	Ideal P_i	Actual P_i
2	5805760	2105600	12	12
4	8140350	3766204	12	12
5	8605310	4335244	12	13
8	9306610	5828024	12	13

[a] *The value of function H using simulated anneaing.*
[b] *The value of function H using double-Objective Genetic Algorithm.*

Using double-objective Genetic Algorithm, the parity units are distributed more even in storage system. As shown in table 2, the value of function H using double-Objective Genetic Algorithm is much smaller than that using simulated annealing. In a word, the double-objective Genetic Algorithm has selected the most two important standards from six standards of ideal data layout. It can produce better data layouts for networked storage.

4 Conclusions

In this paper, we use the double-objective Genetic Algorithm to design the optimum data layout for networked storage system. To the objective functions in the algorithm, we chose *Distributed recovery workload* and *Distributed parity*, which are more important standards to the performance of storage system. The experimental results show that the double-objective Genetic Algorithm can produce better data layout for networked storage. We can conclude that multi-objective Genetic Algorithm is a feasible and effective optimization algorithm for designing ideal data layout. As far as we know, our paper is the first to use multi-objective Genetic Algorithm in the design of ideal data layout, and this application may become popular to multi-objective Genetic Algorithm in the future.

Acknowledgements

This paper is sponsored by NSF of China (90612001), Science and Technology Development Plan of Tianjin, (043185111-14) and Nankai university science and technology innovation fund and ISC.

References

1. Patterson, D., Gibson, G., Katz, R.: A case for redundant arrays of inexpensive disks(RAID). Proceedings of ACM SIGMOD, Seattle, Washington, USA(1998)109-116
2. Muntz, R., Lui, J.: Performance Analysis of Disk Arrays Under Failure. Proceedings of the conference on Very Large Data Bases, Brisbane, Queensland, Australia(1990)162-173
3. Holland, M., Gibson, G., Sieworuk D.: Architectures and Algorithms for On-Line Failure Recovery in Redundant Disk Arrays. Journal of Parallel and Distributed Databases 2(1994)295-335
4. Alvarez, G., Burkhard, W., Stockmeyer, L., Cristian, F.: Declustered Disk Array Architectures with Optimal and Near-Optimal Parallelism. Proceedings of the 25th Annual ACM/IEEE International Symposium on Computer Architecture, Barcelona, Spain (1998)109-120
5. Schwarz, T., Steinberg, J., Burkhard, W.: Permutation Development Data Layout (PDDL) Disk Array Declustering. Proceedings of the Fifth International Symposium on High-Performance Computer Architecture, Orlando, FL, USA (1999)214-217
6. Fonseca, C., Fleming, P.: Genetic algorithm for multiobjective optimization. Proceedings of 5th International conference on Genetic Algorithms, San Mateo, CA, USA(1993)416-423
7. Horn, J., Nafpliotis, N., Goldberg, D.: A niched Pareto genetic algorithm for multiobjective optimization. Proceedings of 1st IEEE conference on Evolutionary Computation, Piscataway, NJ, USA(1994) 82-87
8. Srinivas, N., Deb, K.: Multiobjective optimization using nondominated sorting in genetic algorithms. Evolutionary Computation 2(1994)221-248

9. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A Fast Elitist Non Dominated Sorting Genetic Algorithm for Multi Objective Optimization: NSGA-2. Proceedings of Parallel Problem Solving from Nature (PPSN) 6th International conference, Paris, France(2000) 858-862

10. Schwabe, E., Sutherland, I., Holmer,B.: Evaluating Approximately Balanced Parity-Declustered Data Layouts for Disk Arrays. Parallel Computing 23(1997) 501-523