Using PCA to Improve Evolutionary Cellular Automata Algorithms

Mehran Najafi
Dept. of Computer Science
McMaster University
Ontario, Canada
najafm@mcmaster.ca

Hamid Beigy
Dept. of Computer Engineering
Sharif University
Tehran, Iran
Beigy@sharif.edu

ABSTRACT

The difficulty of designing cellular automatons' transition rules to perform a particular problem has severely limited their applications. Using a genetic algorithm to evolve cellular automata for fining these rules, is a good solution. Conventional evolutionary methods use random test configurations for calculating fitness values of each transition rule. In this paper, we use Principal Component Analysis to build better test configurations. By emphasizing on diversity between test instances in a test plan, we can evaluate rules better and faster as well as increase their accuracy. In this paper, we propose two models based on this idea. Experimental results on density classification and synchronization tasks prove that our methods are more efficient than the conventional one.

Categories and Subject Descriptors

I.2.11 Distributed Artificial Intelligence

General Terms

Algorithms, Performance

Keywords

Genetic Algorithms, Evolutionary Cellular Automata, Improvement in test strategies, Principal Component Analysis

1. INTRODUCTION

Recently, parallel computers based on cellular automata (CAs [1]), have gained attention as a method of designing complex systems. However, the difficulty of designing CAs' transition rules to perform a particular task has severely limited their applications. The evolutionary design of CA rules has been studied by the EVCA group [2] in detail. A genetic algorithm (GA) is used to evolve CAs for calculating transition rules.

For testing a transition rule, the number of possible test cases is too large, so we can not define fitness as the fraction of correct classifications over all possible initial configurations (ICs). Instead, fitness was defined as the fraction of correct classifications over a sample consisting of small number of ICs. Almost in the all proposed methods, these finite ICs have been determined randomly.

In this paper, we interest in evaluating transition rules more accurately. For this purpose, test configurations must prepare different scenarios for each transition rule. So we use Principal Component Analysis (PCA) to select test fields with the maximum diversity. We propose two models based on this idea.

2. Cellular Automata and Evolutionary Cellular Automata

A Cellular Automata (CA) consists of a collection of time-dependent variables S^i_t called the local states, arrayed on a lattice of N sites (or cells). This collection of cells can be organized in one or more dimensions. The collection of all local states is called the configuration. Where S_0 denotes an initial configuration (IC). Typically, the equation of motion for a CA is specified by a look up table that maps a site's neighborhood to a new local state.

Inverse problem in CA concerned with designing transition rules for a desirable CA final status. Evolutionary Cellular Automata [3] is a solution for the inverse problem. In this approach an initial population of chromosomes, which each of them is a candidate transition rule, is created. Then the fitness of each rule is evaluated on some test configurations. After calculating all the chromosomes fitness, some genetic algorithm operators (like as mutation or cross over) are applied on the chromosomes. The next generation is selected from the current modified generation. This procedure will be iterated until a transition rule with satisfactory fitness is reached.

Several modified version of the initial approach have been introduced. Coevolving CA [4] evolves a population of test configurations as well as a population of transition rules in each iteration. Hence evolving test configurations makes learning hard after a few iterations, this method can not be effective. In the rule changing cellular automata [5], each chromosome contains several transition rules instead of only one rule.

3. Proposed Method

With consideration to the all methods which have been introduced so far, it can be concluded that all of them, calculate fitness on random test plans. On the other hand, the number of possible test plans is too far in compare to actual number of test plans which is used in these methods. (For example, in a lattice with 149 binary cells there are 2^{149} different test plans, but only 100 of them are used for calculating rules fitness.). In our proposed methods, we use PCA to extract some different test scenarios (As opposed to random scenarios in other methods) for evaluating each rule better.

Principal Component Analysis (PCA) [6] is a very famous method which used in pattern recognition applications directly or indirectly. The main characteristic of PCA is its feature reduction. PCA can describe a set of instances with a feature space whose dimension is less than the initial space. Basises for the new space are chosen such that when the instances are projected on them, the differences between instances will be maximized. These base vectors are built with considering the covariance matrix of instances (Covariance matrix indicates instances similarities).

Two different models based on PCA are proposed here for generating test plans more efficient:

Model I-According to the first PCA coefficient

1) A set of 100 random test configurations is built. 2) PCA coefficients for all instances in the set are calculated. 3) Instances are sorted according to their first PCA coefficient. 4) From the sorted set, some instances are chosen to add to the test set. This work is done according to their positions (For example in our experiments, we select test plans with indexes 1,25,50,75 and 100 for adding to the test set.) 5) Go to step 1 until the sufficient number of test plans is provided.

Model II- According to some of the PCA coefficients

1) A large enough set of random test plans is created. 2) PCA coefficients for all members of this set are calculated. 3) With using a clustering method (like as K-means [7]) test plans are clustered according to the some of their highest PCA coefficients. 4) From each cluster, a test plan is selected and is added to the test set. (In our experiments, we create 100,000 random test plans, select 10 first PCA coefficients for each instance and use K-means algorithm for their clustering. Also we assume K=100).

It can be shown that differences between first PCA coefficients of a set are equal to the differences between their values from the set average. So the first model can be done with averaging instead of calculating PCA on a set. As a result the first model is faster than the second one. In the other way, the second model selects test plans among more instances as well as compares all of them once. So it gets the better results but in a longer time. Either of these models can be used in the conventional EVCA algorithms as following. (The same algorithm we use in our experiments).

1) Select 100 chromosomes randomly. 2) Use mode I or mode II for extracting 100 test plans. 3) Calculate chromosomes fitness on the test plans. 4) Select 20% of the chromosomes based on their fitness values and transfer them to the next generation. 5) Select two random chromosomes as parents between the above 20% chromosomes 6) Do single point cross-over between them and get the child. 7) Mute the child with a 0.1 probability for each chromosomes cell. 8) Add the child to the next generation. 9) Go to step 5 until remaining 80% of the next generation is provided.

These 9 steps are formed one generation of our evolutionary cellular automata. These generations continue until the generation number is reached to the pre-defined iteration. In the last generation the rule whose fitness is the highest one, will be selected as the final transition rule. (In the conventional method step 2 is substituted by Extracting 100 random test plans).

4. Experimental Results

For examining our models, we compare them with the conventional evolutionary methods in two famous fields. The density classification task and the synchronization task [5]. (We use the same problem definitions and conditions as [5] in our experiments).

The final transition rule which obtained for density classification task without our modification has 49% accuracy while using model I improves accuracy to 57% and model II improves to 62%.

Likewise for the synchronization task, the second case study, the result is improved from 44% (conventional method) to 46% (In the model I) and 57% (In the model II).

5. Conclusions

In this paper, a new approach for the better evaluation of transition rules is proposed. In this method, we use PCA feature space as a measure for extracting test plans which have maximum differences with each other. Two models are proposed for this purpose. Model I use only one PCA coefficients and is faster while model II uses several PCA coefficients and gets better result. Our experiments reveal that the proposed methods converge faster and also have better performance in compare to the conventional EVCA.

6. REFERENCES

[1] Wolfram, S. A New Kind of Science. Book. Wolfram Media Inc. (2002).

[2] Mitchell, M., Crutchfield, J. P., and Hraber, P. Evolving Cellular Automata to Perform Computations: Mechanisms and Impediments. Physica D 75 (1994).

[3] Wolfram, S. Cellular automata as models of complexity, Nature. Vol. 311,(1984).

[4] Paredis J., Coevolving Cellular Automata: Be Aware of the Red Queen!, Proceedings of the Seventh International Conference on Genetic Algorithms (ICGA97), (1997).

[5] Kanoh1 H., Wu2 Y., Evolutionary Design of Rule Changing Cellular Automata, Knowledge-Based Intelligent Information and Engineering Systems, (2003)

[6] Jolliffe I., Principal Components Analysis, Book, Springer (2002).

[7] Kanungo T., Mount D., Netanyahu N., Piatko C., Silverman R., Wu A., An Efficient k-Means Clustering Algorithm: Analysis and Implementation, IEEE Transactions on Pattern Analysis and Machine Intelligence , Vol 24, (2002).