2008 International Symposium on Nonlinear Theory and its Applications
NOLTA'08, Budapest, Hungary, September 7-10, 2008

Towards the Automatic Design of Algorithms for Cellular Wave Computers

Giovanni Egidio Pazienza
and Xavier Vilasís-Cardona
Enginyeria i Arquitectura La Salle
Universitat Ramon Llull
Barcelona, Spain
Email: {gpazienza,xvilasis}@salle.url.edu

Abstract— **Our previous papers prove that the automatic design of programs for the Cellular Neural Network - Universal Machine not only is feasible, but also is convenient in many situations. Now, we discuss on the changes that should be made to such a system in order to apply it to a generic cellular wave computer, without any constraint on the kind of data and instructions.**

I. Introduction

The introduction of the cellular wave computer architecture [1] is a breakthrough in the field of computation, since it is the paradigm that may be most extensively applied in the future. In fact, the last generation of processing devices already contains several processors - sometimes fairly simple - working on a unique complex task, and the resultant synergy allows to go beyond the limits of the traditional digital computers.

This paradigm requires also a new approach to design algorithms and programs, since some of the characteristics of computation on traditional digital computers have become obsolete. For example, the solution of a nonlinear partial differential equations is one of the most complex tasks to perform on a digital microprocessor, but it is the elementary instruction of a cellular wave computer: This change represents a Copernican revolution.

It is possible to prove that cellular wave computers are universal; in other words, there exists a program for a cellular wave computer to perform any possible task. Therefore, one of the main challenges for the future is designing efficiently new programs for this kind of machine.

In [2] a particular cellular wave computer is considered, for which input and output are images (and not image flows) and instructions are binary masks (and not spatial-temporal waves). In this case, it is possible to prove that all the programs have to fit into a fixed structure, and they can be found find automatically by using a a machine learning technique called Genetic Programming (GP) [3], [4]. In this paper we recall these results, defined for a Cellular Neural Network - Universal Machine (CNN-UM), and we extend them to the most general kind of cellular wave computer, describing the steps that should be taken in order to make this system feasible.

The paper is structured as follows: firstly, we give a proof of the universality of the CNN-UM and analyze its consequences; then, we quickly describe GP and show its application to the design of CNN-UM programs; finally, we discuss on how these results can be extended to a general cellular wave computer and delineate future work.

II. On the Turing completeness of the CNN-UM

Proofs of universality are in general complicated and sometimes tedious due to the theoretical nature of Turing machines; for this reason, in most cases people resort to analogies with well-known universal machines, like *Game of Life* [5]. This kind of proof was also used for the Cellular Neural Network - Universal Machine; however, despite its enormous theoretical importance, it does not give any hint about the structure of the CNN-UM programs. This last aspect is generally left to the expertise of the designer, except for specific cases, like the design of Boolean functions [6].

An alternative proof of the universality of the CNN-UM was presented in [2]. Such result, found by drawing an analogy between the CNN-UM and another paradigm equivalent to an arbitrary Turing machine [7], implies that any program for the CNN-UM can be represented as follows

```
Repeat
Evaluate < CNN-UM function >
Until
< Some specific state of the memory >
```

in which the difference between a CNN-UM program and a CNN-UM function is that the former can contain iterative and recursive processes, the latter cannot. Therefore, thanks to this alternative proof, we know that every algorithm for the CNN-UM can be represented as a single repeat-until loop inside which there are only a combination of CNN templates and nested if-then-else.

The intrinsic nature of the proof suggests that the search space constituted by all the possible CNN-UM programs with such structure can be efficiently explored through a technique called Genetic Programming, which converges to a solution faster than random search, and has been already successfully applied to CNNs (see [2] [8]).

III. Exploring the search space through GP

A. Generalities on Genetic Programming

Genetic Programming repeatedly selects and evolves instances from a population of computer programs, eventually achieving a solution for a given computational task. We consider here that the population is represented by means of binary trees, as we did in our previous works; note that other choices are also possible. The GP working principles can be summarized as follows: firstly, a population of initial programs is created; secondly, a fitness value is assigned to each; thirdly, programs are combined through the so-called GP operators to create a population of offspring; fourthly, the new individuals replace the current population; finally, the whole set of operations — called a generation — is repeated until a stop condition is met, and the result is retrieved from the last generation according to a certain criterion. If the various GP parameters — number of generations, population size, etc. — are set correctly, the fitness of the population improves generation by generation, and its superiority to random search is documented in the literature.

In GP it is not infrequent for a large percentage of the population to converge to a suboptimal result, and the subsequent lack of diversity makes practically impossible the creation of new individuals. In general, the species that dominates a given niche depends on the initial conditions and the subsequent history of probabilistic events. Therefore, the negative effect of the premature convergence can be minimized by performing multiple independent runs starting from entirely separate populations; the best-of-run individual is then designated as the result of the group of runs.

B. Fitness function and genetic operators

In a GP system the fitness function indicates how good an individual is. For instance, in a supervised image processing problem the fitness of the actual result may be quantified by measuring its resemblance with the desired output image. Other parameters can also be taken into considerations, like the number of levels and nodes of a tree: the simpler the tree, the better its fitness. These different kinds of fitness can be either combined into a single value through a weighted sum, or used to in some form of Pareto-based selection. Usually, the choice of the most appropriate fitness function strongly depends on the problem of interest.

As for the operators, in our experiments we employ only the three most important ones: reproduction, crossover, and mutation. Each one is applied with a certain probability that can either be fixed *a priori* or changed during the execution of the GP algorithm. The reproduction is the simplest operator, as it just copies an individual from the current generation to the following one; its main function is assuring a sort of continuity between generations. The crossover operator is applied to pairs of individuals probabilistically extracted from the population according to their fitness: a random node is selected within each tree and then the subtrees rooted at such positions are swapped, generating two new individuals that become part of the next generation. Finally, mutation provides diversity to the population, avoiding local minima. Its mechanism will be explained in more detail in Sec. III-C.

C. A GP system to evolve CNN-UM programs

The population of our GP system consists of CNN-UM programs, which evolve to find the best solution for the given problem. The templates that can belong to such programs are selected within a set chosen by the designer. Thanks to this feature, any *a priori* knowledge about the problem is used profitably and the search space can be significantly reduced.

The mutation operator is applied to a single randomly selected individual, and it can act according to three different mechanisms: choosing randomly a cloning template of the algorithm and substituting it for another one; changing the value of the variable of a parametric template; or it can selecting randomly a point of the tree representing the CNN-UM program, and then replace the whole branch from this point upwards - that is, to the input level - with a new subtree not related with the previous one.

The initial population of CNN-UM programs is set randomly using a Ramped Half-and-Half method [3], which assures a very diverse population composed by trees of several different depths. The designer has the possibility to fix the minimum and maximum size - i.e. maximum and minimum number of levels and nodes - of the individuals, which constraint is enforced also during the evolution. Although other methods can be employed, we found that this one gives the best performances in terms of average number of generations to converge to the solution.

In general, in GP the operation set must satisfy the requirements of closure and sufficiency [4]; for our system, this property is proved in [2].

IV. Applying the genetic approach to find programs of cellular wave computers

In the previous sections we summarized the main results proved in [2] about a Cellular Neural Network - Universal Machine: universality, general structure for the programs, and validity of GP as a searching method in the solution space. However, the model we have considered so far differs from a generic cellular wave computer in several aspects: Firstly, the input (and the output) is a single image, and not an image flow; secondly, instructions are masks, and not spatial-temporal waves; finally, in [2] was not analyzed any example including branches or global conditions. In order to extend our approach to a generic cellular wave computer, we need to analyze how to include these features into the existing system. As for the first point, the fitness measure can be easily generalized to an image flow. There exist several methods in literature, however the simplest one is just sampling the input and the output in a finite number of points, and then performing the current fitness measure on each pair input/output belonging to this set of images. If the sampling is well-performed — i.e. the sample frequency is large enough — the information retrieved using a finite number of images will be sufficient to evaluate

the whole flow.
The second point, regarding the kind of instruction, can be included easily in the our system, since a partial differential equation can be expressed through a standard CNN template [9]. This is not a limit to the application of a genetic approach, though it is important to adjust the mutation operator to avoid undesired combinations of numbers in the feedback matrix A.
Finally, it is true that in [2] no practical examples showing the application of branches and flow control structures were shown, but only because the experiments considered were not complex enough to require such features. We are currently working on new examples in which it is necessary to take them into account.

V. Discussion

In this paper we have summarized the results that made possible the automatic design of programs for the CNN-UM, and we have also showed that they can be extended, with minimal variations, to a generic cellular wave machine, including more complex features. In the near future we plan to carry out experiments to confirm, or amend, our expectations, showing that a genetic approach is valid tool to design programs for the cellular wave machine.

Acknowledgment

The authors would like to thank Prof. Riccardo Poli for his contribution to the work. G. E. Pazienza thanks DURSI and the 'Generalitat de Catalunya' for the FI grant.

References

[1] T. Roska, "Cellular wave computers for nano-tera-scale technology — beyond boolean, spatial-temporal logic in million processor devices," *Electronics letters*, vol. 43, no. 8, 2007.
[2] G. E. Pazienza, R. Poli, and X. Vilasis-Cardona, "An alternative proof of the universality of the CNN-UM and its practical applications," in *Proc. 11th International Workshop on Cellular Neural Networks and their Applications (CNNA'08)*, Santiago de Compostela, Spain, July 14-16 2008.
[3] J. Koza, *Genetic programming - on the programming of computers by means of natural selection*. Cambridge, MA: MIT-Press, 1992.
[4] R.Poli, W. B. Langdon, N. F. McPhee, and J. R. Koza, "Genetic programming: An introductory tutorial and a survey of techniques and applications," University of Essex, UK, Tech. Rep. CES-475, Oct. 2007.
[5] E. Berlekamp, J. H. Conway, and R. K. Guy, *Winning ways for your mathematical plays*. New York: Academic Press, 1982.
[6] K. R. Crounse and L. O. Chua, "The CNN universal machine is as universal as a Turing machine," *IEEE Trans. Circuits Syst. II*, vol. 43, no. 4, pp. 353–355, Apr. 1996.
[7] A. Teller, "Turing completeness in the language of genetic programming with indexed memory," in *Proceedings of the 1994 IEEE World Congress on Computational Intelligence*, vol. 1. Orlando, Florida, USA: IEEE Press, 27-29 June 1994, pp. 136–141.
[8] V. Preciado, M. Preciado, and M. Jaramillo, *Genetic Programming for Automatic Generation of Image Processing Algorithms on the CNN Neuroprocessing Architecture*. Springer, 2004, vol. 3040, pp. 374–383.
[9] L. Kék, K. Karacs, and T. R. (Eds.). (2007) Cellular wave computing library, version 2.1 (templates, algorithms and programs). [Online]. Available: http://cnn-technology.itk.ppke.hu/Library_v2.1b.pdf