Proc. of the ICONIP Conference (International Conference on Neural Information Processing), Workshop on Artificial Immune Systems, vol. **3**, pp. 1469-1473, Singapura 18-22 November, 2002.

IMMUNE, SWARM, AND EVOLUTIONARY ALGORITHMS PART II: PHILOSOPHICAL COMPARISONS

Leandro Nunes de Castro {*lnunes@dca.fee.unicamp.br*}
http://www.dca.fee.unicamp.br/~lnunes
Computer and Electrical Engineering School (FEEC)
State University of Campinas (Unicamp), Brazil

ABSTRACT

In the first part of this paper, the standard evolutionary, immune, and swarm algorithms were reviewed. This second part starts by presenting a philosophical discussion about some similarities and differences among the various approaches in terms of their basic components, structure, knowledge storage, adaptation paradigm, interactions, and metaphor. Then, the identification of the main features of each technique is performed in order to shed some light into how to create hybrid algorithms.

1. SIMILARITIES AND DIFFERENCES

1.1. General Features

All the three approaches under discussion have similar features. First, they are all composed of a set (or sets) of individuals that interact with the environment and/or each other. These individuals are represented using a certain structure, such as attribute strings, vectors and networks. All of them present two important and intertwined properties: *self-organization* and *emergence*. There are many debates within the scientific community regarding a unified definition of emergence. In evolutionary, swarm and immune algorithms, it is possible to observe complex behaviors emerging out of the interactions of individuals and the environment, and with each other. The algorithms only specify how each individual behaves and with whom it interacts, there is no central control. The result – recognition of a given antigen or optimization of a function – is an outcome of (emerges from) individual behaviors. In self-organizing systems, there is no central control; the 'intention' of the population is distributed throughout its members who, by themselves, are unaware of the role they are playing. Individuals follow simple rules, and a certain pattern of behavior emerges from the lower-level behaviors.

1.2. Basic Components

The 'artificial chromosomes' in an evolutionary algorithm can have several types of configuration. The most common are binary strings used in genetic algorithms. Real-valued vectors are employed in evolutionary programming and evolution strategies. Genetic programming deals with evolving computer programs that are represented as trees.

In particle swarm optimization algorithms, the individuals of the population (particles) are either binary strings or real-valued vectors. As ant colony algorithms are mostly employed to discrete (combinatorial) optimization, the individuals of the population (ants) are usually represented as a string of integer numbers. Another important component of an ACO algorithm is the pheromone level of each edge that composes the discrete search problem.

Artificial immune systems have four main types of representation: binary, real-valued, integer and symbolic. However, other structures can be used to represent cells and molecules in an artificial immune system, such as DNA sequences.

1.3. Structure

With the exception of network-based artificial immune systems, in all the other algorithms the components of the systems are structured around matrices, which represent *populations* (of chromosomes), *swarms* (of particles), *colonies* (of ants), or *repertoires* (of immune cells and molecules). These matrices can have either fixed or variable sizes. These components are discrete in the sense that they are not linked to any other element through a connection. They interact with each other indirectly as will be discussed later. In PSO algorithms, connections exist between elements indicating with whom they interact, i.e., exchange knowledge. In the case of immune networks, the components of the system are connected with each other via a connection link. This link has a strength, similarly to neural networks, that quantifies the degree of interaction (similarity or difference) between two elements. Note however, that the connection strength in neural networks weights the input signal to the neuron, while the connection in immune networks quantifies the degree of interaction and thus, they have different meanings and play different roles.

1.4. Knowledge Storage

Knowledge here corresponds to the information carried by each individual of the population, and that is a result of their interaction with the environment and each other (adaptation). Therefore, knowledge in all cases is stored in the basic components of the system, i.e., in individual chromosomes, in a particle, in an 'artificial ant' and the pheromone level of each edge, and in an attribute string of an AIS. In the case of immune networks, the connection strengths among individuals also carry information.

1.5. Adaptation

Adaptation is certainly the most distinct feature of all paradigms. It corresponds to how the components of the system vary over time, i.e., how the *dynamics* (behavior) of the system changes along the iterative procedures of adaptation. Two main types of adaptation procedures can be identified: *evolution* and *learning*.

Evolution can be defined as a change in the genetic composition of a population of individuals during successive generations, as a result of natural selection acting on the genetic variation among individuals, and resulting in the development of new species. From a computational perspective, an evolutionary algorithm is a population-based search technique that incorporates reproduction, genetic variation and selection processes. Learning by contrast, can be understood as a change in behavior as a result of previous experience, and interaction with the environment and other individuals.

As evolutionary algorithms were developed taking inspiration from natural evolution, their procedures of adaptation are evolutionary in nature. In the case of swarm algorithms however, the type of adaptive procedure incorporated is controversial. For instance, in a comparative paper [7] between genetic algorithms and particle swarm optimization, the authors start their work with the following sentence: "This paper compares two evolutionary computation paradigms: genetic algorithms and particle swarm optimization." My view is that PSO algorithms do not belong to the evolutionary paradigm, but to the learning paradigm.

To give some support to my claim, let us take a look at equation (1) in Part I of this paper. This general equation describes how a particle varies (moves around the space) over the iterative procedure of adaptation. The first important aspect to be noticed is that the current coordinate of a particle is a result of its past coordinate values plus another vector that indicates in which direction and at what distance the particle is going to move. The vector to be added to the particle is a function of the previous move of the particle, the influence of its previous position, the position of a number of neighbors, its position which led to the best performance so far, and the position that led to the best performance so far in the neighborhood, as described by equation (3) in Part I. In PSO, particles do not reproduce; the population of particles does not change in size even temporarily, and there is also no selection. Therefore, PSO algorithms cannot belong to the evolutionary paradigm. *There cannot be evolution without reproduction and selection* (cf [3]).

If one takes a closer look at artificial neural networks, ANN, (not discussed in these two papers, but another very important biologically motivated paradigm), it is possible to notice some similarities between these two approaches: PSO and ANN. The most remarkable one being the way particles and connection strengths are updated. In ANN, the weight vector $\mathbf{w}_i$ of a given neuron i is updated according to the following rule:

$$\mathbf{w}_i(t) = \mathbf{w}_i(t-1) + \Delta\mathbf{w}_i(t) \quad (7)$$

where $\Delta\mathbf{w}_i$ is the adjustment vector to be added to the neuron weight vector.

The similarities between PSO and ANN can also be found in the definition of a neighborhood function between the elements of the system. Alike PSO, in neural networks the elements of the system are connected to a number of other elements. There is a remarkable distinction however, in the sense that the connections in neural networks can usually assume a number of values, while in PSO algorithms the connections are either zero or one, meaning that two individuals interact or not with a given neighbor. Distinction also exists in the fact that 'artificial neurons' are information-processing elements, while particles are basically information-storage elements. Concerning neighborhood, some neural network models, such as Kohonen's self-organizing maps [12], also take into account the neighborhood of individual neurons to define the influence a neuron is going to exert in its neighbors, similarly to the approach adopted in PSO.

In [2], the authors liken ant colonies to connectionist systems, supported by the fact that individual ants interact with each other (indirectly) via pheromone trails. The network in this case, is a network of communication in which individual elements are not truly, but virtually, connected with each other. This view is not unique to them; other authors (e.g., [15]) also defend the viewpoint that ant colonies can be compared with biological neural networks. In the comparison made by [2], the authors liken pheromone trails to synaptic strengths and pheromone updating to weight updating. Pheromones are updated by reinforcing paths that lead to good solutions and iteratively evaporating the trails (Step 1.3 of Algo 3). Again, there is no reproduction and no selection, thus ACO algorithms cannot be classified as a type of evolutionary algorithm as well. They belong indeed to the learning paradigm.

Before turning the discussion to artificial immune systems, I would like to conclude by saying that I believe some researchers have been arguing PSO and ACO algorithms are types of evolutionary algorithms because EAs are population-based strategies used mainly for search (optimization) purposes. PSO and ACO algorithms share these same features and main application domain with EAs, but these are not sufficient to qualify them as evolutionary algorithms. As ACO and PSO do not usually use any information about desired goals, they belong to the self-organized learning paradigm.

The case of immune algorithms though, is slightly different. For instance, the negative selection algorithm involves affinity (fitness) evaluation and selection, but does not involve either reproduction or genetic variation. Is 'natural selection' (the main step of negative selection) sufficient to qualify evolution? My viewpoint is again 'no'. If the population of individuals does not actually change (variate), then evolution cannot occur. However, the negative selection algorithm is also not part of the learning paradigm, because no adaptation takes place. It is basically a *selection strategy* responsible for building a set of detectors that do not recognize any self individual.

The clonal selection algorithm, in contrast to PSO, ACO and negative selection, has all the steps involved in an evolutionary algorithm: reproduction and genetic variation, affinity (fitness) evaluation, and selection. The question remains thus in regards to what is the difference between a CLONALG and an EA. The differences are subtle. Clonal selection algorithms are primarily based upon affinity-proportionate reproduction, mutation and selection. But indeed CLONALG is a type of evolutionary algorithm inspired by the immune system. Due to a lack of space, I am not going into details as to the differences between a particular CLONALG implementation and specific EAs, but a discussion can be found in [6].

The last immune algorithm described is an immune network model. It is important to note that the algorithm presented in Algo 6 is generic. There are many variations of it that use some of the steps described and that incorporate other steps as well. This is true for all the algorithms discussed, but for the particular case of immune networks there seems to be an even greater variety of algorithms. Without knowing how some of the steps performed by an immune network are accomplished, it is hard to discuss which kind of adaptation is involved in immune networks. However, it is possible to say that immune networks adapt basically by altering the number and attributes of individual cells (Steps 1.3 and 1.5 of Algo 6). Some immune network models present evolutionary procedures of adaptation, while others present an adaptation more akin to learning. Algorithms incorporating a hybrid between learning and evolution can also be found (e.g., [5]).

It is appropriate to highlight some remarkable differences between immune and neural networks as well. The cells in immune networks are information-processing and storage elements. The computation they perform is by determining their stimulation level (Equation (6)), which takes into account the influences exerted by other cells in the network and by antigens (external stimuli). By contrast, artificial neurons perform a linear combination of the neuron inputs and its weight vectors. The neuron output is the result of the application of an activation function to this linear combination. In addition, connections in an immune network correspond to the degree of interaction between two cells, while in neural networks they quantify the input stimuli to the neuron. Immune cells are usually distributed over the space in a way that tends to follow (mimic) the spatial distribution of the universe of antigens. Artificial neurons are usually connected in a predefined structure (single- and multi-layer feedforward or recurrent networks). Despite being completely different types of networks, composed of different units and architectures, they often share domains of application, such as pattern recognition and classification. Comparisons between immune and neural networks can be found in [5].

Immune networks and clonal selection algorithms, though described to perform pattern recognition in Algos 5 and 6, have also been used in function optimization.

1.6. Interaction with Other Components

In evolutionary algorithms, individual chromosomes interact with each other in an indirect fashion, via crossover operators and a fitness function. As the fitness function evaluates the quality of each individual, it serves as a measure for selecting the individuals that will survive in detriment of those that will die off.

In PSO algorithms, the particle-updating rule updates the set of coordinates of a given particle based upon its coordinates that led to its best performance so far. It also takes into account a number of structural neighbors of the particle. This way, there is an exchange of information among particles in a neighborhood. The goodness function in PSO algorithms is a bit more restrictive than the fitness function in EAs. For goodness refers to a comparison of the current particle with its previous best set of coordinates, and with a set of neighbors, while fitness in EAs is a relative measure usually used to compare an individual with the whole population.

In ACO algorithms, individual ants interact with each other to the extent that they leave a virtual pheromone trail along the path they visited. This trail serves as a sort of *positive feedback* (*reinforcement*) mechanism that stimulates other ants to follow this path. The goodness of each ant is evaluated to decide which ant presented the best performance among all the ants in the colony.

The interaction that exists between the elements of a negative selection algorithm involves the comparison between the elements of a set of candidate detectors and the elements of the self-set. If they match, the candidate detector is eliminated; else it is kept in a set of detectors. In the monitoring part of Algo 4, detectors are matched against other elements to monitor for nonself.

Clonal selection algorithms, as inspired by the clonal expansion of immune cells followed by affinity maturation, do not involve the crossing over of genetic material between members of the repertoire. Individual cells reproduce under a cloning procedure, subjected to a high error, namely *somatic hypermutation*. Similarly to EAs, immune cells interact indirectly through their affinity measures, that quantify how good they are in recognizing an invading antigen, and only the better ones survive.

Immune networks are composed of a set or sets of interconnected elements, which, as such, interact directly with each other. A high connection strength value between two cells means that they have a strong interaction, and vice-versa. This interaction is embodied in equation (6) – Step 1.4 of Algo 6.

1.7. Interaction with the Environment

In all algorithms, the interaction of individual elements with the environment is clear. In evolutionary algorithms, there is a fitness function that evaluates how good each individual is regarding the search it is performing. In swarm algorithms (PSO and ACO) there is a goodness function that quantifies the quality of the solution obtained by each particle or ant. Immune algorithms have an affinity measure that indicates how good each immune cell is at recognizing a given antigen. When these algorithms are used to perform search and optimization, the affinity function is usually termed fitness function as in EAs. This is mainly because clonal selection algorithms and some immune networks are evolutionary in nature, thus fitness is a suitable name to indicate the quality measure of individuals. It is also the case to find immune algorithms that have affinity and fitness measures: affinity quantifying interactions within the system itself, and fitness quantifying interactions with the environment.

1.8. The Metaphor

What all these algorithms have in common is a development inspired by nature. They are all part of a computational intelligence paradigm broadly referred to as *computing with biological metaphors* [14].

Computing with biological metaphors can be a result of two front lines of research. One is by observing how nature deals with problems and trying to mimic these problem-solving techniques (*inspiration*). The other is by the realization that some models developed by theoretical biologists and naturalists can be used to solve problems in many domains (*use of models*), mainly computer science and engineering.

Genetic algorithms were developed as a result of the study of adaptation in natural systems [9]. In [1], the author theorized that similarity between pairs of individuals can result in the spread of culture, a model that led to the proposal of an algorithm to solve optimization problems [10]. Ant colony optimization algorithms have been inspired by the experiments of [8] about the self-organizing behavior of the Argentine ants while foraging for food. The inspiration of immune algorithms range from the application of theoretical models to computational problems, to the development of algorithms by studying how the immune system behaves in some situations [4]. Another classical example is that of artificial neural networks. By studying the activity of individual neurons in the nervous system of human beings, [13] came to develop what is to date known as the first model of an 'artificial neuron'. This model has been widely used and adapted by the ANN community. As a last example of the use of metaphors for computing, there is the case of the *simulated annealing* algorithm [11]. Annealing corresponds to a physical process where a crystal is cooled down from the liquid phase to the solid phase on a heat bath. If the cooling is done carefully enough, the energy state of the solid at the end of the cooling stage is at its minimum. The simulated annealing algorithm is a process analogous and inspired by the annealing of a crystal.

Despite having been developed using very different sources of inspiration, the discussion presented about the algorithms demonstrate some similarities in scientific thinking. An important issue to be raised is concerned with the accuracy of the metaphor and its importance for the understanding of how to implement and study the behavior of a particular algorithm. All the approaches reviewed (evolutionary, swarm, and immune), have algorithms that can be written in abstract algebraic symbols, similarly to any other algorithmic process, without requiring a great knowledge of its biological motivation. However, it was the metaphor that enabled researchers to conceptualize and create a computational tool in the first place, and it is the metaphor that helps (allows) people to understand how the algorithms behave. It would be much more difficult to understand how all the different approaches work if we disregarded the metaphors that led to their development.

2. ON THE DESIGN OF HYBRIDS

As one last issue I would like to discuss in this paper, there is the possibility of integrating one or more of these strategies in order to create useful hybrids. To do so, it is important to first remark what are the main features of each approach.

The idea of using specific mechanisms of one technique into the other is much more straightforward than the suggestion of a high-level abstraction of how to hybridize these techniques. For example, in [7] the authors have already conceived the use of an elitist strategy, common in GA applications, to PSO algorithms. But that is not the line I am going to pursue here. The aim is to identify the remarkable features of each paradigm and to instigate the reader about the possibility and benefits of incorporating the features of one approach into another.

The processes of reproduction with inheritance, genetic variation and selection allow evolutionary algorithms to generate increasingly fitter individuals in an unknown environment. Particle swarm algorithms are rooted in the idea that the exchange of information (learning from previous experience and interactions with neighbors) is important for the creation of subpopulations of individuals that share some 'knowledge'. The process of sharing information also allows these individuals to become increasingly fitter to the environment. Ant colony algorithms are based on the concept of indirect communication via a positive feedback mechanism, i.e., while ants explore the environment they release a 'chemical' indicating good paths to find a solution. Some immune algorithms use selective strategies to define 'apt' individuals to compose a population, others use evolutionary procedures of adaptation in which individuals of the population suffer variation inversely proportional to their performance. In addition, immune algorithms also present individuals interconnected in a network fashion.

All these features suggest a few avenues for the creation of hybrid algorithms. For instance, PSO algorithms suggest that the exploration of a given neighborhood is interesting for the location and maintenance of groups of individuals sharing similar features. Actually, similar ideas have already been incorporated in models such as evolutionary algorithms through the use of speciation methods, in which an individual is only allowed to reproduce with those on its immediate neighborhood, given a certain neighborhood criterion. Nevertheless, an important feature of the PSO algorithm is that neighborhood is defined by the structure of the population of particles, not by their spatial location as in speciation methods for EAs. The question still remain thus, as if it would be useful to employ these ideas of structural neighborhood in EAs; and also in the other approaches as well, such as ACO and immune algorithms.

In ant algorithms, a virtual pheromone trail is attributed to each traversed by an ant. This way, the knowledge is distributed between the environment and the actual elements that are performing the search. Can we actually incorporate this idea of a pheromone trail in the other algorithms so that the environment can provide some hints about its portions that are being explored?

An evolutionary equivalent to this discussion would be concerning the possibility of incorporating reproduction with inheritance and selection in PSO and ACO. In the case of immune and evolutionary algorithms, could we use fitness proportionate mutation in EAs?

3. FINAL REMARKS

This part of the paper classified the approaches reviewed into a learning (swarm, immune) and/or evolutionary paradigm (evolutionary, immune), and presented a number of similarities and differences among the many approaches discussed. It also discussed the relevance of the metaphor for the development and understanding of a biologically motivated paradigm. This part also highlighted some of the main features of each technique that could be beneficially used for the creation of hybrid algorithms.

ACKNOWLEDGMENTS

The author thanks CNPq (Profix, Proc. n. 540396/01-0) for the financial support.

4. REFERENCES

[1] Axelrod, R. (1997), "The Dissemination of Culture: A Model with Local Convergence and Global Polarization", *Journal of Conflict Resolution*, 41, pp. 203-226.

[2] Bonabeau, E., Dorigo, M. & Theraulaz, G. (1999), *Swarm Intelligence From Natural to Artificial Systems*, Oxford University Press.

[3] Darwin, C. R. (1959), *The Origin of Species*, Wordsworth Editions Limited (1998).

[4] de Castro, L. N. & Timmis, J. (2002), *Artificial Immune Systems: A New Computational Intelligence Approach*, Springer-Verlag.

[5] de Castro, L. N. & Von Zuben, F. J. (2001), "Immune and Neural Network Models: Theoretical and Empirical Comparisons", *International Journal of Computational Intelligence and Applications*, 1(3), pp. 1-19.

[6] de Castro, L. N. & Von Zuben, F. J. (2002), "Learning and Optimization Using the Clonal Selection Principle", *IEEE Transaction on Evolutionary Computation*, Special Issue on AIS (in print).

[7] Eberhart, R. C. & Shi, Y. (1998), "Comparison Between Genetic Algorithms and Particle Swarm Optimization", *Lecture Notes in Computer Science*, 1447, pp. 611-616.

[8] Goss, S., Aron, S., Denebourg, J. –L., & Pasteels, J. M. (1989), "Self-Organized Shortcuts in the Argentine Ant", *Naturwisschaften*, 76, pp. 579-581.

[9] Holland, J. H. (1992), *Adaptation in Natural and Artificial Systems*, 2nd Ed., MIT Press.

[10] Kennedy, J. (1998), "Thinking is Social: Experiments with the Adaptive Culture Model", *Journal of Conflict Resolution*, 42, pp. 56-76.

[11] Kirkpatrick, S., Gelatt Jr., C. D. & Vecchi, M. P. (1987), "Optimization by Simulated Annealing", *Science*, 220(4598), pp. 671-680.

[12] Kohonen T. (1982), "Self-Organized Formation of Topologically Correct Feature Maps", *Biological Cybernetics*, 43, pp. 59-69.

[13] McCulloch W. & Pitts W. (1943), "A Logical Calculus of the Ideas Immanent in Nervous Activity", *Bulletin of Mathematical Biophysics*, 5, pp. 115-133.

[14] Paton, R. (ed.) (1994), *Computing with Biological Metaphors*, Chapman & Hall.

[15] Solé, R. & Goodwin, B. (2002), *Signs of Life: How Complexity Pervades Biology*, Basic Books.