# Zooming In, Zooming Out:
# A Framework for Hierarchical Genetic Algorithms

**Jennifer Seitzer**
Department of Mathematics and Computer Science
Rollins College, Winter Park, FL 32789
jseitzer@rollins.edu

## Abstract

We present a framework of algorithms and techniques involving hierarchical genetic algorithms. These algorithms attempt to generate models and solutions at both the structural and atomic levels simultaneously using atomic and global chromosomes. Complex adaptive emergent systems (CAESs) are comprised of heterogeneous, interacting, adaptive agents that exhibit the properties of self-organization, emergence, and connectivity *[Ahmed2005].* This paper presents early work and is meant to serve as a starting point in the synthesis of complex adaptive emergent systems using HGAs. Two implementations are discussed[1] that exemplify the compositional nature of hierarchical genetic algorithms and their applicability to both modeling and iteratively generating complex adaptive emergent systems.

## Introduction

A Hierarchical Genetic Algorithm (HGA) is an algorithmic technique of artificial intelligence that converges on a solution at both the atomic and structural levels. Complex adaptive and emergent systems (CAESs) are ubiquitous in our world—appearing as beehives, to social systems, to the brains in our heads that contemplate them. The modeling techniques used for understanding and predicting behavior of these systems in the past been ordinary differential equations, cellular automata, evolutionary game theory, agent based models, and networks *[Ahmed2005].* Because of their implicit nested structure, self-organizing and iterative nature, however, we contend that HGAs show great promise to provide an alternative method of modeling CAESs. Moreover, by using HGA's to intentionally generate systemic simulations of CAESs, we expect to gain insight as to how CAESs are triggered, internally controlled, and sustained.

The work presented here includes the author's taxonomy of techniques based on the nesting of chromosomes and of the overarching *generate-measure-select* algorithm canonical to genetic algorithms. Two computer systems from recent work are discussed in the context of the framework presented here.

## Background and Basics

The thrust of this work lies in the simultaneous evolution of a solution at two different abstraction layers of any given system: the atomic and the structural. How the simultaneity is executed – via interleaving or as an emergent by-product of a finer level's evolution – is largely what defines each of the algorithms presented. We first offer some fundamental definitions and algorithms.

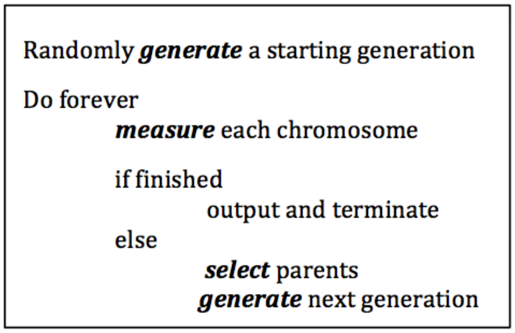
### Genetic Algorithms and Hierarchical GAs

Genetic algorithms (GAs) form a technique of artificial intelligence that iteratively converges on a solution by employing techniques borrowed from genetics. Solutions are iteratively refined executing a series of *generate-measure-select* sequences as shown in Figure 1 below. Problem states (chromosomes) are represented as binary strings, and are iteratively improved upon until a near-optimal solution is attained.

The *measure* step in the algorithm involves the invocation of a *fitness function* to produce a valuation of the merit of that chromosome. The *selection* step involves a ranking mechanism to choose those chromosomes that will serve to produce future chromosomes for the next iteration. The *generate* step involves the creation of future chromosomes using the binary manipulation operators of mutation, selection, and crossover [Holland 1989].

[1] One of these systems was presented in a previous paper. Both systems are used here for illustrative purposes of the framework being presented.

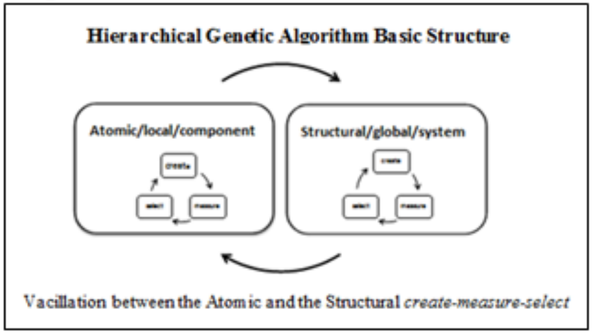
**Figure 1: Underlying Algorithm of GA**



```
Randomly generate a starting generation
Do forever
        measure each chromosome
        if finished
                output and terminate
        else
                select parents
                generate next generation
```

A hierarchical genetic algorithm (HGA) converges on a solution at both the atomic and structural levels (see Figure 2). They are particularly effective in domains where both fundamental building blocks and optimal configuration from structure among the building blocks is sought. HGAs use a nested *create-measure-select* algorithm in both the small and the large. It is exactly how the nesting takes place, both algorithmically and within the chromosomal syntax, that determines the author's taxonomy described in this paper.

Figure 2: HGA Internal Structure


Hierarchical Genetic Algorithm Basic Structure
Atomic/local/component
create
select
measure
Structural/global/system
create
select
measure
Vacillation between the Atomic and the Structural *create-measure-select*

## How HGAs Have Been Used

In past work, the author and her students have used HGAs by strategically ordering selected chromosomes and by retaining unselected chromosomes of the building block. For example, in our music generation system, MELEC [Calvo2011], we intentionally included a series of chromosomes to musically reflect the motif's evolution. Thus, part of the overall form of the musical work was the embodiment of the process of evolution. Our HGA formed an overall system (musical piece) that structurally depicted the evolutionary process of the atomic level building block (the musical motif).

## MELEC (Meta-Level Evolutionary Composer)

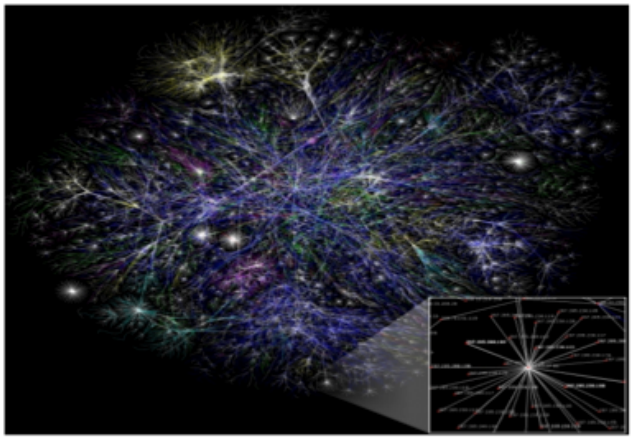
In [Calvo2011], our implementation of the meta-level evolutionary composer (MELEC) system was shown to be capable of autonomously generating melodies through the use of HGAs. The system uses a GA to generate musical motifs and iteratively refines them through evolving generations. It utilizes nested evolutionary computation: one computation generates motifs and another arranges them to form the musical structure. Since each motif is not altered in successive computations, MELEC generates melodies that stimulate a sense of familiarity in listeners by playing the motifs in an order that reflects their evolution. The significance of this work is that it pushed the boundaries of GAs into a realm of performing simultaneous evolution at both the atomic and structural levels while including a "history of evolution" within the solution itself.

## Complex Adaptive Emergent Systems (CAESs)

CAESs are made up of components nested in larger components, and often behaviorally defy compositionality by exhibiting emergent properties. By definition, CAESs exhibit the following properties: emergence, co-evolution, sub-optimality, variety, connectivity, simple rules, iteration, self-organization, and nested systems [Eindhoven, 2004].

Because of their implicit nested structure, self-

**Figure 3: Example Complex Adaptive Emergent System**



Comprised of heterogeneous, interacting, adaptive agents exhibiting properties of self-organizing emergence and connectivity [Ahmed 2005].

organizing and iterative nature, the author contends that HGAs show great promise to provide a more effective and accurate form of modeling of CAESs. Moreover, by using HGA's to intentionally generate systemic simulations of CAESs, we expect to gain insight as to how CAESs are triggered, internally controlled, and sustained. By applying the nested HGA formalism, we hope to expose and more clearly define a new genre of problems: those that can be purposefully evolved with a *semi-controlled emergent behavior* (that is, through the direction of an HGA).

## The HGA Taxonomy

The following algorithms are based on evolution at the atomic and structural levels by alternating the execution of respective cycles of *generate-measure-select*. In so doing, the solutions generated all exhibit an implicit self-organizing, iterative, and nested structure.

### Two Formalisms

We have devised two formalisms of HGAs: composite and nested. In the *composite* formalism, atomic and systemic chromosomes share at least one property or goal. Thus an intentional modification to an atomic chromosome will directly affect the encompassing structural chromosome. In the *nested* HGA formalism, however, atomic and systemic chromosomes evolve independently attempting to attain seemingly unrelated goals. In either the composite or the nested HGA formalism, the atomic chromosome may be a substring of the structural chromosome.

### Composite HGAs

The salient characteristic of composite HGAs is that the atomic and structural levels share at least one goal. Thus, the atomic and systemic chromosomes share at least one property. The valuation (fitness) functions tend to be hierarchical or recursive, and the algorithm may order and retain un-selected chromosomes at the atomic level in construction of a chromosome at the outer, structural level. Evolution of the atomic and structural levels takes places separately. As mentioned earlier, MELEC exemplified this technique in its production of music. Other examples of genres of systems that could exhibit composite properties are those related to computer art using fractals.

### Nested HGAs

Nested HGAs simultaneously evolve atomically and systemically. By using separate fitness functions, however, atomic and systemic chromosomes evolve attempting to attain seemingly unrelated goals. There is an implicit nesting of chromosomes where the systemic chromosome is (possibly partially) composed of atomic chromosomes. The amalgamated structural chromosome, however, is interpreted differently using its own higher level fitness function that realizes the overarching system goal.

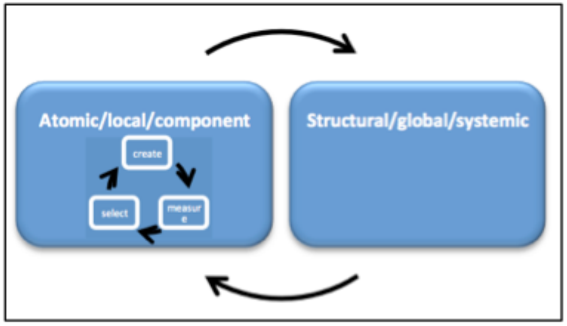
To date, we have experimented with two different sets of systemic/atomic chromosome configurations. There are, however, numerous more possibilities. Specifically, nesting naturally depicts layers of the structure of the problem being modeled. Thus, the nesting of atomic chromosomes within the systemic one can structurally mimic the real-world system being modeled.

An intuitive way to exemplify the nested HGA approach is to consider individual atomic-chromosomes as individual binary strings. The concatenation of the binary strings forms the meta-chromosome and can be equivalently interpreted as a string of hexadecimal digits each comprised of four bits. Thus, we have simultaneously represented two levels of structure with one binary string: the atomic bit level, the structural hex level. Using the *generate-measure-select* cycle on either level could have evolutionary effects on the other.

### HGA Algorithms

The Emergent HGA algorithm is a nested HGA. Shown below in Figure 4, the systemic solution is evolved as a by-product of atomic genetic operations. Even though no explicit operations are applied at the structural level chromosome, that chromosome is transformed 'emergently' by the directed evolution of its constituent atomic chromosomes.

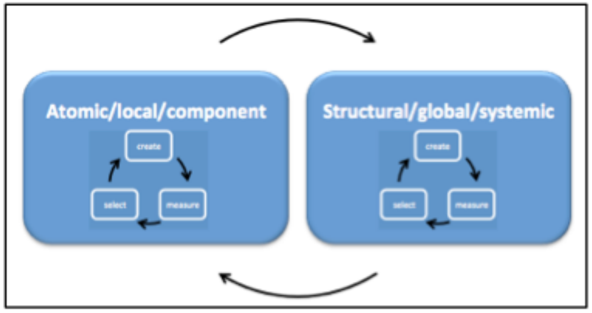
**Figure 4: Emergent HGA Algorithm**



Simultaneously evolving systemic and atomic structure by participating in an inner (atomic level) iterative cycle of *generate-measure-select*

In the Level Alteration Algorithm shown below in Figure 5, a controlled level of alternation between the atomic and system components is used. Each level has its own set of genetic operations randomly assigned. In this approach, genetic operation at the structural level will preserve atomic chromosomal boundaries. The algorithm is exemplified in our graph theoretic problem below.

**Figure 5: Level Alternation Algorithm**



Evolving systemic and atomic structure by interleaving between inner (atomic level) and outer (structural level) iterative cycles of *generate-measure-select*

## HGAs and Graph Theory

One genre of problems that allow us to study CAESs using HGAs are graph theory problems. The undirected graph is a fundamental, widely used formalism for problem representation [Gross1999]. Moreover, cellular automata, which are based on one graph, have been successfully used to model CAESs [Fryer2004]. Our interest in expanding the consideration of graph theory to HGAs and CAESs is that many graph theory algorithms require both local and global attention. This 'zooming in' at each vertex and 'zooming out' to consider the whole graph as a structural entity is precisely the interaction of an HGA system as it vacillates between the atomic and structural levels.

## Graph Theory and Euler Tours

In this early work, we have implemented a system to generate Eulerian Graphs (i.e., graphs that have Euler tours) [Wei2011]. *Finding an Euler tour* is a problem that challenges the reader to find a trace of every edge of the graph exactly one time and then return to the starting vertex, without lifting one's pencil [Epp2011]. Euler showed that only graphs made up of even-degreed vertices[2] of at least degree two, have an Euler tour.

In our implementation, each atomic chromosome evolves a vertex. Since we know that each vertex must has even degree of at least two, our atomic fitness function can easily reward these local properties by simply adding the values of the chromosome as shown in Figure 7.

**Figure 6: Eulerian Graph evolved with the Alternating Level Algorithm**

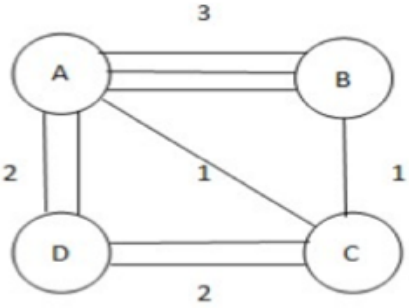


**Figure 7: Atomic Level Chromosome corresponding to vertex A and its neighborhood**

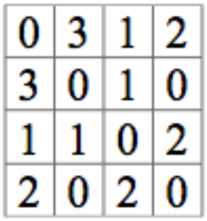
| | | | |
|---|---|---|---|
| 0 | 3 | 1 | 2 |

At the structural, global level, an Euler tour mandates that every vertex is accessible from the starting point (called "source"). In particular, the global graph in which the Euler tour exists must be

[2] An even-degreed vertex is a vertex that emanates an even number of edges. In Figure 6, vertex A has degree 6, and vertices B, C, and D have degree 4.

strongly connected; the graph is inclusively one connected component. As shown in Figure 8, the structural-level chromosome is made up atomic level chromosomes and is (emergently) affected by atomic chromosomal manipulations as well as being explicitly transformed by structural level genetic operations.

**Figure 8: Structural Level Chromosome corresponding to graph of Figure 6**



| | | | |
|---|---|---|---|
| 0 | 3 | 1 | 2 |
| 3 | 0 | 1 | 0 |
| 1 | 1 | 0 | 2 |
| 2 | 0 | 2 | 0 |

In our implementation, we used the Level Alternation Algorithm (Figure 5) and successfully evolved many Eulerian graphs. The two phases of alternation were defined by the two main properties discussed above: every vertex has even degree of at least two, and the graph is one connected component (that can be verified with the depth first search checking for accessibility of each vertex.) The structural chromosome represents the full graph and its fitness function rewards maximal accessibility. Moreover, each row of the adjacency matrix is made up of an atomic level chromosome. The atomic level fitness function rewards even degree (easily computed by summing the values of the atomic chromosome).

The program alternates between the *generate-measure-select* phase of the both levels and each evolutionary phase runs for a [3]pre-selected amount of time after which the resultant evolved information is passed to the other level for processing.

## Conclusions

We presented a taxonomy of hierarchical genetic algorithms and techniques that the author has used in recent work [Calvo2011] [Seitzer 2010] [Wei2010]. For some problems (such as generation of Eulerian graphs), it is possible to nest atomic chromosomes inside structural chromosomes, and apply the *create-measure-select* iterative GA to one, or the other, or both levels. Using these techniques, we were able to generate interesting solutions that emerged by iterating chromosomes that self-organized through the genetic operations, were interconnected by the underlying representation of the graph, and were intrinsically nested by definition. Thus, these emerging, self-organizing, connected solutions are clearly the precursors of complex-adaptive-emergent-systems.

## Future Work

Currently, our most pressing goal is to increase the magnitude of the problem sizes in Eulerian Graph generation and to expand our repertoire of graph theory problems we tackle. We are currently working on the design of a graph coloring HGA using both the Emergent HGA algorithm (as shown in Figure 4) and the Alternating Level algorithm (Figure 5).

## References

*[Ahmed2005]* Ahmed E. Elgazzar A.S, and Hegazi A.S. 2005. An Overview of Complex Adaptive Systems. Adaptation and Self-Organizing Systems, June 2005, 1.

*[Calvo2011]* Calvo, A., Seitzer, J. 2011. MELEC: Meta-Level Evolutionary Composer. Journal of Systemics, Cybernetics, and Informatics; 2011, Vol 9, No. 1, 45-50.

*[Eindhoven, 18th June 2004* http://www.fractal.org/Fractal-systems.htm

*[Epp2011]* Epp, S. Discrete Mathematics with Applications. Brooks/Cole Cengage Learning, 2011.

*[Gross1999]* Gross, J. and Yellen, J. Graph Theory and Its Applications. CRC Press 1999.

*[Fryer2004]* Fryer, P. and Ruis, J. What are Fractal Systems? (CAES), www.fractal.org/Fractal-systems.htm, June, 2004.

*[Holland1995]* Holland, J. H. *Hidden Order: How Adaptation Builds Complexity*. Addison-Wesley, Reading, MA. 1995.

*[Seitzer2010]* Seitzer, J. 2010. Hierarchical Genetic Algorithms, presentation at the 6th Annual Dayton Engineering Sciences Symposium, October, 2010.

*[Wei2011]* YuanWei and Jennifer Seitzer , 2011. Implementation of A Hierarchical Genetic Algorithm Generating an Euler Tour, Technical Report, 2011.

[3] In future work, we expect to experiment with phase time selections.