Regulatory Networks and Genomic Algorithms

Hassan Masum and Franz Oppacher
Department of Computer Science, Carleton University, Canada
{hmasum, foppache}@ccs.carleton.ca

ABSTRACT

We propose steps toward enhancing evolutionary computation via mechanisms from molecular biology, in particular the ideas of regulatory networks and embryogenesis. First we survey key facts and developments in bioinformatics and molecular genetics, followed by speculations on future implementations in evolutionary computation. We then focus on computational implementations of the mechanisms of regulatory networks and embryogenesis, surveying previous work in the field and showing the potential of these methods through analysis and analogy. Connections and applications of the regulatory network idea to other fields are also discussed.

Keywords: Evolutionary Computation, Genetic Algorithms, Regulatory Networks, Genomic Algorithms.

1. EVOLUTIONARY COMPUTATION AND MOLECULAR BIOLOGY

The Idea

Evolutionary Computation ("EC") has proven to be an effective tool for generating robust adaptive computational algorithms. However, the underlying evolutionary paradigm to date has been that of neo-Darwinian genetics; recent advances in molecular and developmental genetics have shown that evolution is a more complex process than previously suspected.

Much progress has been made in discovering and elucidating mechanisms involved in this complex process. It is therefore time to reexamine the underlying assumptions of EC, and to enhance EC with more complex, robust, and powerful mechanisms from nature.

We focus particularly on regulatory networks and embryogenesis, as these provide a mechanism that is key in developmental biology and genetics but has not been substantively used to date in EC. Reviewing the evidence suggests that regulatory networks are probably responsible for much of the algorithmic compression in biological sequences, and hence can be expected to provide effective heuristics in EC implementations.

Finally, we close with more speculative discussion on potential future work and connections with other fields. The idea of regulatory networks may be a powerful mechanism that could be of use in many complex adaptive systems.

Evolutionary Computation

EC, while tracing its roots back to isolated experiments in the 1960's on adaptive heuristics, rose to prominence as an important subfield in the 1990's. Although quite a few variations exist -- such as Genetic Algorithms, Genetic Programming, and Evolution Strategies -- they can all be considered specializations of the general EC theme. A good overview is given in the two-volume set [Back et al 2000].

The idea behind EC is to specify an algorithmic abstraction of the evolutionary process, explore the properties of this algorithm via experimentation, and then tweak the algorithm to solve problems for which it is suited. Most such algorithms use some variant of a Generate - Select - Vary loop, creating many candidate solutions and iteratively selecting the best-performing ones. A key advantage of EC is that it has been shown to perform well across a wide range of problem domains; EC thus provides a "Swiss army knife" approach, which may not be the best method for a given problem but is often simple and "good enough".

If there is already an approximation algorithm for a problem which yields great solutions in low-order polynomial time, EC methods are unnecessary and undesirable; the most suitable problems are large and complex ones where our intuition for generating even approximate solutions breaks down. For many such complex problems, EC often matches or exceeds the best known solution through other methods; the series of books on Genetic Programming by Koza has many examples. And in the long run, the apparent success of biological evolution suggests that additional layers or methods will be added to the basic "Mendelian genetic" paradigm of EC, that will allow it to scale well beyond levels achieved to date.

Molecular Biology in a Nutshell

An organism's genetic information is called a **genome**. Genomes are stored as single (haploid) or complementary double (diploid) chains of nucleotides; each nucleotide has one of four possible bases attached to it, and hence genome length can be measured in **base pairs (bp)**. Genomes use a **"universal genetic code"**; this mapping from the 64 possible nucleotide triplets ("codons") to the 20 amino acids (plus a stop symbol to demarcate the end of genes) is the same across

almost all species. This codon to amino acid mapping has intermediary RNAs: mRNA transcribes DNA codons and travels to ribosomes, while tRNA translates and rRNA facilitates the actual protein construction in ribosomes.

Where codon synonyms exist, there may be differential preferences between organisms for the codons used for a given amino acid ("codon bias"). There is a moderate correlation between the number of codon synonyms and the frequency of that amino acid in the organism's **proteome**. (An organism's proteome is the set of proteins expressed from its genome.) It has been speculated that the code's error-correcting structure minimizes the deleterious effect of mutations.

An extra layer of DNA coding comes from **DNA methylation**; some organisms (particularly mammals and other vertebrates) have an extra methyl group attached to certain nucleotides, which acts as a functional tag. DNA methylation is thought to repress genes by locally inhibiting DNA transcription; in mammals it differentiates genes identical in base-pair sequence by which parent they are from, and in bacteria it differentiates methylated host DNA from unmethylated phage or plasmid DNA.

"Action-at-a-distance" occurs due to geometric and physical properties of DNA folding; there is also a distinction between "upstream" and "downstream" regulation, since RNA transcription is directional along each DNA strand. Short DNA sequences called promoters and enhancers tell the RNA polymerase where on the DNA to start transcribing; promoters are typically slightly upstream of the gene coding region, while enhancers can be far away in linear base sequence distance (but perhaps close in real space).

Specific base pair sequences can have different functional meanings, and particular combinations of genes may have unusually high or low probabilities of recombination, suggesting the richness of **gene sequence grammar**. This ties into **computational molecular biology algorithms**; large-scale genome composition must be inferred from error-prone observation of small DNA fragments. [Setubal & Meidanis 1997] discuss several other categories of algorithms: sequence similarity searching, phylogenetic tree reconstruction, genome rearrangement modeling, and protein structure and function prediction from amino acid sequence data. Common ranges of protein-coding genes are 500 to 5000 for bacteria, and 10 000 to 50 000 for animals (including humans).

Introns are the sequences of DNA which are "spliced out" of the primary mRNA after initial transcription. In many higher organisms (e.g. humans) the intron portions of the genome are much larger than the expressed exon portions. Individual exons often code for complete functions or modules, in the protein sense; different types of cells then regulate RNA splicing to combine different subsets of the exon "function library", in a process known as "alternative splicing".

Bacteria have few or no introns, probably since their survival is largely determined by reproductive speed. (This suggests an analogy to **optimization** of computer programs where execution speed is essential. Higher organisms, with more introns and greater modularity of DNA, would be analogous to the modularity and sanity checks of large programs, where ease of programming, comprehensibility, maintainability, and robustness become dominant concerns.) There is a controversy in the molecular biology world about "introns first" or "introns later" in bacterial evolution: were bacteria modularly designed and then optimized, or was spaghetti code followed by modularization and introns to allow for the modular development of higher organisms?

Viruses are as small as 10 genes, although they are parasitic code fragments which cannot reproduce independently; the lower limit on bacteria is a few hundred genes. With the greater selection pressure for **efficient coding**, bacteria sometimes use bidirectionality and frameshifting for efficient compression. In contrast, larger organisms are so profligate that much of their genetic material is often structural.

Mutations seem to happen with roughly the same low probability throughout the DNA sequence, although repair probabilities may differ (though they are usually well over 99%). There are cases where a harsh environment triggers a stress response in organisms, increasing mutation probabilities; in other instances mutation probabilities are actually decreased upon radiation exposure, as more robust DNA repair mechanisms are triggered. There is also controversial evidence for advantageous mutations occurring more often under specific circumstances, as well as for reverse RNA transcription as a method of incorporating phenotype changes back into the genotype. Many different kinds of mutations exist: deletion, duplication, inversion, insertion, transposition, translocation (where gene position is changed but functionality is mostly conserved), and point mutation.

Macro-mutations are selected against at the phenotypic level, since they often lead to deleterious effects. However, mutations in regulatory genes allow macro-mutations of higher expected survival value to occur. For instance, if a regulatory gene controls the size of some body part, a simple change in the gene (or in the concentration of gene-activating chemicals in the local environment) could cause a drastic change in body part size; combine several such changes, and one can see how very different phenotypes can arise from similar genotypes. Therefore, regulatory genes give a feasible mechanism for large phenotypic and morphological changes from small genotypic changes.

According to the **Central Dogma** of molecular genetics, information flow goes from DNA $\rightarrow$ RNA $\rightarrow$ Protein, with flow within state also possible, but reverse flow not possible. However, this has recently been contradicted by the examples of RNA viruses (RNA $\rightarrow$ DNA) and arguably prions (infectious protein $\rightarrow$ protein shape changes); in addition, RNA $\rightarrow$ DNA can happen with the help of the "reverse transcriptase" enzyme, which can be used to get an intron-free

version of DNA. Finally, there exist RNA that can self-reproduce in the absence of protein. Of the three basic building blocks (DNA, RNA, protein), RNA is likely the one in which self-reproduction originally arose.

The **DNA → protein process** in a nutshell:

1. DNA in the nucleus gets transcribed into
2. primary transcribed RNA, which goes through RNA processing control to make
3. mRNA, which exits the nucleus.

Now the mRNA has to pass through the nuclear envelope to get outside the nucleus and into the cytosol of the cell-at-large. The exit takes place through the nuclear pore, a complex system containing ~200 different proteins; mRNA or other substances passing through need a biochemical ticket to pass. (Prokaryotes have no nucleus and so skip this step; this suggests the importance of the nucleus and other complex organelles that distinguish eukaryotes.)

4. mRNA outside the nucleus goes to ribosomes, where rRNA and tRNA translate it to
5. protein (which can be structural or an enzyme).

Several hundred bases at a time are unzipped during RNA transcription, and then rezipped. Some DNA loops may break off and be spliced elsewhere in their entirety; this is related to translation-type mutations. Observations of repressor proteins in bacteriophages suggest that, during translation, the protein apparatus that reads the DNA may be sensitive to the local 3D configuration of the DNA; this would imply an extra layer of DNA encoding.

Since mRNA is single-stranded, it doesn't have the error-correction of DNA base pair complementarity. Anticodons of tRNA take amino acids from the cytoplasm and make proteins in **ribosomes**, which are protein factories. Humans have at least ~10^5 types of proteins.

Proteins are usually enzymes (which catalyze specific chemical reactions) or structural. They have structure on four different spatial scales; secondary and tertiary structure (3D layout) largely determines function, and changes much more slowly than primary sequence structure (where chemically similar amino acids can be substituted without changing protein function). Many proteins are composed of semi-independent modular domains, which occur independently in different proteins; tertiary structure similarly has recurring topological motifs. Proteins can undergo dynamic changes of conformation which are integral to their functionality.

In true distributed computation fashion, the operation (and even formation) of proteins is a stochastic diffusion process, where hundreds or thousands of different kinds of interactions are taking place simultaneously in the same relatively undifferentiated physical space. Coherence is maintained in this mess with the aid of **molecular specificity**: proteins adopt different 3D configurations, and protein-protein interaction only takes place if the configurations fit like jigsaw pieces. (This is largely because the attractive van der Waals forces are strong only at distances between 3 and 4 angstroms, and so large portions of both molecular surfaces must be that distance apart.) Therefore, proteins are like coded messages sharing the same channel; this "information flow" view of molecular activity is discussed further in [Loewenstein 1999].

Genomic Algorithms

In [Masum et al 2000], we proposed many new mechanisms that could be computationally implemented:

DNA is a multilayered protocol. DNA sequences are not the sole input variable in the (dynamic and recurrent) genotype-phenotype mapping; as discussed in [Kanehisa 2000], this "genome as source code" idea is too simplistic. Many other layers are present including DNA methylation and DNA spatial arrangement, as well as the gene expression layers discussed later. It might therefore be useful to experiment with multilayer genomes, with each layer encoding different phenotype properties; this could be a natural implementation method for multiobjective EC. Subsumption architectures provide a model for this sort of multilayer programmed behavior.

Mitochondrial and chloroplast DNA is separate from nucleic DNA. Similarly, bacteria contain extrachromosomal DNA in the form of plasmids and phages. Such **endosymbiosis** may be useful for quick adaptation or long-term preservation of crucial core functions; for example, the influenza virus frequently recombines (shuffling genes) and thus eludes adaptation by immune systems and vaccines. The fact that plasmids are present only in bacteria suggests their suitability for small-scale computational genomes, where swapping functions or immunities has less probability of damaging the overall structure. Computational immune systems as overviewed in [Dasgupta 1998] are a good testbed for such ideas in distributed EC.

Biological genomes appear to have **differential probabilities of mutation and crossover** for different areas. Can these probabilities be affected by phenotypic environment cues, and do mutation operators coevolve with their genomes? This line of thought suggests adaptive mutation probabilities in different parts of computational genomes, to optimize resistance to change in parts that have proven effective or that must undergo a large mutation in genotype space to reach another region of reasonable fitness; see e.g. [Vekaria & Clack 1999]. Estimation of distribution algorithms are surveyed in [Pelikan et al 1999]; these work by sampling and estimating properties of the population distribution, and using these inferred properties to generate new population members.

The relatively simple genomes of nanobacteria and viruses are good starting points to **decompile genomes**; which will likely yield new algorithms and distributed coordination methods.

Fully understanding the genome will require a complex holistic approach with lots of tangled semantic webs, as is put forth in [Chicurel 1999].

Phylogenetic trees reconstruct development through evolutionary time; when combined with genomic analysis, they can highlight the innovations leading to jumps in biological fitness. This could in turn suggest design strategies for scaling up EC. Phylogenetic trees and other genome analysis algorithms might also be adapted to pick out key lines of development in an evolving population of computational genomes.

Analogously to the Human Genome Project, a **Computational Genome Project** is an intriguing possibility. Of course, the space of extant computer programs seems more heterogeneous than that of biological genomes; however, a wide-scale survey at the object or component level could yield useful software and human use metrics, which could in turn suggest heuristic and statistical methods for GP. Given such metrics and a **"semantic hash function"** that could functionally differentiate program modules with reasonable accuracy, one could then search for "similar functions", allowing GP (and human programmers) to use previously-utilized methods from a persistent library.

Molecular specificity lets genes evolve relatively independently; changes in the binding properties and expressed amounts of one protein will usually only affect a relatively specific metabolic function. Because protein products often interact in cascading networks (with the local concentration of products activating or inhibiting expression of another gene), there is an analogy with **stigmergic methods** which offload computational memory to the environment; a fusion of EC with the swarm intelligence methods in [Bonabeau et al 1999] may therefore prove fruitful.

Bacteria have fluid genomes, with plasmid swapping and high RNA mutation rates; in contrast, higher organisms change more slowly, hence we can build phylogenetic trees going back millions of years for higher eukaryotes. (In fact, the commonality of e.g. homeobox genes indicates that higher organisms may be formed from relatively few ancestor genes.) Searching for genes that are homologous across a wide variety of species will give us proteins and regulatory functions of universal value, which may be worthwhile to implement. This strongly suggests that a similar **hierarchy of strategy complexity** may be the best way to proceed in EC, with the one-off optimization problems using simple data structures evolved from scratch each time, and more complex large persistent problem domains utilizing a library of tried and tested subroutines. After all, if small and fast was always best then microbes would rule the world.

Building a **taxonomy space** of genomic algorithm models will help in visualizing the design space, understanding the properties of existing algorithms, and finding underexplored areas for new algorithms. Many formal measures of complexity have been devised in biology, physics, and computer science, as surveyed in [Badii & Politi 1999]. Using these and other sources, taxonomy space complexity measures should include:

- temporal and spatial differentiation,
- degree of depth, parallelism and recurrence of gene expression networks (e.g. [Nehaniv 1999])
- extragenetic environment usage,
- number and correlation of heredity methods,
- ratio of number of cell types to total number of cells (and related measures in [Bonner 1988]).

Analysis of biological genomes will also suggest features to distinguish computational organisms:

- heterogeneity of fitness function (e.g. molds where all cells have the same environment vs humans),
- prokaryote / eukaryote (e.g. amount of non-genetic material and protection of genetic material),
- entropy, intron to exon ratio, n-gram frequency counts, and other statistical metrics,
- Kolmogorov and software complexity, and other semantic complexity measures.

Taxonomies could also be generated for proteomes, functional genomics, and metabolic pathways.

When moving from theory to model-building, one clearly cannot duplicate the physical and temporal resources of Earth's biosphere: $O(10^{40})$ molecules interacting for $O(10^{17})$ seconds. The hope, of course, is we can implement an approximating subset of algorithms from molecular genetics into our EC methods with modest increase in time complexity, but large increase in problem-solving effectiveness.

Molecular biology suggests many interesting mechanisms which could be applied computationally. Some of the key areas where such ideas could improve EC are:

- **Scalability**, through modularity, hierarchy, large-scale coherence, and other ways of moving up from our current "primordial slime" stage;
- **Genome structure**, differentiating GP space through studying the "algorithmic topology" and implemented algorithms of biological genomes;
- **Adaptation**, via morphogenetic models, environmentally-sensitive gene expression, functional specialization and networks, and protein analogues.

As [Wooley 1999] emphasizes, the next challenge of genetics is to move from sequence-level genomics (e.g. the Human Genome Project) to structural and functional genomics. Hopefully, great strides will be made in holistic understanding of genetic and protein functionality, leading in turn to many new ideas for computational algorithms.

Clearly this is **fertile ground** for future work. Let's now focus

on one of these areas: regulatory networks and embryogenesis.

2. REGULATORY NETWORKS

Biological Background

Regulatory genes control the expression of other genes (or co-ordinated groups of genes called operons) according to temporal, positional, and environmental cues. These genes can functionally interconnect in very complex networks. A simple example is the lac operon in E. coli:

- Lactose absent → repressor protein binds to operator site → gene transcription stopped.
- Lactose present → it binds to repressor site → no repressor produced → gene transcription happens.

Thus the gene's expression is dependent on local environmental conditions. The gene being transcribed could also be a transcription factor for other genes.

Currently DNA microarrays test an organism for expression of ~10^4 gene sequences; this process can be done periodically to the same organism, with or without external inputs, generating large data sets on activatory and inhibitory relationships between gene products. **Inferring the regulatory gene network** which causes a given set of correlations is a challenging problem: [Chen et al 1999] analyzes a graph model, giving approximation algorithms and showing that even in restricted cases the problem is NP-complete; [Karp et al 1999] discusses picking a minimal set of differential gene expression experiments to distinguish between possible biological pathways, with pathways represented by feedback-free boolean circuits; [Gilbert et al 2000] uses GP to find functional classes for genes.

Differential gene expression is ubiquitous in organisms. The transcription and translation process is happening simultaneously for all genes expressed by a particular cell; however, the rate of protein formation varies widely by gene. This rate depends on such factors as the presence of required transcription factors in the local environment, DNA methylation and current conformation, mRNA concentration and longevity, progress through the cell life cycle, reaction to external stimuli, and of course cell type. As an analogy, the collection of genes in a human cell could be thought of as similar to a large room with 30 000 pianists, each with a different musical score. [D'haeseleer et al 2000] review approaches to understanding the underlying gene networks, and discuss implications of successful analysis.

Bone, pancreas, brain, and other cells are all made from the same gene set; differential reading and expression due to embryogenetic and physical factors causes different cell types. During **embryogenesis**, once the initial cell has doubled 2 or 3 times cells will be "inside" or "outside", which along with gradients of embryonic chemicals called morphogens helps provides the "symmetry breaking" necessary to start cell differentiation. A simplistic view of cell differentiation is to imagine a tree of possibilities, with the presence or absence (or more generally concentration) of a regulatory protein or environmental factor increasing the number of possible phenotypes at each level.

Cells intercommunicate and respond to conditions in their local environment through **cell signaling**, in which information travels across the external plasma membrane to the cell interior (often to the nucleus). Stimuli-specific receptors on the plasma membrane surface transmit a signal to effector molecules within the cell, causing changes in gene expression, enzyme activity, cytoskeleton configuration, ion permeability, and DNA synthesis. Sometimes even programmed cell death occurs, as with excess nerve cells or embryonic tissue, and potential cancer cells. Multiple cell signaling pathways are often interconnected; they can be convergent, divergent, or networked in more complex ways.

Stem cells are the "biological seeds" out of which different organs and tissues form; the same stem cell can develop into very different macrostructures depending on local conditions, as reviewed in [Fuchs 2000]. They are related to homeobox genes, which are a small set of expressed genes that control cell differentiation and body part expression and are highly conserved across many widely divergent species (e.g. mice, flies, and humans). We can similarly think of "computational seed structures" that can, depending on their local physical or computational environment, develop in different ways into computational macrostructures.

The power and beauty of regulatory networks in biology is well explained in [Coen 1999]. Both the developmental process of embryogenesis and adaptation to changing environments depend crucially on the complex recursive network formed by gene regulatory networks. Coen shows how this process can be understood via analogy to **"hidden colors"** in an artistic canvas; different hidden colors correspond to different gene products, and the regulatory process as a whole creates a complex differentiated field of many such colors throughout an organism, each of which then has some visible biological effects.

Computational Implementations

In [Liang et al 1998] **gene expression networks** are modeled as boolean networks with synchronous state evolution. The authors report that their scheme for inducing models from observed data works surprisingly well. This type of non-spatial model abstracts the embryogenetic process into temporal evolution of a boolean-valued vector. A related paper, [Wuensche 1998], analyzes the dynamics of regulatory networks when modeled as a discrete dynamical feedback network.

A gene regulation model abstracted from *Drosophila* is presented in [Luke et al 1999], and used to successfully evolve deterministic finite automata. A dynamic model is analyzed in [Reil 1999]. Finally, simulation methods for prokaryotic gene

circuits are reviewed in [McAdams & Arkin 1998]; most simulations done thus far use either systems of differential equations (for small networks) or boolean circuit models (for larger networks). The goal is to understand the design rules for these regulatory circuits, so that they can be modified for altered functionality and evolved more efficiently. Having a computational analogue of transcriptome and proteome time series analysis will provide a visualization of different levels of phenotypic change and evolution.

In embryogenesis, several factors aside from "inside / outside" differentiate cells, e.g. internal cell counters, local chemical gradients, and boundary layer detection. [Hamahashi & Kitano 1998] discuss a relevant *Drosophila* computational model. How can the whole developmental process be so robust? Protists (single-celled eukaryotes) are self-contained organisms and more complex than most cells of multicellular creatures, but the latter achieve greater overall complexity through **cell differentiation**, specialization, and co-ordination. This suggests a metaphor for computational scaling, since no complex non-differentiating organisms seem to exist.

There is ongoing work with "embryonic systems" that looks at using embryogenetic models and regulatory networks to provide a flexible developmental structure for evolvable hardware and software; two recent proceedings from conferences that focus on these themes are [Miller et al 2000] and [Lohn et al 2001]. Other current work in the field can be found in the *Pacific Symposium on Biocomputing* and *RECOMB* conference proceedings.

3. ANALYSIS AND USE OF REGULATORY NETWORKS

A Simple Cell Model

To illustrate the type of model that could productively be analyzed, let's consider a cell as containing P different gene products, with $\mathbf{c} = (c_1, c_2, \ldots c_P)$ being the vector of concentrations of these products at some point in time. If we want to discuss a time-evolving series of concentration vectors, we can denote the vector at time t by $\mathbf{c}^t = (c_1^t, c_2^t, \ldots c_P^t)$. (Note that there is no spatial dimension to this model -- this is biologically reasonable as a first approximation, since diffusion rates are quick enough that gene products can rapidly disperse throughout the cell.)

Each product c_k^t depends on the concentrations of at most N products at the previous time step, with these products indexed by the matrix **PREV[k,x]**. This dependence could be as simple as a Boolean function, or it could be a more complex function of the relevant N products of the previous time step:

c_k^t = **DependenceFn (**previous products at time t-1)
= **DependenceFn** $(c_{\mathbf{PREV[k,1]}}^{t-1}, c_{\mathbf{PREV[k,2]}}^{t-1}, \ldots c_{\mathbf{PREV[k,N]}}^{t-1})$.

We can discretize the concentrations possible into D intervals; in many models, D is often set equal to 2. Since there are P different gene products, this gives a total of D^P possible states that a cell can be in at a single time; denote the set of states by S. (Note that |S| defines the maximum "cycle time" before a single cell repeats a previous state and starts looping. It is an interesting question to analyze the average instead of maximum cycle times; we expect the average cycle time to be much less.)

Since our cells are defined both by their static set of gene product concentrations and by their network of recurrent links, we need to include the latter to fully specify a cell. In the most general case, we could specify a mapping $F: S \rightarrow S$ that specifies the successor state for any given state. The set of such mappings of a set to itself has cardinality $|S|^{|S|}$ if we allow states to map to the same state (since there are |S| choices for each state to map to), and the lesser cardinality |S|! if states cannot map to the same state (since each permutation defines a qualifying mapping).

In our model, however, the number of possible cells will be substantially less, since we are restricting each c_k to depend on only $N < P$ other c's. (The simplest case is where N = 1, and each c_k depends on only D possible "states of the past" in the appropriate c_j which maps to it.)

The process can be viewed as a set of P nodes with recurrent links from the nodes feeding back into each node. (By our simplifying assumption, each node will have at most N incoming links.) Note the similarity of this model to Boolean networks as discussed in [Wuensche 1998], recurrent neural networks, the evolution of 1D Cellular Automata, and feedback registers.

Cell-Environment Interaction

How could this process be linked to an external environment? Let's suppose that each positive c_k has some effect in its cell environment (in biological situations these effects would typically be structural or enzymatic). The set of external states **e** will each be affected by up to M of the c's; increasing M from 1 upward defines an "independence spectrum" on the environments. One could explore the situation where the e's are 1st, 2nd, ... order Markov, and the cell gets some reward based on how well it performs against the environment; note that this mathematically tractable but unrealistic case assumes the cell doesn't affect its external environment.

Each c could evolve semi-independently, to be better or worse at different tasks. An initial non-zero c_k can be formed from i) reproduction / embryogenesis, ii) external environment feedback, iii) sensor / temporal / inter-cell stimulus, iv) "cognitive" stimulus. The recurrent links between c's should also evolve, causing activation and inhibition of other c's.

Given this environmental model, it's of interest to compare the performance of cells with regulation vs cells without regulation under simple assumptions. This would give an idea of the potential benefit to be gained from adding the regulatory network layer. As a start in this direction, a mathematical

model could be specified to get an idea of key parameters such as:

- # of different cell states that can be encoded by a cell.
- # of environmental states that can be responded to by a cell.

One can divide the P gene products into:

- ActionProducts that cause some effect in the cell environment and may affect further gene activity, and
- SignalProducts that only affect further gene activity.

(with |{ActionProducts}| + |{SignalProducts}| = P).

Both ActionProducts and SignalProducts can have their concentration changed by the concentrations of up to N other Products. If N = 0, then each cell is simply fixed; every cell will, once created, make a constant amount of each Product. There are two main cases where use can be made of N > 0: embryogenesis and environmental adaptation.

Embyrogenesis is the process of organism growth, in which an initial "seed" replicates and differentiates into a coherent multicelled creature. The key issue is to find a mechanism which lets a single cell with a single "program" replicate to billions of differentiated cells in an organized and environmentally-sensitive way. The "environment" is formed by internal cell environments, and resource constraints from external environment. In this case, the SignalProducts are the enabling mechanism; a complex set of overlapping SignalProduct gradients is formed across the set of cells, with local values in each cell encoding its position, function, and growth stage. This gradient-encoded information in turn triggers appropriate ActionProduct formation in the cell.

In contrast, environmental adaptation usually happens by the changing production of ActionProducts in a mature organism, in response to a changing external environment (or the passage of time). This is done by the infusion of ActionProducts into the cell (either directly or through some form of receptor), which in turn triggers or inhibits other ActionProducts. So the two cases differ in the relative importance of ActionProducts and SignalProducts, while being similar in their use of a recurrent cascading network of Product concentrations, with the behavior of the recurrent network determined by the Product sensitivities of each ActionProduct and SignalProduct.

Advantages of Regulatory Models

After understanding the single-cell case, the next step would be to examine the possible gain of using multiple cells. Different cells are often in differing local environments, and communicate with each other. This is a key advantage of regulatory networks: the same cell can express different properties in differing environments, providing a kind of compression of many behavioral repertoires into a single underlying program.

There is an interesting connection with the Busy Beaver functions discussed in [Brady 1988], which are maximally compressed Turing Machines. Considered as abstract machines, cells with regulatory networks are similar to Busy Beavers in that they achieve high behavior compression through writing to a shared environment that a fixed program keeps reading, but such cells are also adaptive and operate in parallel.

We expect regulatory networks to provide a compact encoding of responses to many external environment situations. With P products and D concentration levels, there are D^P possible gene product environments in each cell, and multiple cells will have characteristic distribution functions across these internal environments (depending on the local external environment).

With embryogenetic mechanisms, regulatory networks provide a way to locally differentiate phenotypes from the same underlying genotype, in order to deal with variations in local conditions. Finally, they provide a simple distributed co-ordination method -- the same overall program is being run in different ways, and since only one program is evolving, it will implicitly evolve to perform well when instantiated in multiple co-operating cells.

4. CONCLUSION AND REFERENCES

Conclusion

We have reviewed basic principles of molecular genetics, and pointed out many ways in which more realistic molecular mechanisms are being incorporated into EC, along with some ways that have not yet been tried. We then focused on the mechanism of regulatory networks, arguing that they are a general principle of much promise for EC in particular and complex adaptive systems in general. Connections to current work in modeling and inferring regulatory gene networks were made, and a simple abstract model described. Regulatory networks are clearly a powerful and promising control heuristic.

References

[Back et al 2000] T Back, DB Fogel, and Z Michalewicz (Editors). *Evolutionary Computation (Volumes 1 and 2).* Institute of Physics Publishers, 2000.

[Badii & Politi 1999] R Badii and A Politi. *Complexity: Hierarchical Structures and Scaling in Physics*. Cambridge University Press, 1999.

[Bonabeau et al 1999] E Bonabeau, M Dorigo, and G Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, 1999.

[Bonner 1988] JT Bonner. *The Evolution of Complexity by Means of Natural Selection*. Princeton University Press, 1988.

[Brady 1988] AH Brady. The busy beaver game and the meaning of life. In *The universal Turing machine: a half-century survey*, R Herken (Ed), pp 259-277; Oxford University Press, 1988.

[Brown 1999] TA Brown. *Genomes*. Wiley-Liss, 1999.

[Calabretta et al 1998] R Calabretta, S Nolfi, D Parisi, and GP Wagner. A Case Study of the Evolution of Modularity. In *Artificial Life VI*, pp 275-284. MIT Press, 1998.

[Chen & Wood 2000] J Chen and DH Wood. Computation with biomolecules. In *Proceedings of the National Academy of Sciences (USA)*, Vol.97 No.4 (Feb 15 2000), pp 1328-1330.

[Chen et al 1999] T Chen, V Filkov, and SS Skiena. Identifying Gene Regulatory Networks from Experimental Data. In *RECOMB99*:, pp 94-103. ACM, 1999.

[Chicurel 1999] M Chicurel. The Bigger Picture. In *New Scientist*, Vol.164 No.2216 (Dec 11 1999), pp 38-42.

[Coen 1999] E Coen. *The Art of Genes: How Organisms Make Themselves*. Oxford University Press, 1999.

[D'haeseleer et al 2000] P D'haeseleer, S Liang, and R Somogyi. Genetic Network Inference: From Co-Expression Clustering to Reverse Engineering. In *Bioinformatics*, Vol.16 No.8 (2000), pp 707-726.

[Dale 1998] J Dale. *Molecular Genetics of Bacteria (3rd ed)*. Wiley, 1998.

[Dasgupta 1998] D Dasgupta (Ed). *Artificial Immune Systems and Their Applications.* Springer, 1998.

[Fuchs 2000] E Fuchs and JA Segre. Stem Cells: A New Lease on Life. In *Cell*, Vol.100 No.1 (Jan 7 2000), pp 143-155.

[Gilbert et al 2000] RJ Gilbert, JJ Rowland, and DB Kell. Genomic computing: explanatory modeling for functional genomics. In *GECCO-2000*, pp 551-557. Morgan Kaufmann, 2000.

[Hamahashi & Kitano 1998] S Hamahashi and H Kitano. Simulation of *Drosophila* Embryogenesis. In *Artificial Life VI*, pp 151-160. MIT Press, 1998.

[Hogeweg 2000] P Hogeweg. Shapes in the Shadow: Evolutionary Dynamics of Morphogenesis. In *Artificial Life*, Vol.6 No.1 (Winter 2000), pp 85-101.

[Kanehisa 2000] Minoru Kanehisa. *Post-Genome Informatics*. Oxford University Press, 2000.

[Karp 1999] G Karp. *Cell and Molecular Biology: Concepts and Experiments (2nd ed)*. Wiley, 1999.

[Karp et al 1999] RM Karp, R Stoughton, and KY Yeung. Algorithms for Choosing Differential Gene Expression Experiments. In *RECOMB99*, pp 208-217. ACM, 1999.

[Levenick 1999] JR Levenick. Swappers: Introns promote flexibility, diversity and invention. In *GECCO-99*, pp 361-368. Morgan Kaufmann, 1999.

[Liang et al 1998] S Liang, S Fuhrman, and R Somogyi. REVEAL, A General Reverse Engineering Algorithm for Inference of Genetic Network Architectures. In *Biocomputing '98*, pp 18-29. World Scientific,1998

[Loewenstein 1999] WR Loewenstein. *The Touchstone of Life: Molecular Information, Cell Communication, and the Foundations of Life*. Oxford University Press, 1999.

[Lohn et al 2001] J Lohn, A Stoica, and D Keymeulen (Editors). *The Second NASA/DOD Workshop on Evolvable Hardware*. IEEE, 2001.

[Luke et al 1999] S Luke, S Hamahashi, and H Kitano. "Genetic" Programming. In *GECCO-99*, pp 1098 - 1105. Morgan Kaufmann, 1999.

[Masum et al 2000] H Masum, F Oppacher, and G Carmody. Genomic Algorithms: Metaphors from Molecular Genetics. In Workshop Proceedings of *GECCO-2000*, pp 173-178.

[McAdams & Arkin 1998] HH McAdams and A Arkin. Simulation of Prokaryotic Genetic Circuits. In *Annual Review of Biophysics and Biomolecular Structure,* Vol.27 (1998), pp 199 - 224.

[Miller et al 2000] J Miller, A Thompson, P Thompson, and TC Fogarty (Editors). *Evolvable Systems: From Biology to Hardware (Third International Conference, ICES 2000).* Springer-Verlag, 2000.

[Nehaniv 1999] CL Nehaniv (Ed). *Mathematical and Computational Biology*. AMS, 1999.

[Oliver 2000] S Oliver. Proteomics: Guilt-by-association goes global. In *Nature*, Vol.403 No.6770 (Feb 10 2000), pp 601-603.

[Pelikan et al 1999] M Pelikan, DE Goldberg, and F Lobo. A Survey of Optimization by Building and Using Probabilistic Models. IlliGAL Report No.99018; UIUC, 1999.

[Reil 1999] T Reil. Dynamics of Gene Expression in an Artificial Genome - Implications for Biological and Artificial Ontogeny. In *Advances in Artificial Life (ECAL 99)*. Springer, 1999.

[Setubal & Meidanis 1997] J Setubal and J Meidanis. *Intro. to Computational Molecular Biology*. PWS, 1997.

[Vekaria & Clack 1999] K Vekaria and C Clack. Biases introduced by adaptive recombination operators. In *GECCO-99*, pp 670-677. Morgan Kaufmann, 1999.

[Wooley 1999] JC Wooley. Trends in Computational Biology. In *J. Computational Biology*, Vol.6 No.3-4, pp 459-474.

[Wuensche 1998] A Wuensche. Genomic Regulation Modeled as a Network with Basins of Attraction. In *Proceedings of the 1998 Pacific Symposium on Biocomputing*, pp 89-102.