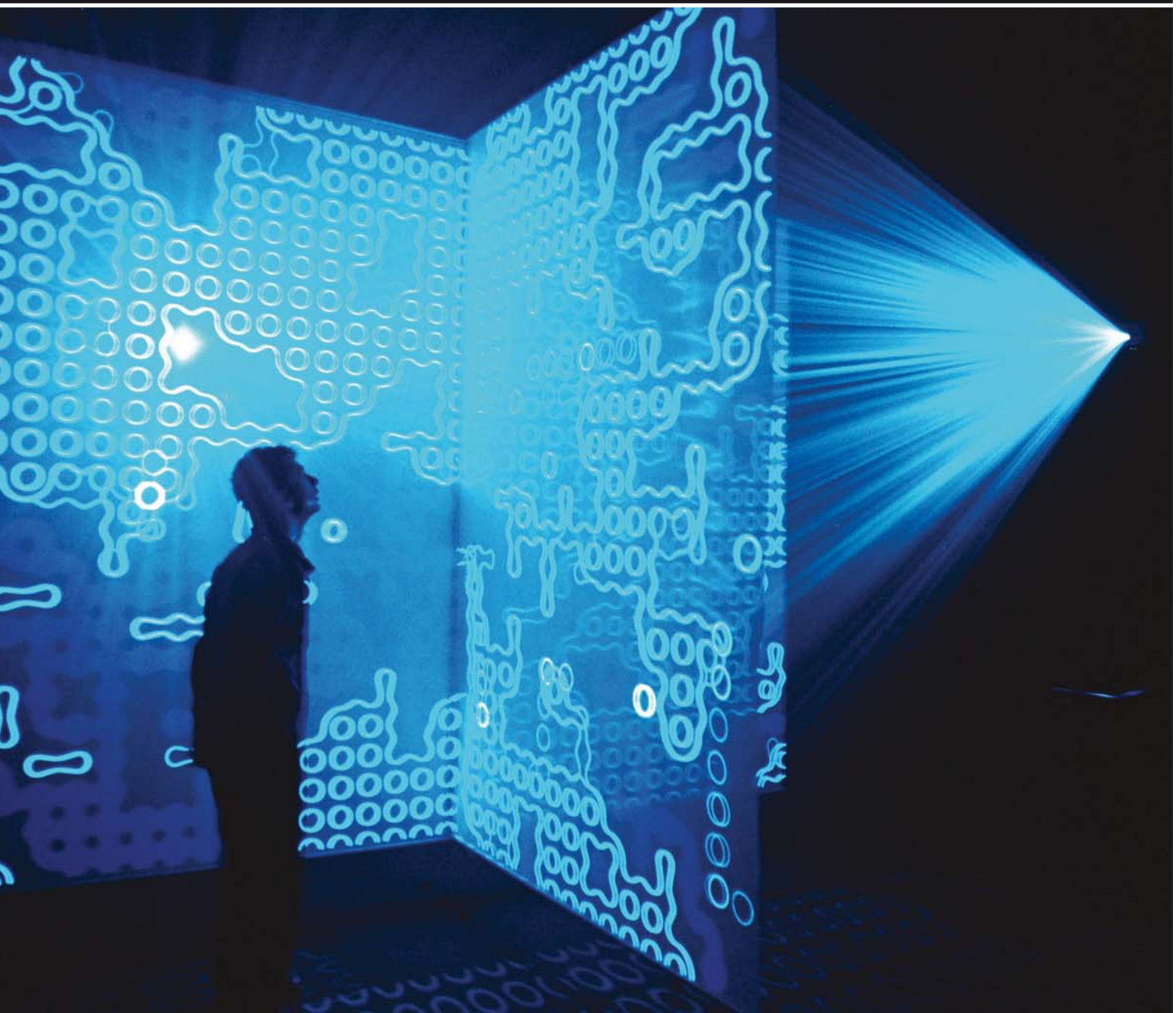
SIGEVOlution

newsletter of the ACM Special Interest Group on Genetic and Evolutionary Computation

April 2006
Volume 1 Issue 1



in this issue

Editorial

Welcome to the first issue of SIGEVOlution, the newsletter of the ACM Special Interest Group on Genetic and Evolutionary Computation (SIGEVO). One year has almost passed since this newsletter was announced during the last GECCO in Washington D.C. and now, while many of us have already booked an airplane ticket to Seattle, the first issue is ready, at last! SIGEVOlution is first of all an opportunity. It has been conceived as a way to facilitate the sharing of information relevant to the EC community. In particular, the information that would not fit in mainstream EC scientific journals, such as, *Evolutionary Computation*, *Genetic Programming and Evolvable Machines* (GPEM), or the *Transactions on Evolutionary Computation*. Ideally, each issue of SIGEVOlution should include a couple of short, general interest articles from EC related fields; a review or a tutorial of available EC software; several columns about various topics, such as, surveys of existing EC research groups and labs, reports of EC conferences or workshops, letters, recently discussed theses, forthcoming papers, books, and events.

This first, inaugural issue gives a bird's eye view of the many opportunities that SIGEVOlution can offer. In the first paper, Charles Neely Harper reports on two applications of evolutionary computation developed at American Air Liquide. In his position paper, Jon McCormack discusses the new challenges in the field of evolutionary art. He also provided the picture for the cover of this issue. In the software corner, Christian Gagné and Marc Parizeau introduce us to *Open BEAGLE*, a C++ EC framework which supports major evolutionary algorithms, including Genetic Algorithms and tree-based Genetic Programming. The subsequent columns provide various information about a newly born EC lab, a recently discussed PhD thesis, the forthcoming issues of EC journals, new books, and the calendar of EC events.

SIGEVOlution has been possible thanks to the help of many people who supported this project in several ways. The members of the SIGEVO board, who entrusted me to be the editor of this wonderful project. Gianluca Pignalberi who gave me the first set of LaTeX classes from which this newsletter has been created and was patient enough to help me at various points. The members of SIGEVOlution board, Dave Davis and Martin Pelikan. Last but not least, the authors, Charles Neely Harper, Jon McCormack, Christian Gagné, and Marc Parizeau, who provided the contents for this issue without having a clue how their contributions would appear.

Pier Luca
April 30th, 2006

SIGEVOlution
April 2006, Volume 1, Issue 1

Newsletter of the ACM Special Interest Group on Genetic and Evolutionary Computation.

Contributors to this Issue

Charles Neely Harper
Jon McCormack
Christian Gagné
Marc Parizeau

Contents

ISSN: 1931-8499

Evolutionary Computation at American Air Liquide

Charles Neely Harper
Director, National Supply & Pipeline Operations
Air Liquide Large Industries U.S. LP

Air Liquide is the world leader in industrial and medical gases and related services. The Group offers innovative solutions based on constantly enhanced technologies. These solutions, which are consistent with Air Liquide's commitment to sustainable development, help to protect life and enable our customers to manufacture many indispensable everyday products. Founded in 1902, Air Liquide has nearly 36,000 employees and is present in more than 70 countries. Sales in 2005 totalled 10,435 million euros.

American Air Liquide Holdings, Inc. oversees the North American operations of Air Liquide. Through its subsidiary businesses, American Air Liquide offers industrial gases and related services to a variety of customers including those in refining, natural gas, chemistry, metals, automotive, chemicals, food, pharmaceutical, electronics, specialty and healthcare markets.

Our products are primarily oxygen, nitrogen, and hydrogen along with the services and technology involved in delivering these gases. We separate atmospheric air into oxygen, nitrogen and argon through a cryogenic distillation process and we produce hydrogen by cracking natural gas. We distribute our products through several methods: in gaseous form through nearly 2,000 miles of pipelines or in compressed cylinders and in liquid form, by truck transportation from our plants to our customers' tanks and facilities. More than half the cost of creating and distributing oxygen, nitrogen and hydrogen lies in the cost of energy, as natural gas or electricity. Operating air separation and hydrogen plants, cogeneration units and our pipeline is an energy-intensive business.

In 1999 we began to investigate ways to substantially reduce our production and distribution costs and to find "smart" ways to manage our supply chain. We hired BiosGroup, a complexity science company based in Santa Fe, NM, to help us assess the potential for cost reduction. A result of that engagement was the decision to pursue two separate streams of optimization: one related to reducing the cost of producing and distributing liquid oxygen, liquid nitrogen and liquid argon, and one related to reducing the cost of producing, compressing and distributing gases in our pipelines. Even though they are related, these are two very different ways of delivering products, one by truck and one by pipeline.

Initial optimization systems

In late 2001 BiosGroup developed a Proof of Concept system for a small area of our business that optimized the distribution of oxygen and nitrogen in liquid form by truck from our more than 40 production plants to more than 8,000 customer sites. This system used an ant colony optimizer to determine truck routes and sourcing from our plants. The performance of this system was very impressive, and we realized that there was a good deal of benefit to be gained from extending the system to schedule the production of our liquid products.

In 2002 BiosGroup created a Proof of Concept system for our pipeline operations. This system used a genetic algorithm to decide how to control the pipeline, and it used a mixed integer programming approach to optimize the operations of the plants. While the system did not perform detailed simulation of the costs and performance of our equipment, its results suggested strongly that there were significant savings to be gained if the pipeline optimization system were to be developed into a full-fledged simulator and optimizer.

The liquid gas system

BiosGroup's consulting operations were acquired by NuTech Solutions, Inc. in 2001 and the subsequent development of these systems was carried out by NuTech. Some of the BiosGroup project members continued with the projects after the transition. From this point onward, the continued development of both projects was performed by NuTech Solutions.

In the next major phase of the liquid supply chain production and distribution project, we wanted to find the best solution to plan both production and distribution of our products. It did not seem to us that a good off-the-shelf solution existed that could solve the problem of coordinating production and distribution. The major supply chain software systems optimized first production and then distribution and the results seemed to us to be substantially suboptimal. In fact, we acquired one industrial gas company that had created an award-winning production and distribution optimization system based on a large commercial supply chain product, and its performance seemed to be well below what could be achieved.

It was clear to us that the problem of coordinating production and distribution was not one that could be adequately solved by mathematical techniques such as linear programming, because our plant production profiles were not linear, and neither were our contract terms or plant costs for start-up and shutting down. Most importantly, power costs—the dominant costs for us—were not linear, and they changed at fifteen-minute intervals throughout the day in some areas.

The ant colony optimizer that sourced our orders to plants and scheduled deliveries to our 8,000 customers worked well, but it took a long time to run. We asked a NuTech team to study our problem and determine whether it was possible to produce an optimization system that would integrate both production and distribution (something that the commercial systems known to us did not achieve) while finding high-quality solutions in a six-hour computer run (that is the time between updating our databases at midnight and our need for a 6 am schedule for the next day).

The NuTech team has created a system that we believe to be unique and unprecedented. They have built a genetic algorithm to schedule production at our 40 plants producing liquid gases, and they linked the genetic algorithm to the ant colony optimizer in an ingenious way. A top-level optimizer asks the genetic algorithm and the ant colony optimizer to produce production schedules and distribution schedules. It then evaluates the combination of the production schedule and the distribution schedules in order to find out how well they work together. Each optimizer is then given the feedback from their joint result. In this way, the ant colony optimizer and the genetic algorithm adapt in conjunction with each other to generate integrated schedules, even though neither system is explicitly aware of the operations of the other.

A significant insight derived from this system was the observation that, while the ant system operating alone took many thousands of iterations and several hours to come to a solution, it could run three or four iterations per solution produced by the genetic algorithm, so that the time required to run the two systems linked as we have described was under our six-hour limit.

Today we use the liquid gas system to help us schedule the production and distribution of our liquid products. The cost savings and operational efficiencies are substantial. We are saving more than 1.5 million dollars per quarter at one of our plants by utilizing optimization techniques in a demanding and changing environment.

We are currently extending the liquid production and optimization system in multiple ways, and we expect its benefits to increase as these extensions are completed. We believe that the combination of the genetic algorithm and ant colony optimization greatly exceeds the performance of any commercially available approach to our situation, and we would recommend that a company seeking ways to coordinate and improve their production and distribution operations consider a similar solution.

The pipeline optimizer

There are several features of an industrial gas pipeline operation that are different from natural gas and oil pipeline operations. Since most of the pipelines in the world carry natural gas and oil, the off-the-shelf tools for controlling pipelines are not suited to our operations. In addition, they use optimization techniques that sometimes fail to find optimal solutions—or any feasible solution—when operating conditions change dramatically.

On the strength of the pipeline optimizer Proof of Concept, we asked NuTech Solutions to continue to develop the pipeline optimizer project. The goals of the next phase were to produce more detailed solutions to incorporate more realistic hydraulic models of our pipeline operations and

models of plant and compressor operations, and to optimize a pipeline with equipment that is not modeled in other pipeline optimization systems (such as devices that can change their functions on command, dramatically altering the hydraulics and topology of our pipelines).

The system that NuTech produced in collaboration with our team greatly exceeded our expectations. The system uses a genetic algorithm at the top level, a deterministic heuristic for analyzing pipeline subsystems and setting pressures within each subsystem, a combination of brute force search and genetic algorithm at the plant level to optimize plant production, and multiple heuristics for modifying solutions based on their performance.

The performance of the system is impressive. The operators in our Operations Control Center have learned a good deal in the process of analyzing the solutions produced by the pipeline optimizer, and have modified the way they think about the pipeline and respond to mechanical upsets and breakdowns as a result of studying solutions produced by the optimizer. The optimizer's results have substantially lowered operating costs for the pipeline and have helped us plan for the configuration and installation of new equipment to improve the efficiency of our operations. We are continuing to think about improvements to the existing tools.

For proprietary reasons, we cannot state the full impact of the pipeline optimization system. But we can say that given the outstanding performance of the NuTech team, we were very proud to recognize them by flying them to our Houston offices from Poland, Massachusetts, North Carolina, and California for a two-day event and recognition reception.

Conclusions

The two systems described here have transformed the way that Air Liquide Large Industries U.S. LP does business. We have lowered our costs, improved our efficiency, and increased our planning ability. In one of the media releases jointly issued by Air Liquide and NuTech describing the effects of these systems, Charles Harper said "Our partners at NuTech Solutions painted the yellow brick road for us, they showed us Oz, and then guided us through the journey". We recommend to other companies with similar problems that they too embark on this journey—it has given us a new understanding of what is possible using contemporary approaches to optimization.

References

Air Liquide, www.airliquide.com

NuTech Solutions, www.nutechsolutions.com

About the author

Charles Neely Harper manages and directs the Operations Control Center (OCC) in Houston, Texas, a specialized team and state-of-the art facility that he has spent his entire career building and refining. Charles also holds the title of Air Liquide Group Senior Expert, one of fifty-seven (57) around the world, for his work in Technical Operations - Plants and Pipelines. Charles began his career with Air Liquide in 1977 in the operations of utility and air separation technologies at the Bayport, Texas plant. It was in 1979 that Charles began to develop what was to become the OCC by implementing the first industrial gas program to optimize the pipeline networks along the Texas Gulf Coast and Mississippi River. By 1984, the OCC team could monitor real-time operations on both networks. In 1989, the Center began relying on satellite communications to enable distributed computing to the industrial gas networks. Since then, the OCC team of engineers and specialists has continued to expand the decision support systems that now serve all Air Liquide's primary production facilities in the U.S. Tools and models are applied to pipeline and bulk distribution networks to ensure system integrity, effective communications, operating efficiency and cost containment. Charles N. Harper holds a Bachelors degree from the University of Houston.

New Challenges for Evolutionary Music and Art

Jon McCormack, Centre for Electronic Media Art (CEMA)
Clayton School of Information Technology, Monash University, Australia

Art , it was once said, is anything you can get away with. So it is not surprising that evolutionary approaches to music and art research are challenging our notions of what is classified as "Art" and who is the "creator" of this work. The relatively new field of Evolutionary Music and Art (EMA) falls within the spectrum of Evolutionary Computing. If EC is a relatively young discipline, then EMA is even more so, if we consider Richard Dawkins' "Blind Watchmaker" software (1986) as the epoch in this field.[1] Dawkins' goal was to demonstrate the power of evolution as a design algorithm, one that could design complexity without the need for an explicit designer. It did not take long for people interested in creativity and aesthetics to grasp the significance of this idea and how it might be used to create a new class of art and design: one that was evolved rather than directly created.

The early adopters of Dawkins' "Blind Watchmaker" process (now known variously as *aesthetic selection*, *aesthetic evolution* or *interactive evolution*) were Karl Sims in the USA and William Latham and Stephen Todd in the UK. Latham and Todd developed new computer-aided design software, called "Mutator" which was used by Latham to aesthetically evolve three-dimensional form. in the early 1990s, Latham created a series of otherworldly, organic, surreal virtual sculptures and animated films based on the idea of evolving form. At the same time, Sims used Dawkins' technique to evolve dynamic systems, Lisp expression images, and plant-like structures. Sims' seminal animations *Panspermia* (1990) and *Primordial Dance* (1991) demonstrated the potential for aesthetic evolution to offer genuinely new possibilities for human-computer creativity. Experiments in evolutionary music began at a similar time with a system devised in 1991 by Andrew Horner and David Goldberg that used GAs for thematic bridging. Al Biles' famous *GenJam* (1994) software used evolutionary methods to create an automated jazz accompanist.

[1] I am aware of previous work before Dawkins in this area and EMA research predates Dawkins in related fields such as Cybernetics. However, Dawkins' software is a well-known and significant starting point for much EMA research.

While there have been a number of significant developments and achievements over the last fifteen years, the potential that much of this early EMA research suggested has yet to be reached. It would be easy to dismiss EMA research as exhibiting dilettantism in relation to a serious study of art, however I believe confronting the EMA agenda itself will significantly strengthen its standing as a valid artistic *and* scientific mode of enquiry. Recently, I published a paper that proposed five major research challenges for EMA [1]. The main idea behind this was not to advocate that there are only five problems worthy of study, rather to catalyse a debate as to what the key research goals of this field should be over the next fifteen years.

Before describing some of these challenges here, it is important to look at motivations within the field. Some researchers come from a mathematical or computing background, others from a visual art, sound art, or music composition background, a number possessing skills across several disciplines. What I feel is important to distinguish however, is not the researcher's primary discipline, rather the *goals* of the research itself. This I divide into two broad categories. The first category is EMA systems that are intended to make art or music that is evaluated and appreciated by a human audience. These I call "art-making/understanding" systems. The second category relates to research that explores the concept of creativity in general. These I call "artificial creative" systems.

The first category is where the majority of current systems lie. Their goal is to create a system that produces an output we recognise as art or music. In the case of EMA they will typically use some form of evolutionary computing to produce their results. The output may be highly "individual" in style and specific to one particular researcher or artist. These systems are typically highly specialised with much domain specific knowledge or personal meta-heuristics involved ("no free lunch" at work again). Other research — still in this first "art-making/understanding" category — may look at creative problems in a more general sense or attempt to produce a broader range of creative output, not just one particular individual's style. Whatever the results, the general premise is that the research is oriented around what humans would ascribe aesthetic properties to, regardless of whether the goal is to generate art or to understand and classify human creativity.

The second category is more problematic, but potentially even more challenging than the first. Creative behaviour is not exclusive to humans. Bowerbirds, for example, create elaborate aesthetic constructs that serve no direct survival advantage, rather act as displays to attract mates. EMA research in this second category attempts to look at creativity in a cultural- and species-independent way. Artificial Life proposed to look at life and living systems more broadly, beyond the "life-as-we-know-it", investigating instantiation in non-biological media, such as computation. Similarly, "artificial creative" systems examine creativity in non-biological systems, typically computational, agent-based systems. This agenda might even include the possibility of discovering new forms of creativity ("art-as-it-could-be"). While this may seem an appealing goal, it is highly likely that it will suffer from the same epistemic problems that Artificial Life went through [2], for example how could we recognise creative behaviour in artificial systems if it were significantly different from our understanding of what creative behaviour is? Nevertheless, even though a theory of creativity in general might be unachievable, it is likely that such research could provide new insights into the creativity we currently observe in humans and other animals.

Having detailed these two categories, let us now look at what I believe are some important challenges for current and future research in EMA.

The Search for an Interesting Phenotype

Let's look at an open problem specific to "art-making/understanding systems". A common practice in EMA is one of search for an "interesting phenotype". In this scenario, the artist or programmer designs some form of parameterised system that produces audio or visual output. In most cases, the number of parameters is very large, making an incremental or ordered search of the entire parameter space intractable. Hence the uses of other search techniques such as genetic algorithms or aesthetic selection.

In this mode of evolutionary search there are two primary considerations:

1. the design of the generative system and its parameterisation;
2. the evaluation of the fitness of phenotypes produced by the system.

In the case of aesthetic selection, the fitness evaluation is implicit, being performed by the user of the system. I will return to this second consideration presently, for now let us examine the first point in more detail.

The well-known system of Karl Sims generated images using Lisp expressions evolved by aesthetic selection. In essence these expressions were a combination of basic arithmetic operations and standard mathematical functions such as trigonometric and fractal functions. Even with a limited number of such expressions, the range or gamut of possible images is extremely large. However, it turns out that all of the images produced by such a system are of a certain "class", that is they all look like images made using mathematical expressions. While there might exist a Lisp expression for generating the *Mona Lisa* for example, no such expression has been found by aesthetic selection.

A number of artists and researchers have extended this model, adding a wider variety of mathematical functions, but no matter how many functions are added, the visual results produced by the functions still just look like images made by mathematical functions.

Indeed, in all uses of aesthetic selection the results produced are "of a certain class", that is they exhibit strong traits of the underlying formalized system that created them (the parameterised system). A natural, but unsuccessful strategy has been to increase the scope and complexity of the parameterised system, giving an even larger gamut of possibilities in the phenotype. In all systems to date, this process is limited by the creativity of the artist or programmer, in that they must use their ingenuity to come up with representations and parameterisations they think will lead to interesting results. The search process has shifted up a level (from parameters to mechanisms), but it is still a search problem that needs to be undertaken by humans: it cannot (yet) be formalised, and hence, automated.

What is needed then is a system capable of introducing novelty *within itself*. The physical entities of the Earth were capable of such a task, in that they were able to create an emergent physical replication system. This was achieved from the bottom up, in a non-teleological process of selection, self-assembly and self-organization. It was possible because atoms, molecules, genes, cells and organisms are all physical entities and part of the same system. Generative systems for EMA could exploit such a mechanism.

Hence the open problem is to devise a system where the both the genotype, phenotype and the mechanism that produces phenotype from genotype are capable of automated and robust modification, selection, and hence evolution.

That is, a system that does not produce images of mathematical functions or biomorphs or any particular class of phenotype, due to a fixed parameterised representation. Rather, the genotype, its interpretation mechanism, and the phenotype exist conceptually as part of a singular system, capable of automated modification. Any such system must be *robust* in the sense that it is tolerant of modification without complete breakdown or failure.

It might be argued that the phenotypes produced by DNA are "of a certain class" (i.e. biological organisms), however DNA is able to build organisms, which in the appropriate environment are capable of open-ended creative behaviour. These systems exploit dynamical hierarchies to achieve their complexity. To date, no computerised system has robustly demonstrated such behaviour.

The Problem of Aesthetic Selection

Aesthetic selection of images carried the promise of being able to search for the most beautiful or interesting phenotypes in a parameterised system. In practical terms however, it can only perform a limited search within a certain class of phenotypes, not all possible phenotypes that can be generated by the system. Therefore, the methodology itself tells us little about creativity in general, and does not really offer the most beautiful or interesting images from any system.

This limitation of aesthetic selection leads us to ask why it is does not achieve its goals and what other methods might be better. Aesthetic selection has several problems:

- Population size is limited by the ability of people to perform subjective comparisons on large numbers of objects (simultaneously comparing 16 different phenotypes is relatively easy, comparing 10,000 would be significantly more difficult). In the case of visual phenotypes, the available display size may also limit the number and complexity of phenotypes that can be simultaneously shown in order to perform subjective comparison.
- The subjective comparison process, even for a small number of phenotypes, is slow and forms a bottleneck in the evolutionary process. Human users may take hours to evaluate many successive generations that in an automated system could be performed in a matter of seconds.
- Genotype-phenotype mappings are often not uniform. That is, a minor change in genotype may produce a radical change in phenotype. Such non-uniformities are particularly common in tree or graph based genotype representations such as in evolutionary programming, where changes to nodes can have a radical effect on the resultant phenotype. This problem is not limited to EMA applications and has been widely studied in the EC community.
- The size and complexity of genotypes is limited. In general, simple expressions generate simple images. Complex images require more resources to compute and in a real-time system genotypes that consume too much time or space are usually removed before they can complete. In general, it is difficult to distinguish a genotype that takes a long time to do nothing (such as a recursive null-op) and one that takes a long time to do something interesting (this is analogous to the halting problem). Fractal and IFS functions are often found in aesthetic image systems, as they are an easy way of generating complexity in an image with minimal time and space complexity. The problem is that this is not a general complexity, but a fractal one, with characteristic shapes and patterns, leading to results "of a certain class".

These limitations are indicative of why we can't find the Lisp expression that generates the *Mona Lisa* by aesthetic selection — the human doing the selecting is limiting population size and diversity to such an extent that the genetic algorithm has little chance of finding anything more than local sub-optima. Moreover, the generation scheme, its mapping and complexity, is limited by representation and resources.

Genotype-Phenotype mapping has also been researched. One interesting approach has been to evolve genotypes that represent some computational process, which is itself generative. That is, the genotype specifies the process of construction and then the construction process builds the phenotype. As the construction process itself is evolvable rather than fixed, more complex outcomes are possible.

To address the problems of subjective fitness evaluation by humans, a different approach has been to try to formalize the fitness function; so aesthetic evaluation (either visual or musical) can be automated. However, to date no general formal function for "interesting" or "beautiful", for example, has been found.

This introduces another open problem: to devise formalized fitness functions that are capable of measuring human aesthetic properties of phenotypes. These functions must be machine representable and practically computable.

Aesthetics, while well studied in art theory and philosophy, has yet to be fully understood by science. While there have been some noble attempts to measure aesthetic properties, many consider the proposition itself doomed to failure. The mathematician G. D. Birkhoff famously proposed an "aesthetic measure", equal to order divided by complexity. Birkhoff defined ways of measuring order and complexity for several different categories of object, including two-dimensional polygons and vases. While somewhat successful for simple examples, it failed to capture aesthetic qualities with any generality, being described more as a measure of "orderliness".

Neuroscientists have also studied human aesthetic response in order to gain understanding about what makes us consider things beautiful. The neuroscientist Ramachandran proposed "ten laws of art which cut across cultural boundaries". These included "peak shift" where exaggerated features exemplify learned classifications, grouping, contrast, isolation, symmetry, repetition, rhythm, balance and metaphor.

By definition, aesthetic measures will focus on the *measurable features* of aesthetic objects. These are commonly geometric properties, dimension, proportion, fixed feature categories, organizational structure, etc. The basis being that any such feature or property can be objectively measured directly. However, there are many things considered important to aesthetic theory that cannot be measured directly. These features or properties are generally *interpreted* rather than measured, often in a context-sensitive way. For example, much has been made of harmonious proportions (such as the golden ratio) in nature, art and music. While these measures are interesting and revealing properties of many different types of structure, they say nothing about the semantics of the structure itself. It not only matters that ancient Greek temples exhibit similar geometric golden ratios, but the context of their form in relation to Greek and human culture, the meaning and significance to the observer, and the perceptual physicality (the interpreted physical relation between observer and observed). It seems that such easily measurable general properties are used at the expense of details that are more specific. That is, they are at a too high level of abstraction, where other important features and specific details are ignored. Scientific theories deliberately choose levels of abstraction applicable for physical laws to be "universal". This has been a reasonably successful strategy for the physical universe. For aesthetic laws, however, it appears that general abstractions or simplistic physical measures are not sufficient.

This raises another problem in current research — that the phenotype (normally the art produced by the evolutionary system) can be evolved and measured in isolation from its environment.

The Role of Environment

One common oversight made by those trying to evolve creative systems is proper consideration of environment. Human creative behaviour such as art-making is practiced across all societies (suggesting the possibility of a biological basis), yet art-making is also a social activity, heavily influenced by culture and environment. Modes of artistic practice favoured in one culture may be close to unrecognisable in another. Fads, fashion and style also play important roles in human social systems and play a role in determining acceptability and popularity of creative acts.

Organisms can be considered complex adaptive systems: adapting to their ecological, environmental and social niches. In this way creative systems could be considered "mirrors" of their environment, so if we build a better and more complex environment in our simulation, we should expect the creative agents who populate that environment to reflect this complexity and detail. Despite an abundance of research in evolutionary biology, social sciences and psychology, most EMA systems have yet to incorporate many of these environmental, cultural and mimetic phenomena into their worldview. The design of environments from which creative behaviour is expected to emerge is at least as important as the design of the agents who are expected to evolve this behaviour.

Research into EMA should include developing systems and devices capable of being recognised by the art community as successful art-generating devices, irrespective of the technical methodology used to create them. This leads to another important open problem: how to make good instruments.

The Extended Interface

So now let us consider the class of evolutionary systems designed for use as art-making machines. Humans have been able to devise numerous musically or visually creative physical devices. When a competent musician interacts with a cello or piano for example, it becomes clear that the instrument acts as a physical cognitive extension to the performer. In a Cybernetic sense, musician and instrument are one, with brain, body and instrument intimately linked as a performance system. Similarly, a seemingly simple tool such as the pencil is capable of a vast array of artistic possibility when placed in the right hands. These creative systems exploit the brain's plasticity in incorporating physical tools and cultural practices as extensions. This idea is based on theories of "extended mind", rooted in Cybernetics research and championed today by researchers such as Andy Clark and Mike Wheeler [3, 4].

If we compare artistic tools such as pencils and pianos to most creative computer software we typically find the software lacking. Mimicry (such as drawing software) is common, and while such systems do offer greater convenience and flexibility over their physical counterparts, they lack a true sense of immediacy, physical tactility and environmental interaction.

We might consider the way most people interact with (and author) software as "physically passive" in the sense that it is predominantly a conceptual exercise, like writing. Computers, in essence, are symbol manipulators and programming languages, for the most part, require the programmer to conceptualise in terms of symbols and the processes that act on them. Take a look at someone hunched up over a keyboard, mouse and computer screen and their physical movement and interaction is highly constrained — little taps on the keyboard and jittery mouse movements — the interface tools mere intermediary inconveniences between expressive intent and result. This mode of interaction has little or no physical expression. Here we see echoes of the symbolic/connectionist debate that preoccupied AI research for many years. The argument here is not for one of adding "usability" or incorporating standard principles of Human-Computer Interaction (HCI) into software design. It is one of conceptualising software as a "performance instrument" — one that is, in the words of Golan Levin, "instantly knowable, and indefinitely masterable". Anyone can pick up a pencil and begin using it, however it may take years of training and practice to master and achieve creative results of significance. One should hope for similar possibilities in the next generation of evolutionary digital tools.

As with physical instruments, all software places constraints on the scope of the interactions and possibilities it permits. However, the possibilities offered within the constraints define the creative potential of that tool. It is therefore imperative to consider the constraints carefully when designing creative tools. As software has no obvious physical constraints, one needs to conceptualise differently, working within and through the constraints to achieve the best outcomes.

Where do evolutionary systems fit into this proposal? One argument is that many so called, "generative systems" potentially offer highly unique and novel phase-spaces, capitalising on the emergent properties these systems typically display. However, the difficulty is in locating these novel phase-spaces and exploring them intuitively — moving the system from one state to another in ways that are creatively rewarding (composition) and surprising (improvisation). In order to do this effectively one must feel an intimacy with the system (possibly gained over years of exploration and practice) that allows them to instinctively anticipate how to "play" the system in order to get the best results [5].

One possibility is for evolutionary and adaptive systems to assist in the exploration and search of this phase space, guiding without dictating, being plastic (in the way that the brain is plastic), leading human and machine to a synergistic embrace of new possibilities. It is in these modes of engagement with machines that we may really see some astonishing results that go much further than any current physical instruments can. That challenge remains for current and future researchers in evolutionary music and art: to make a "software instrument" that equals or exceeds traditional instruments in terms of creative possibility. We will know we have succeeded when these tools are used by many, mastered by a few; subject to study in Art and Music schools; embraced by cultural institutions as significant new art forms. It would be easy to argue that this is already the case for some computer tools, however a closer analysis shows that this is really only part of the broader automation of society and culture that the computer has lead over the last fifty or so years. Yes, artists and designers now work with computers, but in most cases this has been with software that mimics traditional tools (pencils, cameras, piano keyboards), not offering media and results unique to computation.

This is an exciting time to be a researcher involved in evolutionary music and art. The challenges to current and future researchers are both numerous and ambitious. The pioneering achievements of early work suggested a vast potential for the field. This has been progressed over the last fifteen years. In this article I have touched on a few issues I think worthy of investigation over the next fifteen years. I believe EMA has no difficulty in finding a worthy research agenda. What is more difficult is addressing this agenda in a methodical and erudite way. EMA research has an important role to play in our understanding of creativity in terms of its mechanisms, purpose, and ultimately, its definition. This gives it a unique status in contributing to both scientific knowledge and contemporary culture.

Bibliography

[1] McCormack, J. 2005, Open Problems in Evolutionary Music and Art, in Rothlauf, F. (ed) Evoworkshops 2005, LNCS 3449, Springer-Verlag, Berlin; Heidelberg. pp. 428-436 (available on-line at http://www.csse.monash.edu.au/~jonmc/resources/OpenProblemsSV.pdf)

[2] Pattee, H.H. 1995, Artificial Life Needs a Real Epistemology, in Morán, F., A. Moreno, J.J. Merelo & P. Chacón (eds), Advances in Artificial Life: Third European Conference on Artificial Life, Springer-Verlag pp. 23-38.

[3] Clark, A. 2003, Natural-Born Cyborgs: Minds, Technologies, and the Future of Human Intelligence, Oxford University Press, New York.

[4] Wheeler, M. 2005, Reconstructing the Cognitive World: The Next Step, MIT Press, Cambridge, Mass.

[5] Eldridge, A.C. 2005, Cyborg Dancing: Generative Systems for Man-Machine Musical Improvisation, in Innocent, T. et. al. (eds) Proceedings of Third Iteration, CEMA, Melbourne. pp. 129-141.

About the author

Jon McCormack is an Australian-based researcher in Artificial Life and Evolutionary Music and Art. His research interests include generative evolutionary systems, machine learning, L-systems and developmental models. McCormack is also a practising electronic media artist. He holds an Honours degree in Applied Mathematics and Computer Science from Monash University, a Graduate Diploma of Art from Swinburne University and a PhD in Computer Science from Monash University. He is currently Senior Lecturer in Computer Science and co-director of the Centre for Electronic Media Art (CEMA) at Monash University in Melbourne, Australia. CEMA is an interdisciplinary research centre established to explore new collaborative relationships between computing and the arts. The monograph “Impossible Nature: the art of Jon McCormack" was published by the Australian Centre for the Moving Image in 2005, and documents McCormack’s creative achievements over the last decade.
Homepage: www.csse.monash.edu.au/~jonmc
Email: jonmc@csse.monash.edu.au

About the cover

The cover image and the images in this article are from “Eden” an interactive, evolutionary sonic ecosystem. Copyright 2000-2005 Jon McCormack.

Eden is an example of interactive evolutionary art system. Virtual creatures evolve to make interesting sounds based on the interest of people who visit the installation. The presence of people near the artwork influences the production of food resources for the virtual creatures. Over time, the creatures learn to make interesting sounds in order to keep the human audience interested and hence increase the supply of food. Eden uses a custom Learning Classifier System (LCS) to achieve this. For more information see www.csse.monash.edu.au/~jonmc/projects/eden/

Open BEAGLE

A C++ Framework for your Favorite Evolutionary Algorithm

Christian Gagné, University of Lausanne, *christian.gagne@unil.ch*
Marc Parizeau, Université Laval, *parizeau@gel.ulaval.ca*

Numerous Evolutionary Computations (EC) software tools are now publicly available to the community – see for instance [1] and [2] for a listing of the most well known. The majority of these tools are specific to a particular EC flavor, however, only a few are truly generic EC softwares [3]. The highly diverse and adaptable nature of Evolutionary Algorithms (EA) make generic EC software tools a must-have for rapid prototyping of new approaches. As we all know, EC comprises a broad family of techniques where populations of solutions to problems are represented by some appropriate data structures (e.g. bit strings, real-valued vectors, trees, etc.) on which variation operators (e.g. mutation, crossover, etc.) are applied using iterative algorithms inspired from natural evolution. Different fitness measures can also be used, with one or several objectives, and it is possible to co-evolve several species of solutions, with different species represented by possibly different data structures.

To allow such great flexibility, these tools require a well-designed software architecture, generally attained using Object Oriented (OO) concepts [3]. Generic EC tools are thus significantly more complex than specialized tools, given all of the underlying mechanisms necessary for component replacement in representation, fitness, variation and selection operations, as well as evolutionary model. In the short-run, these mechanism may induce some cost to the user who is confronted with a somewhat steeper learning-curve. But we argue that, in the long-run, they also provide a very beneficial return on this investment, by allowing the efficient unification of different EC paradigms around a single flexible OO framework, which can provide elaborate additional facilities like dynamic configuration, logging and check-pointing of evolutions. Moreover, the maturing of a generic EC toolbox can, in the end, enable the construction of a black-box model, where components can be glued together without explicit programming, using a graphical interface and some scripting language.

The Open BEAGLE Framework

In 1999, the development of a small lightweight C++ library for Genetic Programming (GP) was started at Université Laval as a summer internship project. Three years later, in January 2002, after two complete rewrites, a generic C++ EC framework was publicly released as Open BEAGLE[1] [4]. Version 1.0 was released in July 2002, then version 2.0 in September 2003, and later on version 3.0 in October 2005.

While enabling most any EC paradigm through its generic mechanisms, Open BEAGLE currently provides direct support of the following major EA, through specialized layers:

- Bit string, integer-valued, and real-valued GA;
- Anisotropic self-adaptive ES and Covariance Matrix Adaptation ES (CMA-ES);
- Tree-based GP;
- Evolutionary multi-objective optimization (NSGA-II and NPGA2);
- Co-evolution through the use of multi-threading.

[1] BEAGLE refers to the name of the English vessel, HMS Beagle, on which Charles Darwin embarked for his famous voyage around the world. It also stands as a recursive acronym for: the Beagle Engine is an Advanced Genetic Learning Environment.

A general and extensible XML file format also allows the specification of EA configurations and parameters, logging of output and debugging information, and check-pointing of evolutions. With this file format, a web interface called BEAGLE Visualizer was designed to allow the viewing of basic evolution statistics using a standard web browser.

Another interesting derivative of Open BEAGLE is called *distributed* BEAGLE [4]. It enables the transparent distribution of the fitness evaluation tasks of any EA over a Beowulf cluster or a grid of workstations on a LAN. Distributed BEAGLE uses a master-slave architecture [5], where the master implements an abstract layer between evolution slaves that evolve a population to the next generation, and evaluation slaves that evaluate the fitness of individuals.

Code Example: OneMax

Despite the inherent complexity of a generic EC framework, the use of Open BEAGLE is relatively simple for a novice programmer. The different components have default values and policies that are suitable for most simple applications. The user is only required to define a fitness evaluation operator and a main method that initializes the different components of the framework. The following listing presents an evaluation operator implementation for the classical GA bit string example OneMax, which consists in searching for bit strings that have a maximum number of bits set to "1".

```
 1. #include "beagle/GA.hpp"
 2. using namespace Beagle;
 3. class OneMaxEvalOp : public EvaluationOp {
 4. public:
 5. OneMaxEvalOp() : EvaluationOp("OneMaxEvalOp") { }
 6. virtual Fitness::Handle
 7.  evaluate(Individual& inIndividual, Context& ioContext)
 8.  {
 9.    GA::BitString::Handle lBitString =
10.      castHandleT<GA::BitString>(inIndividual[0]);
11.    unsigned int lCount = 0;
12.    for(unsigned int i=0; i<lBitString->size(); ++i)
13.      if((*lBitString)[i]) ++lCount;
14.    return new FitnessSimple(float(lCount));
15.  }
16. };
```

In this listing, line 5 is to construct a fitness operator named `OneMaxEvalOp`. Lines 6 to 15 corresponds to the function called to evaluate an individual fitness. Lines 9 and 10 cast the generic individual to evaluate into a bit string individual. Lines 11 to 13 count the number of ones in the bit string while line 14 returns the fitness measure, that is a single real value to maximize.

Now, the following listing presents the associated main routine for the OneMax problem.

```
 1. #include <cstdlib>
 2. #include <iostream>
 3. #include "beagle/GA.hpp"
 4. #include "OneMaxEvalOp.hpp"
 5. using namespace Beagle;
 6. int main(int argc, char** argv) {
 7.   try {
 8.     GA::BitString::Alloc::Handle
 9.       lBSAlloc = new GA::BitString::Alloc;
10.     Vivarium::Handle lVivarium = new Vivarium(lBSAlloc);
11.     OneMaxEvalOp::Handle lEvalOp = new OneMaxEvalOp;
12.     const unsigned int lNumberOfBits = 20;
13.     GA::EvolverBitString lEvolver(lEvalOp, lNumberOfBits);
14.     System::Handle lSystem = new System;
15.     lEvolver.initialize(lSystem, argc, argv);
16.     lEvolver.evolve(lVivarium);
17.   }
18.   catch(Exception& inException) {
19.     inException.terminate(std::cerr);
20.   }
21.   return 0;
22. }
```

Lines 8, 9, and 10 build a bit string population. Line 11 instantiates the fitness evaluation operator defined above. Lines 12 and 13 define a bit string GA evolver where individuals are initialized as a string of 20 bits each. Line 14 creates the evolution system while line 15 initializes the evolver and the evolution system, parses the command line, and reads configuration files. Finally, the evolution is launched at line 16. The entire routine is in a try-catch block in order to intercept exceptions which may be thrown by Open BEAGLE, if a problem is detected at runtime.

Different configurations of the evolutionary algorithm is possible. For example, if the user wants to use a steady-state GA instead of the default generational model, he must define a XML configuration file similar to the following one.

```
<?xml version="1.0"?>
<Beagle>
  <Evolver>
    <BootStrapSet>
      <GA-InitBitStrOp/>
      <OneMaxEvalOp/>
      <StatsCalcFitnessSimpleOp/>
    </BootStrapSet>
    <MainLoopSet>
      <SteadyStateOp>
        <OneMaxEvalOp>
          <GA-CrossoverOnePointBitStrOp
            matingpb="ga.cx1p.prob">
            <SelectTournamentOp/>
            <SelectTournamentOp/>
          </GA-CrossoverOnePointBitStrOp>
        </OneMaxEvalOp>
        <OneMaxEvalOp>
          <GA-MutationFlipBitStrOp
            mutationpb="ga.mutflip.indpb">
            <SelectTournamentOp/>
          </GA-MutationFlipBitStrOp>
        </OneMaxEvalOp>
        <SelectTournamentOp repropb="ec.repro.prob"/>
      </SteadyStateOp>
      <StatsCalcFitnessSimpleOp/>
      <TermMaxGenOp/>
      <MilestoneWriteOp/>
    </MainLoopSet>
  </Evolver>
</Beagle>
```

No re-compilation is necessary, the user only needs to execute the program with a command-line option referring to the previous configuration file. This example, as well as many others, are packaged together with the source code of Open BEAGLE.

Conclusion

Open BEAGLE is an open source LGPL framework for EC, freely available on the Web [4]. Written in C++, it is adaptable, portable, and quite efficient. Given its open and generic nature, it can be used to federate software development for EC, using an ever-growing library of components and tools, some of which have already been donated by different researchers from different institutions around the world. Through this newsletter, the authors hope that new EC researchers can join the pool of Open BEAGLE users and, eventually, become BEAGLE developers that contribute in their modest way to the progress of our community.

Bibliography

[1] John Eikenberry. GNU/Linux AI and Alife HOWTO. http://zhar.net/howto/html.

[2] EvoWeb Software Listing. http://evonet.lri.fr/evoweb/resources/software.

[3] Christian Gagné and Marc Parizeau. Genericity in evolutionary computation software tools: Principles and case study. *International Journal on Artificial Intelligence Tools*, 15(2):173–174, April 2006.

[4] Christian Gagné and Marc Parizeau. Open BEAGLE W3 page. http://beagle.gel.ulaval.ca.

[5] Marc Dubreuil, Christian Gagné, and Marc Parizeau. Analysis of a master-slave architecture for distributed evolutionary computations. *IEEE Transactions on Systems, Man, and Cybernetics – Part B*, 36(1):229–235, February 2006.

About the authors

Christian Gagné obtained his Ph.D. degree in 2005 from Université Laval, Québec City, Canada. He is currently postdoctoral fellow jointly at the TAO Team of the INRIA Futurs, located in the Laboratoire de Recherche en Informatique of the Université Paris-Sud, Orsay, France, and at the Information Systems Institute of the University of Lausanne, Switzerland. He is supported by postdoctoral fellowships from the ERCIM (Europe) and the FQRNT (Québec). His research interests include evolutionary computations, machine learning, pattern recognition, artificial neural networks, object-oriented software engineering, and high-performance computing. He is also the principal developer and maintainer of Open BEAGLE.

Marc Parizeau obtained his Ph.D. degree in 1992 from École Polytechnique de Montréal, Canada. He is now a full professor at Université Laval, Québec City, Canada. His main research interests are in pattern recognition and computational intelligence, including evolutionary computations, artificial neural networks, and incremental learning.

Grand Opening of MEDAL

I am pleased to announce the grand opening of the Missouri Estimation of Distribution Algorithms Laboratory (MEDAL) located at the Department of Math and Computer Science at the University of Missouri in St. Louis.

MEDAL focuses on the design, enhancement, analysis, and applications of estimation of distribution algorithms (EDAs), which represent a powerful class of stochastic optimization techniques inspired by evolutionary computation and machine learning. Besides EDAs, MEDAL's research interests cover other branches of genetic and evolutionary computation, learning classifier systems, bioinformatics, and machine learning.

The web page of MEDAL located at http://medal.cs.umsl.edu/ provides online downloads of technical reports and publications, source code and selected presentations.

MEDAL is supported by the National Science Foundation under CAREER grant ECS-0547013, the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant FA9550-06-1-0096, and the University of Missouri in St. Louis through the High Performance Computing Collaboratory sponsored by Information Technology Services, and the Research Award and Research Board programs.

For questions, comments, and suggestions, please contact MEDAL at the following address:

Missouri Estimation of Distribution Algorithms Laboratory

Department of Math and Computer Science, 321 CCB
University of Missouri–St. Louis
One University Blvd., St. Louis, MO 63121
E-mail: medal@cs.umsl.edu
WWW: http://medal.cs.umsl.edu/

Martin Pelikan
Director, Missouri Estimation of Distribution Algorithms Laboratory

Dissertation Corner

Test techniques for advanced processors

Doctoral Thesis by Edgar Ernesto Sánchez Sánchez

In the last years, performance of processor and microprocessor cores has impressively increased due to mainly four aspects: huge quantities of resources, high work frequencies, low power and elevated parallelism of instruction execution. Even though all these issues are correlated, physical resources, work frequencies, and low power are directly backed by advances in technology, while parallel execution depends on the evolution of the processor architecture. Today most of the integrated circuit (IC) manufacturing cost is brought about by the test and validation processes, raising costs up to near 70%. The problem of test generation is especially critical in the case of high-performance circuits such as microprocessors and microcontrollers. Regarding to modern processors, the only possibility to test and verify both practically and economically processor cores, relies on the execution of carefully crafted programs. These programs, usually called test programs, are composed of a valid sequence of assembly instructions, that is fed to the processor through its normal execution instruction mechanism, and whose goal is to uncover any possible design or production flaw in the device under test. An automatic software-based method to automatically generate test programs should be characterized by (i) high flexibility regarding the target microprocessor, in order to allow the maximum applicability of the method; (ii) syntactically correct generation of assembly programs depending on the specific singularities of the target processor; (iii) high versatility with respect to the evaluation system in order to allow tackling different problems such as test or validation; (iv) the ability of driving the generation process exploiting coverage metrics, for example statement coverage, as feedback values.

In this thesis, a software-based methodology to automatically generate test programs is described. The methodology is based on an evolutionary algorithm able to automatically generate test programs for microprocessor cores, and may be used for different processors since their instruction set architecture is described in the form of an instruction library, and because a fitness function can be defined, computed, and used to drive the automatic generation of test programs. The evolutionary algorithm, called μGP, is composed of three blocks: the instruction library that describes the microprocessor assembly language; the *μGP core* that cultivates a population of individuals, where each individual is a assembly program; and finally, the external evaluator that simulates or executes the generated programs delivering a fitness value to the evolutionary core. The loose coupling between the instruction library and the generator enables exploiting the approach with different instruction sets, formalisms and metrics. The evolutionary approach was initially exploited in [1] and the latest improvements were presented in [2]. In this thesis, a description about the algorithm, its evolution, the main characteristics of the evolutionary process, the more recent novelties such as the exploitation of previously written programs, and the flexibility of the method are presented. The usefulness of the algorithm is backed up by the presentation of 3 different flavored cases of study: the first one tackles the verification of the DLX/pII processor, the second one generates post-silicon verification programs for the Pentium 4, and the third one evolves a test set for the PLASMA processor. The experimental results demonstrate the algorithm versatility and efficiency.

Bibliography

[1] F. Corno, E. Sanchez, M. Sonza Reorda, G. Squillero, “Automatic Test Program Generation - a Case Study”, *IEEE Design & Test*, vol. 21, issue 2, 2004, pp 102-109

[2] F. Corno, E. Sánchez, M. Sonza Reorda, G. Squillero, “Evolving Assembly Programs: How Games Help Microprocessor Validation”, IEEE Trans. on Evolutionary Computation, Dec. 2005, vol. 9, pp. 695-706

Edgar Ernesto Sánchez Sánchez is a Post-doctoral researcher at the Politecnico di Torino. His research interests include test techniques for advanced processors and evolutionary algorithms. Sánchez has a degree in electronic engineering from Universidad Javeriana, Colombia. For additional information about uGP, please visit http://www.cad.polito.it/research/microgp.html.

Forthcoming Papers

Evolutionary Computation, Volume 14, Number 2

- **Genetic Parallel Programming: Design and Implementation**, Sin Man Cheang, Kwong Sak Leung, and Kin Hong Lee, 129–156
- **Error Thresholds in Genetic Algorithms**, Gabriela Ochoa, 157–184
- **A probabilistic Classifier System and its application in Data Mining**, Jorge Muruzábal, 185–224
- **GASAT: a genetic local search algorithm for the satisfiability problem**, Frédéric Lardeux, Frédéric Saubion, and Jin-Kao Hao, 225–255

Genetic Programming and Evolvable Machines

- **Introducing Lateral Thinking in Search Engines**, Y. Landrin-Schweitzer, P. Collet and E. Lutton
- **Solving differential equations with genetic programming**, I. Lagaris
- **Emergence of Genomic Self-Similarity in Location Independent Representations**, A. Wu and I. Garibay
- **Morphological Algorithm Design for Binary Images**, M. Quintana and R. Poli
- **Coevolution and the Red Queen Effect Shape Virtual Plants**, M. Ebner
- **Finding Needles in Haystacks is Harder with Neutrality**, M. Collins
- **Using Genetic Algorithms for Early Schedulability Analysis and Stress Testing in Real-Time Systems**, L. Briand, Y. Labiche, and M. Shousha
- **On the Impact of Objective Function Transformations on Evolutionary and Black-Box Algorithms**, T. Storch
- **Multi-objective Evolutionary Design and Knowledge Discovery of Logic Circuits with an Improved Genetic Algorithm**, S. Zhao and L. Jiao
- **GP-Sumo: Using Genetic Programming to Evolve Sumobots**, S. Sharabi and M. Sipper
- **Shortcomings with Using Edge Encodings to Represent Graph Structures**, G. Hornby
- **“Special Issue on "Evolutionary Computation in Dynamic and Uncertain Environments”**

In Other Journals

- D. Ortiz-Boyer, C. Hervás-Martínez and N. García-Pedrajas (2005) "CIXL2: A Crossover Operator for Evolutionary Algorithms Based on Population Features", JAIR, Volume 24, pages 1-48. http://www.jair.org/papers/paper1660.html

New Books

MIT Press

- *Evolutionary Computation A Unified Approach*, Kenneth A. De Jong. 2006, 250 pp., 95 illus., cloth, ISBN 0-262-04194-4.

Springer Natural Computing Series

- *Experimental Research in Evolutionary Computation – The New Experimentalism*, Thomas Bartz-Beielstein. 2006, XIV, 214 p. 66 illus., Hardcover. [WWW]
- *Nanotechnology: Science and Computation*, Junghuei Chen, Natasha Jonoska, Grzegorz Rozenberg Editors, 2006, XII, 393 p., Hardcover. [WWW]
- *Theoretical and Experimental DNA Computation*, Martyn Amos, 2005, XIII, 173 p., Hardcover. [WWW]
- *Biologically Inspired Algorithms for Financial Modelling*, Anthony Brabazon, Michael O'Neill 2006, XVI, 275 p. 92 illus., Hardcover. [WWW]
- *Differential Evolution: A Practical Approach to Global Optimization*, Kenneth V. Price, Rainer M. Storn, Jouni A. Lampinen 2005, XX, 538 p. 292 illus. with CD-ROM., Hardcover. [WWW]
- *Applications of Membrane Computing*, Gabriel Ciobanu, Mario J. Pérez-Jiménez, Gheorghe Păun 2006, X, 439 p., Hardcover. [WWW]
- *Spatially Structured Evolutionary Algorithms: Artificial Evolution in Space and Time*, Marco Tomassini 2005, XIII, 193 p., Hardcover. [WWW]

Studies in Fuzziness and Soft Computing

- *Extending the Scalability of Linkage Learning Genetic Algorithms Theory & Practice*, Ying-ping Chen. Vol. 190, 2006, XX, 120 p. 37 illus., Hardcover. [WWW]
- *Rule-Based Evolutionary Online Learning Systems: A Principled Approach to LCS Analysis and Design*, Martin V. Butz. Vol. 191. 2006, XXI, 266 p. 82 illus., Hardcover. [WWW]
- *Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms*, J.A. Lozano, P. Larrañaga, I. Inza, E. Bengoetxea (Eds.) Vol. 192, 2006, XV, 294 p. 109 illus., Hardcover. [WWW]

Lecture Notes in Computer Science

- Genetic Programming: 9th European Conference, EuroGP 2006, Budapest, Hungary, April 10-12, 2006. Proceedings Series: Lecture Notes in Computer Science, Vol. 3905 Editors: Marco Tomassini et al. 2006, XI, 361 p., Softcover. [WWW]
- Applications of Evolutionary Computing: EvoWorkshops 2006, Budapest, Hungary, April 10-12, 2006, Proceedings Series: Lecture Notes in Computer Science, Vol. 3907 Editors: Franz Rothlauf, et al. 2006, XXIV, 813 p., Softcover. [WWW]
- Evolutionary Computation in Combinatorial Optimization 6th European Conference, EvoCOP 2006, Budapest, Hungary, April 10-12, 2006, Proceedings Series: Lecture Notes in Computer Science, Vol. 3906 Gottlieb, Jens; Raidl, Günther R. (Eds.) 2006, XI, 293 p., Softcover. [WWW]

Calls and Calendar

June

ALife X

June 3-7, 2006, Bloomington, IN USA

www.alifex.org

The Artificial Life X conference marks two decades of the birth of this enterprise, a period marked by vast advances in the life sciences. The conference will showcase the best current work in all areas of research in Artificial Life, while highlighting its achievements and challenges, especially in an age of unparalleled computational power and access to data about various biological processes.

July

GECCO-2006: Late-Breaking Papers

Seattle, Washington, USA, July 8-12, 2006

http://www.sigevo.org/GECCO-2006/

Deadline May 5, 2006

Papers describing late-breaking developments in the field of genetic and evolutionary computation are being solicited for presentation at the Genetic and Evolutionary Computation 2006 Conference (GECCO-2006) to be held on July 8-12 (Saturday-Wednesday), 2006 at the Renaissance Hotel in Seattle, Washington, USA, and inclusion in a special CD-ROM to be distributed to all attendees of the conference. This special CD-ROM is distinct from the conference proceedings.

GECCO-2006: Human-Competitive Results

Submission Deadline May 29, 2006

www.human-competitive.org

Entries are now being solicited for awards totaling $10,000 for 2006 awards for human-competitive results that have been produced by any form of genetic and evolutionary computation (including, but not limited to genetic algorithms, genetic programming, evolution strategies, evolutionary programming, learning classifier systems, grammatical evolution, gene expression programming, differential evolution, etc.) and that have been published in the open literature between June 20, 2005 (the deadline for the previous competition) and the deadline for 2006 entries, namely Monday May 29, 2006.

GECCO-2006: Competition

Seattle, Washington, USA, July 8-12, 2006

http://cswww.essex.ac.uk/staff/rpoli/GECCO2006/

Deadline June 14, 2006

A number of competitions will take place as part of GECCO 2006. These include (i) Prime prediction, (ii) TinyGA, and (iii) Pasta segmentation.

August

ACM KDD-2006

August 20 - 23, 2006 Philadelphia, USA

http://www.acm.org/sigs/sigkdd/kdd2006/

The 12th ACM SIGKDD conference will provide a forum for researchers from academia, industry, and government, developers, practitioners, and the data mining user community to share their research and experience. The SIGKDD conference will feature keynote presentations, oral paper presentations, poster presentations, workshops, tutorials, and panels, as well as the KDD Cup competition.

ECAI Workshop on Evolutionary Computation

Riva del Garda, Italy, 28 August 2006

http://www.ce.unipr.it/ec2ai2006

The workshop will comprise tutorials and technical presentations, in order to address the participation of as wide an audience as possible, from researchers and students who are already working in the field of Evolutionary Computation and Artificial Intelligence, to representatives of industry and everybody who is interested in approaching evolutionary computation from the point of view of both basic research and applications.

September

PPSN - Parallel Problem Solving from Nature IX

Reykjavik, Iceland, September 9 - 13, 2006

http://ppsn2006.raunvis.hi.is/

The conference emphasises original theories and novel applications of natural computing. World-leading researchers in the field of natural computing will present keynote talks and tutorials at the conference. The conference proceedings will be published by Springer in its LNCS series. A number of specialist workshops will be run just before the main conference.

From Animals to Animats 9

25-29 September 2006, CNR, Roma, Italy

http://sab06.org

The objective of this interdisciplinary conference is to bring together researchers in computer science, artificial intelligence, alife, control, robotics, neurosciences, ethology, evolutionary biology, and related fields so as to further our understanding of the behaviors and underlying mechanisms that allow natural and artificial animals to adapt and survive in uncertain environments.

ABiALS-2006

30 September 2006, CNR, Roma, Italy

http://www-illigal.ge.uiuc.edu/ABiALS/

Deadline June 15, 2006

The Anticipatory Behavior in Adaptive Learning Systems (ABiALS) workshops are designed to encourage interdisciplinary research on anticipatory behavior in animals, animats, and artificial intelligence systems. Submission that investigate anticipatory behavior mechanisms are encouraged.

October

SEAL'06

15-18 October 2006, Hefei, Anhui, China

http://nical.ustc.edu.cn/seal06/

Evolution and learning are two fundamental forms of adaptation. SEAL'06 aims at exploring these two forms of adaptation and their roles and interactions in adaptive systems. Cross-fertilisation between evolutionary learning and other machine learning approaches will be strongly encouraged by the conference. The other major theme of the conference is optimisation by evolutionary approaches or hybrid evolutionary approaches.

January 2007

Foundations of Genetic Algorithms

7-11 January 2007, Mexico City, Mexico

Deadline September 20, 2006

Requests for attendance due September 20, 2006

We invite submissions of extended abstracts for the ninth biennial workshop on the Foundations of Genetic Algorithms. The workshop covers the theoretical foundations of all forms of evolutionary computation. FOGA will be held 7-11 January, 2007 in Mexico City. Attendance at the workshop will be limited; the goal is to create a small interdisciplinary forum with close interaction among participants from different fields - evolutionary computation, population genetics, animal behaviour, physics and biochemistry, among others. Individuals submitting papers will be given priority for attendance, and some slots will be reserved for students. Anyone wishing to attend must indicate this by either submitting a paper or requesting attendance in advance (see deadline).

Extended abstracts must be received by 20th September, 2006. Submissions should address theoretical issues in evolutionary computation. Papers that consider foundational issues and/or are of a multidisciplinary nature are especially encouraged. This does not preclude the acceptance of papers that use an experimental approach, but such work should be directed towards validation of suitable hypotheses concerning foundational matters.

Extended abstracts should be between 10-12 pages (single column). To submit an extended abstract, please email a compressed postscript or a pdf file to stephens@nucleares.unam.mx and mtoussai@inf.ed.ac.uk no later than 20th September 2006. In their submission message authors should provide the title of the paper, and the name, address and affiliation of the author(s). Authors should submit papers in single column format with standard spacing and margins, and 11pt or 12pt font for the main text. Authors using LaTeX should either use the standard article style file or the FOGA style file which can be found at the conference web-site.

About the Newsletter

SIGEVOlution is the newsletter of SIGEVO, the ACM Special Interest Group on Genetic and Evolutionary Computation.

To join SIGEVO, please follow this link [WWW]

Contributing to SIGEVOlution

We solicite contributions in the following categories:

Art: Are you working with Evolutionary Art? We are always looking for nice evolutionary art for the cover page of the newsletter.

Short surveys and position papers: We invite short surveys and position papers in EC and EC related areas. We are also interested in applications of EC technologies that have solved interesting and important problems.

Software: Are you are a developer of an EC software and you wish to tell us about it? Then, send us a short summary or a short tutorial of your software.

Lost Gems: Did you read an interesting EC paper that, in your opinion, did not receive enough attention or should be rediscovered? Then send us a page about it.

Dissertations: We invite short summaries, around a page, of theses in EC-related areas that have been recently discussed and are available online.

Meetings Reports: Did you participate to an interesting EC-related event? Would you be willing to tell us about it? Then, send us a short summary, around half a page, about the event.

Forthcoming Events: If you have an EC event you wish to announce, this is the place.

News and Announcements: Is there anything you wish to announce? This is the place.

Letters: If you want to ask or to say something to SIGEVO members, please write us a letter!

Suggestions: If you have a suggestion about how to improve the newsletter, please send us an email.

Contributions will be reviewed by members of the newsletter board.

We accept contributions in LaTeX, MS Word, and plain text.

Enquiries about submissions and contributions can be emailed to pierluca.lanzi@polimi.it.

Notice to Contributing Authors to SIG Newsletters

By submitting your article for distribution in the Special Interest Group publication, you hereby grant to ACM the following non-exclusive, perpetual, worldwide rights:

- to publish in print on condition of acceptance be the editor
- to digitize and post your article in the electronic version of this publication
- to include the article in the ACM Digital Library
- to allow users to copy and distribute the article for noncommercial, educational or research purposes

However, as a contributing author, you retain copyright to your article and ACM will make every effort to refer requests for commercial use directly to you.