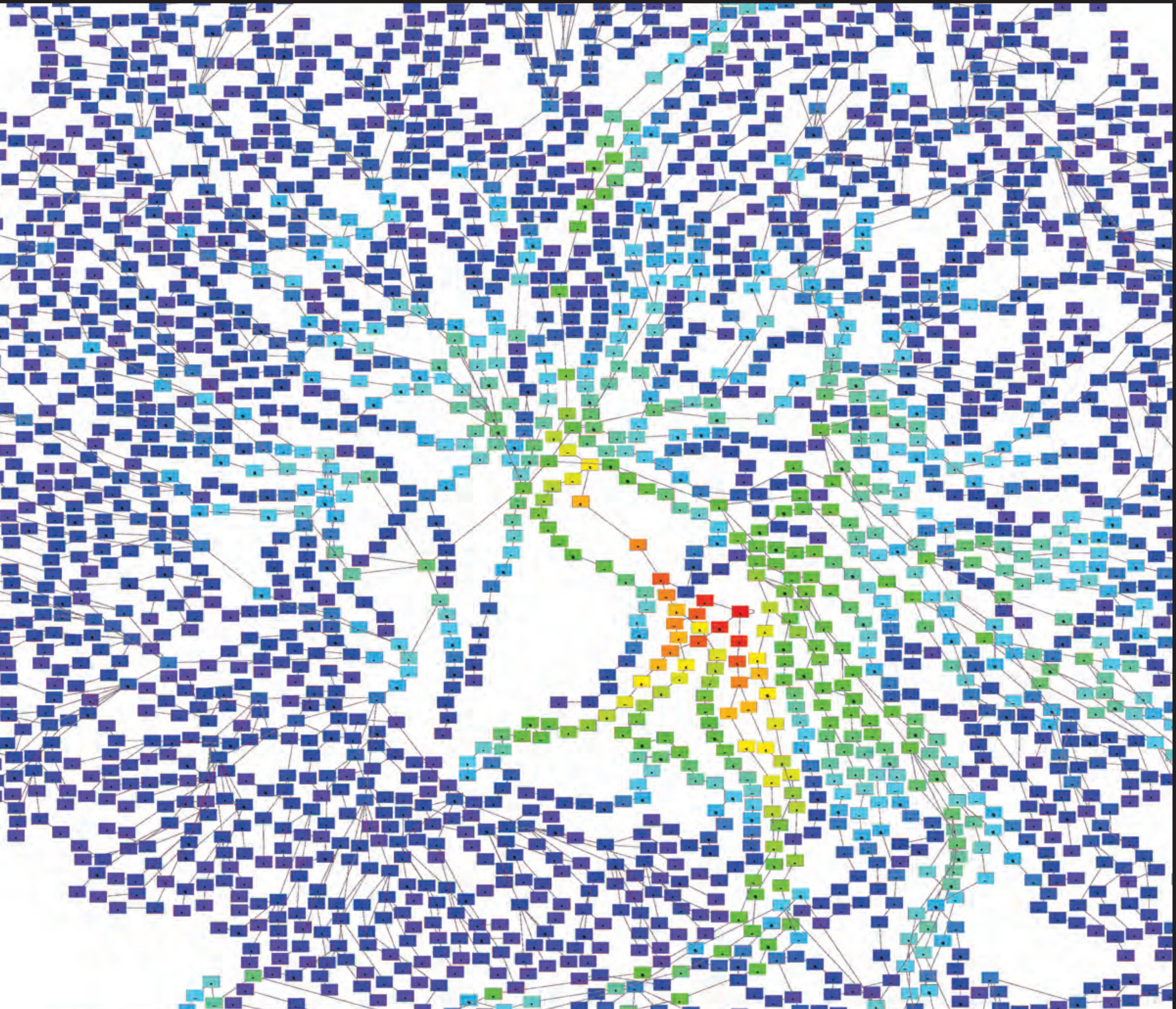
SIGEVOlution

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

Volume 7
Issue 2-3



in this issue

Editorial

Autumn has been good to us and we received enough material to publish a double issue packed with really good stuff! We have a tutorial on ExSTraCS, a new model of learning classifier system, developed by Ryan J. Urbanowicz for complex classification problems like the ones found in bioinformatics and epidemiology. We have a paper by Ricky Vesel who uses genetic algorithms in Race Optimal, a project aimed at creating an effective and fully automated process for racing line optimization to apply to all the circuits around the world. We also have a paper by Alwyn V. Husselmann about an approach to visualize populations of Karva expressions, also used for the cover of the newsletter. Carola Doerr and Gabriela Ochoa tell us about the 2014 Women @ GECCO workshop while Nadia Alshahwan reports on the 2014 Symposium on Search-Based Software Engineering, which was held in the beautiful Fortaleza, Brazil. At the end, there are the summaries of two new PhD theses available. One by Jacob Schrum (*Evolving Multimodal Behavior Through Modular Multiobjective Neuroevolution*), one by Alwyn V. Husselmann (*Data-parallel Structural Optimisation in Agent-based Models*). That's a lot! We hope Winter will be even better and we look forward to receiving more contributions to complete the last issue of this volume. So if you have something interesting for the newsletter, just drop an email!

Do you realize that the deadline to submit your paper to the next GECCO is less than three months away? Yup! It is time to start the engine, run those experiments left behind, and let the CPU/GPU fans roar until February 4! This year there will be no deadline extension, pretty scary huh? You surely don't want to miss the next GECCO in Madrid.

This issue was possible only with the help of Ryan J. Urbanowicz, Ricky Vesel, Alwyn V. Husselmann, Carola Doerr, Gabriela Ochoa, Nadia Alshahwan, William B. Langdon, Daniele Loiacono, Cristiana Bolchini, Viola Schiaffonati, and Francesco Amigoni.

Pier Luca
November 20, 2014

[illegible]

SIGEVOlution Volume 7, Issue 2-3

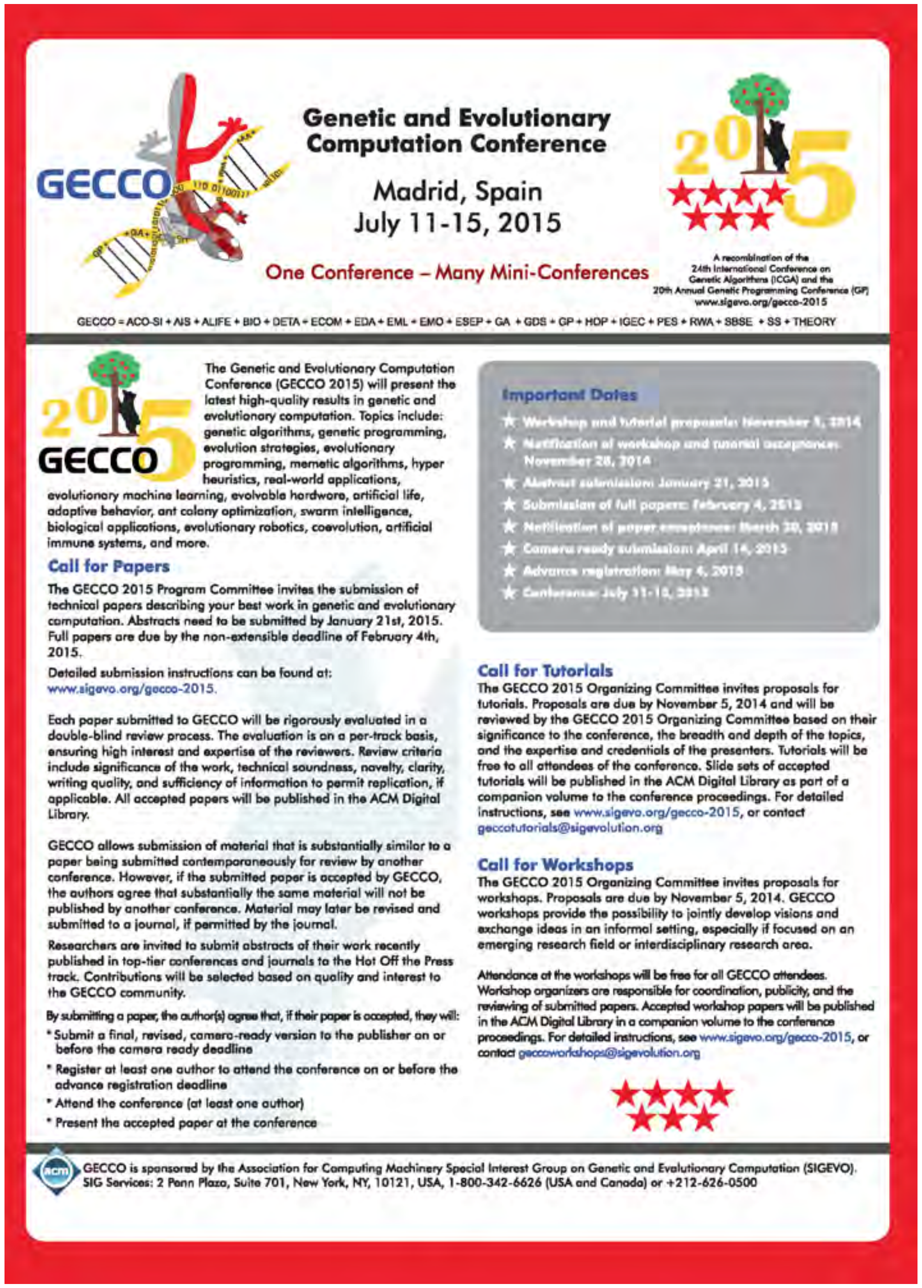
Newsletter of the ACM Special Interest Group on Genetic and Evolutionary Computation.

Contributors to this Issue

Ryan J. Urbanowicz
Ricky Vesel
Alwyn V. Husselmann
Carola Doerr
Gabriela Ochoa
Nadia Alshahwan

Contents

ISSN: 1931-8499



GECCO

Genetic and Evolutionary Computation Conference

Madrid, Spain
July 11-15, 2015

One Conference – Many Mini-Conferences

A recombination of the
24th International Conference on
Genetic Algorithms (ICGA) and the
20th Annual Genetic Programming Conference (GP)
www.sigevo.org/gecco-2015

GECCO = ACO-SI + AIS + ALIFE + BIO + DETA + ECOM + EDA + EML + EMO + ESEP + GA + GDS + GP + HOP + IGEC + PES + RWA + SBSE + SS + THEORY

The Genetic and Evolutionary Computation Conference (GECCO 2015) will present the latest high-quality results in genetic and evolutionary computation. Topics include: genetic algorithms, genetic programming, evolution strategies, evolutionary programming, memetic algorithms, hyper heuristics, real-world applications, evolutionary machine learning, evolvable hardware, artificial life, adaptive behavior, ant colony optimization, swarm intelligence, biological applications, evolutionary robotics, coevolution, artificial immune systems, and more.

Call for Papers

The GECCO 2015 Program Committee invites the submission of technical papers describing your best work in genetic and evolutionary computation. Abstracts need to be submitted by January 21st, 2015. Full papers are due by the non-extensible deadline of February 4th, 2015.

Detailed submission instructions can be found at: www.sigevo.org/gecco-2015.

Each paper submitted to GECCO will be rigorously evaluated in a double-blind review process. The evaluation is on a per-track basis, ensuring high interest and expertise of the reviewers. Review criteria include significance of the work, technical soundness, novelty, clarity, writing quality, and sufficiency of information to permit replication, if applicable. All accepted papers will be published in the ACM Digital Library.

GECCO allows submission of material that is substantially similar to a paper being submitted contemporaneously for review by another conference. However, if the submitted paper is accepted by GECCO, the authors agree that substantially the same material will not be published by another conference. Material may later be revised and submitted to a journal, if permitted by the journal.

Researchers are invited to submit abstracts of their work recently published in top-tier conferences and journals to the Hot Off the Press track. Contributions will be selected based on quality and interest to the GECCO community.

By submitting a paper, the author(s) agree that, if their paper is accepted, they will:

* Submit a final, revised, camera-ready version to the publisher on or before the camera ready deadline
* Register at least one author to attend the conference on or before the advance registration deadline
* Attend the conference (at least one author)
* Present the accepted paper at the conference

Important Dates

- ★ Workshop and tutorial proposals: November 5, 2014
- ★ Notification of workshop and tutorial acceptance: November 28, 2014
- ★ Abstract submission: January 21, 2015
- ★ Submission of full papers: February 4, 2015
- ★ Notification of paper acceptance: March 20, 2015
- ★ Camera ready submission: April 14, 2015
- ★ Advance registration: May 4, 2015
- ★ Conference: July 11-15, 2015

Call for Tutorials

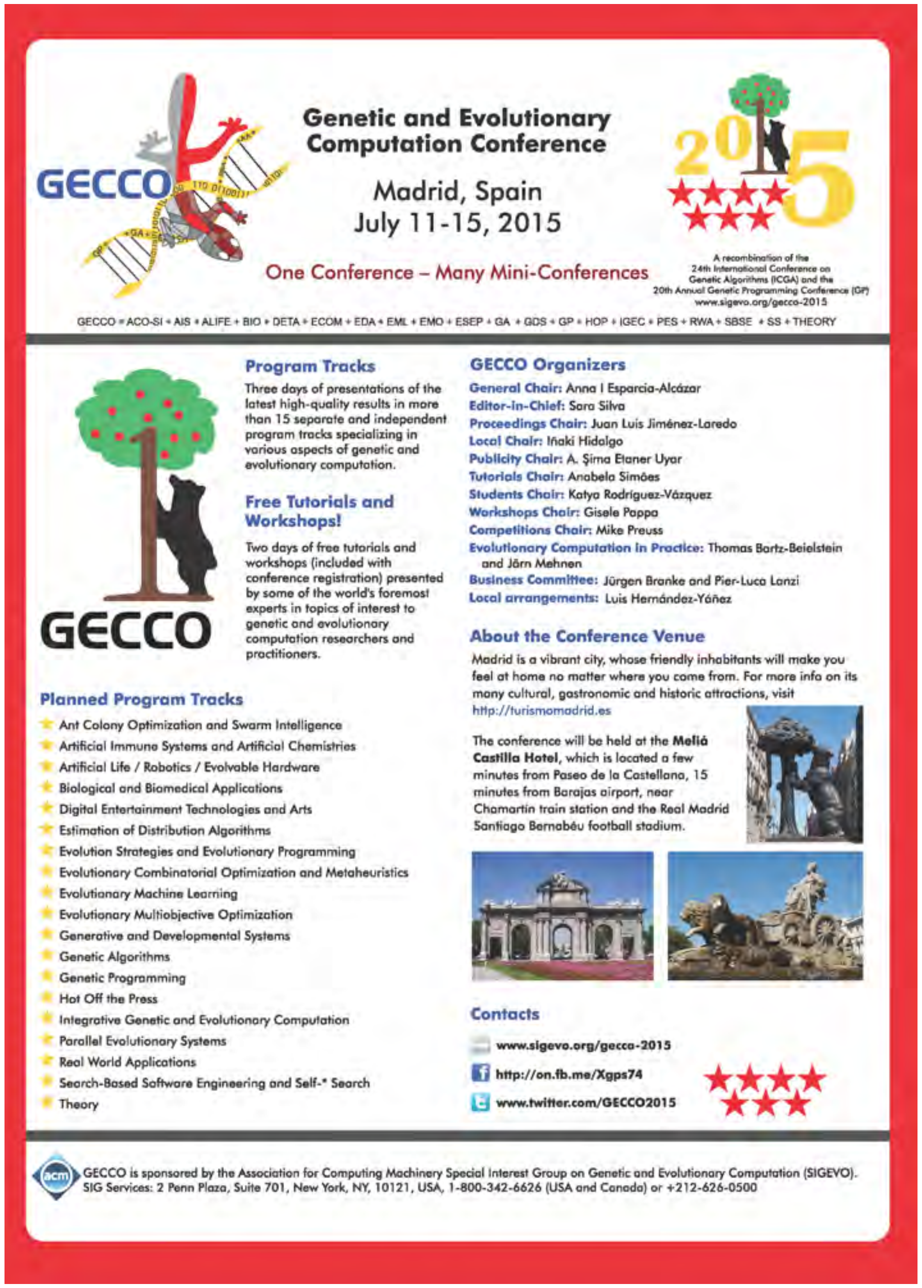
The GECCO 2015 Organizing Committee invites proposals for tutorials. Proposals are due by November 5, 2014 and will be reviewed by the GECCO 2015 Organizing Committee based on their significance to the conference, the breadth and depth of the topics, and the expertise and credentials of the presenters. Tutorials will be free to all attendees of the conference. Slide sets of accepted tutorials will be published in the ACM Digital Library as part of a companion volume to the conference proceedings. For detailed instructions, see www.sigevo.org/gecco-2015, or contact geccotutorials@sigevolution.org

Call for Workshops

The GECCO 2015 Organizing Committee invites proposals for workshops. Proposals are due by November 5, 2014. GECCO workshops provide the possibility to jointly develop visions and exchange ideas in an informal setting, especially if focused on an emerging research field or interdisciplinary research area.

Attendance at the workshops will be free for all GECCO attendees. Workshop organizers are responsible for coordination, publicity, and the reviewing of submitted papers. Accepted workshop papers will be published in the ACM Digital Library in a companion volume to the conference proceedings. For detailed instructions, see www.sigevo.org/gecco-2015, or contact geccoworkshops@sigevolution.org

GECCO is sponsored by the Association for Computing Machinery Special Interest Group on Genetic and Evolutionary Computation (SIGEVO). SIG Services: 2 Penn Plaza, Suite 701, New York, NY, 10121, USA, 1-800-342-6626 (USA and Canada) or +212-626-0500



GECCO

Genetic and Evolutionary Computation Conference

Madrid, Spain
July 11-15, 2015

One Conference – Many Mini-Conferences

A recombination of the
24th International Conference on
Genetic Algorithms (ICGA) and the
20th Annual Genetic Programming Conference (GP)
www.sigevo.org/gecco-2015

GECCO = ACO-SI + AIS + ALIFE + BIO + DETA + ECOM + EDA + EML + EMO + ESEP + GA + GDS + GP + HOP + IGEC + PES + RWA + SBSE + SS + THEORY

GECCO

Program Tracks

Three days of presentations of the latest high-quality results in more than 15 separate and independent program tracks specializing in various aspects of genetic and evolutionary computation.

Free Tutorials and Workshops!

Two days of free tutorials and workshops (included with conference registration) presented by some of the world's foremost experts in topics of interest to genetic and evolutionary computation researchers and practitioners.

Planned Program Tracks

- Ant Colony Optimization and Swarm Intelligence
- Artificial Immune Systems and Artificial Chemistries
- Artificial Life / Robotics / Evolvable Hardware
- Biological and Biomedical Applications
- Digital Entertainment Technologies and Arts
- Estimation of Distribution Algorithms
- Evolution Strategies and Evolutionary Programming
- Evolutionary Combinatorial Optimization and Metaheuristics
- Evolutionary Machine Learning
- Evolutionary Multiobjective Optimization
- Generative and Developmental Systems
- Genetic Algorithms
- Genetic Programming
- Hot Off the Press
- Integrative Genetic and Evolutionary Computation
- Parallel Evolutionary Systems
- Real World Applications
- Search-Based Software Engineering and Self-* Search
- Theory

GECCO Organizers

General Chair: Anna I Esparcia-Alcázar
Editor-in-Chief: Sara Silva
Proceedings Chair: Juan Luis Jiménez-Laredo
Local Chair: Iñaki Hidalgo
Publicity Chair: A. Şima Etaner Uyar
Tutorials Chair: Anabela Simões
Students Chair: Katya Rodríguez-Vázquez
Workshops Chair: Gisele Pappa
Competitions Chair: Mike Preuss
Evolutionary Computation in Practice: Thomas Bartz-Beielstein and Jörn Mehnen
Business Committee: Jürgen Branke and Pier-Luca Lanzi
Local arrangements: Luis Hernández-Yáñez

About the Conference Venue

Madrid is a vibrant city, whose friendly inhabitants will make you feel at home no matter where you come from. For more info on its many cultural, gastronomic and historic attractions, visit http://turismomadrid.es

The conference will be held at the **Meliá Castilla Hotel**, which is located a few minutes from Paseo de la Castellana, 15 minutes from Barajas airport, near Chamartín train station and the Real Madrid Santiago Bernabéu football stadium.

Contacts

www.sigevo.org/gecco-2015
http://on.fb.me/Xgps74
www.twitter.com/GECCO2015

GECCO is sponsored by the Association for Computing Machinery Special Interest Group on Genetic and Evolutionary Computation (SIGEVO). SIG Services: 2 Penn Plaza, Suite 701, New York, NY, 10121, USA, 1-800-342-6626 (USA and Canada) or +212-626-0500

ExSTraCS

Rule-Based Machine Learning Classification and Knowledge Discovery for Complex Problems

Ryan J. Urbanowicz — Department of Genetics, Dartmouth College, USA — ryan.j.urbanowicz@dartmouth.edu

Learning classifier systems (LCSs) are an advantageous, powerful, and flexible class of algorithms that have, to date, been underutilized largely due to the perception that they are difficult to apply, evaluate, and interpret. ExSTraCS is an **Ex**tended **S**upervised **Tra**cking and **C**lassifying **S**ystem based on the Michigan-Style LCS architecture [4]. It offers an accessible, user friendly LCS platform for supervised rule-based machine learning, classification, data mining, prediction, and knowledge discovery. ExSTraCS seeks to make no assumptions about the data, and is therefore model free and particularly well suited to complex problems that are multi-factorial, interacting (non-linear), heterogeneous, noisy, class imbalanced, multi-class, or larger-scale. ExSTraCS is written in Python, open source, well documented, and freely available at sourceforge.net.

Introduction

This article begins with a brief description of the ExSTraCS algorithm, it's unique features, and key abilities. The overall focus will be to highlight how ExSTraCS seeks to make the application of an LCS algorithm approachable. A simple example problem is used illustrate this.

The ExSTraCS algorithm is descended from a lineage of Michigan-style LCS algorithms founded on the architecture of XCS [9], the most successful and best-studied LCS algorithm to date. ExSTraCS is most closely related to UCS [2] which replaced XCS's reinforcement learning scheme with a supervised learning strategy to deal explicitly with single-step problems such as classification and data mining.

Michigan-style LCSs are rule-based machine learning algorithms that uniquely distribute learned patterns over a collaborative population of individually interpretable (IF:THEN) rules/classifiers. This makes them somewhat of ensemble learners all on their own. These *rule-based* algorithms combine the global search of evolutionary computing (i.e. a genetic algorithm) with the local optimization of reinforcement or supervised learning. They also apply iterative, rather than batch-wise learning, meaning that rules are evaluated and evolved one instance at a time. These characteristics make Michigan-style LCS's proficient at flexibly and adaptively capturing complex and diverse problem space niches such as those found in multi-class, latent-class, or heterogeneous problem domains. Michigan-style LCSs are also naturally multi-objective, evolving rules toward maximal accuracy and generality/simplicity. They can also offer transparency, in that individual rules can be unambiguously interpreted, and straightforward strategies have been developed to derive global knowledge from the rule population as a whole through the use of LCS rule-population metrics, significance testing, and visualizations [7]. In short, if you're dealing with a hard or distributed problem and interpretation is as important to you as predictive/classification performance, consider using an LCS.

Algorithm Overview

ExSTraCS was primarily developed for application to bioinformatic and epidemiological problems involving the detection, modeling, and characterization of predictive attributes associated with common complex diseases [4].

In these types of problems, associations can be very noisy, epistatic, heterogeneous, or involve an unknown number of predictive attributes within a larger set of potentially predictive attributes. The main goals in the design and ongoing development of ExSTraCS is to (1) allow data driven learning by minimizing data/model assumptions, (2) optimize supervised learning within the Michigan-style LCS architecture, (3) expand algorithm flexibility to accommodate the integration of different data types (e.g. discrete and continuous variables, missing data, balanced/imbalanced data, binary or multi-class endpoints), (4) improve scalability, and (5) improve factors contributing to ease of use including run parameter selection, documentation, output, interpretation, and run time efficiency. While developed with bioinformatics applications in mind, ExSTraCS would be well suited to any supervised learning problem, especially when complexity is an issue.

ExSTraCS brings together a number of recently developed LCS advancements designed to provide users with better performance, data flexibility, and new ways to get information out of LCS learning. All features are directly integrated into ExSTraCS and utilized by default, allowing the user to run the algorithm without having to worry about setting a large number of run parameters. Key features include; (1) Adaptive data management: which automatically differentiates discrete from continuous attributes in a dataset loaded by the user. (2) Mixed discrete-continuous rule representation: that allows the LCS to learn on datasets with both discrete and continuous attributes [1]. (3) Expert knowledge discovery and rule initialization: that pre-processes the dataset using one of four built-in attribute weighting algorithms, assigning scores to attributes based on their likelihood of being class-predictive, and applying these scores to a smart rule initialization (a.k.a. rule covering) [8]. (4) Attribute tracking: a mechanism that updates and stores attribute weights for each instance in the training dataset as a form of long-term memory. Post-training, these scores can be applied to characterize patterns of association such as heterogeneity within the dataset [6]. (5) Attribute feedback: a mechanism that utilizes attribute tracking scores in the genetic algorithm to probabilistically utilize attribute combinations as building blocks for new rules [6]. (6) Rule compaction: that post-processes the rule population, consolidating the rule population removing poor, redundant, or inexperienced rules with the goal of facilitating interpretation and knowledge discovery. Six built in rule-compaction strategies are available within ExSTraCS [3].

How It Works

The ExSTraCS algorithm is comprised of a set of interacting mechanisms working together to evolve and sustain an adapting set of accurate and general rules. Figure 1 gives a schematic overview of the ExSTraCS algorithm analysis pipeline. This pipeline operates in three major phases (A) data pre-processing, (B) algorithm learning/training, and ends with (C) rule population post-processing.

As summarized in [4], (A) ExSTraCS will accept a finite training dataset and an optional testing dataset in order to evaluate the predictive performance of the algorithm. Adaptive data management initially determines and stores key characteristics of this dataset for use during learning iterations. Expert knowledge (EK) attribute weights are loaded or discovered from the dataset. (B) The core ExSTraCS algorithm repeats the following 10 steps up to a maximum number of learning iterations: (1) A single training instance is taken from the dataset without replacement. (2) The training instance is passed to a population [P] of rules that is initially empty. A rule is represented as a simple IF/THEN statement comprised of a condition (i.e. specified attributes with corresponding states), and the class prediction. (3) A match set [M] is formed, that includes any rule in [P] that has a condition matching the training instance. (4) [M] is divided into a correct set [C] and an incorrect set [I] based on whether each rule specified the correct or incorrect class. (5) If, after steps 3 and 4, [C] is empty (i.e. no correct rules were found), covering applies EK to intelligently generate a matching and 'correct' rule added to [M] and [C]. (6) For every rule in [P], a number of parameters are maintained and updated throughout the learning process such as: numerosity (the number of copies of a given rule in [P]), rule accuracy, which is the proportion of times that a rule has been in a [C] divided by the times it has been in a [M]; and rule fitness, which is a function of rule accuracy and the parameter nu. Rule parameters are updated for rules within [C] and [I]. (7) Subsumption, a rule generalization mechanism is applied to [C] [9]. A similar subsumption mechanism is also applied to new rules generated by the genetic algorithm (GA). (8) Rules in [C] are used to update attribute tracking scores for the current training instance. (9) The GA uses tournament selection to pick two parent rules from [C] based on fitness and generates two offspring rules which are added to [P]. The GA applies two well known discovery operators: crossover and mutation. All rules in [C] and [I] are returned to [P].

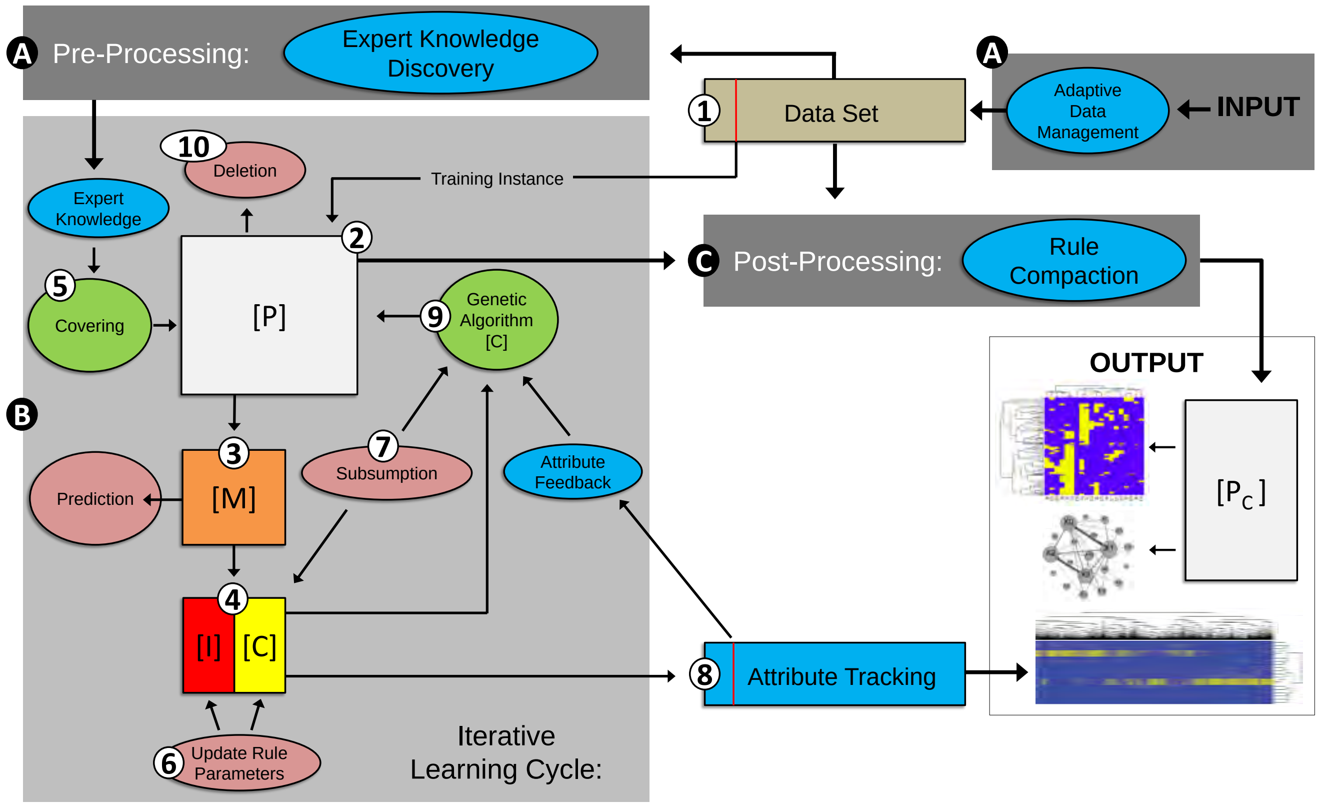


Fig. 1: **ExSTraCS Schematic:** Ovals are mechanisms, bordered squares are sets of either data or rules, green = rule discovery mechanism, pink = traditional LCS mechanism, and blue = mechanisms unique to ExSTraCS.

(10) Whenever the size of [P] is greater than the specified maximum, a deletion mechanism removes rules or rule copies from the population. For performance tracking and classification prediction, a prediction array is generated for the [M] formed from each training or testing instance. A class prediction is made by a fitness weighted vote of all rules within [M]. The class with the largest 'vote' is the predicted class. (C) After all learning iterations have completed, rule compaction is applied as a post-processing step to remove poor and/or redundant rules from [P] to yield [P_c].

ExSTraCS will yield up to five distinct output files after the final iteration, or any iteration at which a full evaluation is requested. These include (1) the population of rules collectively comprising the prediction 'model', (2) population statistics, summarizing major performance statistics including global training and testing accuracy of the rule population, (3) co-occurrence scores for the top specified pairs of attributes in the dataset, (4) attribute tracking scores for each instance in the dataset, and (5) predictions on testing data with respective class votes. This last output allows an evolved ExSTraCS rule population to be applied as a prediction machine. These outputs may be evaluated and visualized to facilitate knowledge discovery as described in [7].

Getting the Software

ExSTraCS 1.0 is available as open-source (GPL) code. It is a cross-platform program written entirely in Python 2.7. It is freely available for download from sourceforge.net/projects/exstracs. While this article focuses on ExSTraCS 1.0, we anticipate that ExSTraCS 2.0 will already be available at this same link. ExSTraCS 2.0 improves the scalability of ExSTraCS, reliably performing complex learning on datasets with upwards of 2000 attributes. Included with both versions of ExSTraCS is an exhaustive users guide that details command line operation, run parameters, and code modules. See [4] and [5] for research articles introducing ExSTraCS 1.0 and 2.0, respectively.

Minimum System Requirements

- Python 2.7 (www.python.org).
- 1 GHz processor
- 256 MB Ram

Example Problem

To demonstrate how ExSTraCS is set up, run, and its output analyzed, we will consider the multiplexer problem as an example. N-bit multiplexer problems are scalable supervised learning toy problems with Boolean attribute states, and a Boolean class. These problems involve attribute interactions as well as overlapping patterns of heterogeneity. Typically these problems do not include any noise, nor do they include any non-predictive attributes. N-bit multiplexer problems have been regularly applied to test and compare many different LCS algorithms, as well as other machine learning algorithms, particularly in the context of algorithm scalability.

For simplicity, we examine the 6-bit multiplexer problem illustrated in Figure 2. In any multiplexer problem, the initial address bits point to some register bit, and the value at this register bit will give the class of the instance. Essentially in the 6-bit multiplexer, 3 attributes states determine the class of a given instance. The dataset we use in this example to train ExSTraCS includes all 64 unique 6-bit strings as instances along with the correct corresponding class labels.

This problem can be completely and accurately captured by only 8 rules each specifying only three attributes. For instance a rule with condition (0,1,#,1,#,#) and class 1 would be optimal for making an accurate prediction on the instance given in Figure 2. The '#' symbol corresponds to a wild card or 'don't care', which indicates that the instance can have any value for that particular attribute (i.e. the rule is generalizing over those attributes). Another example of an optimal rule would be (1,1,#,#,#,0) with class 0. Here, the address bits '1,1' correspond to the fourth register bit which specifies a value of 0 for class.

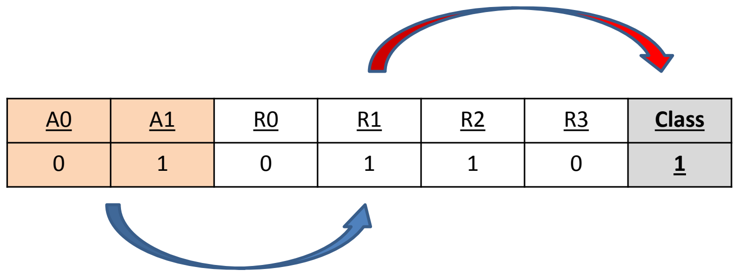


Fig. 2: Example 6-bit multiplexer training instance. The first two bits highlighted in orange are considered to be address bits and the remaining 4 bits are considered to be register bits. The class for this instance is highlighted in grey. Notice that in the multiplexer problem, the class is always equivalent to the Boolean value at the register bit specified by the address bits.

Running ExSTraCS

Preparing to run ExSTraCS from the command line requires only three things. First, make sure that Python 2.7 is installed on your computer. Second, make sure that you have a properly formatted training dataset file. Third, make sure that you have a properly formatted configuration file that specifies, at minimum, the set of 4 run parameters that have not been assigned a default value within ExSTraCS. Note, that in ExSTraCS 2.0 we have further simplified operation of the algorithm by requiring at minimum, only the path/name of the training dataset.

Data Format

Any training or testing dataset files should be formatted as a tab-delimited '.txt' file, where the first row includes column headers (i.e. identifiers for attributes, and the class variable). The header label for the class column is 'Class' by default, but this can be changed as a run parameter to match an existing class column label. Additionally, the class column can be located anywhere, as ExSTraCS identifies this column by the class label. Missing values in the dataset should have a standard, unique designation (we suggest 'NA' by default). The size of the data that ExSTraCS can handle, (i.e. the number of attributes and the number of instances) may be limited by the user's hardware specifications, when datasets get extremely large.

Configuration File

The configuration file in ExSTraCS can be used to specify all run parameters available in the software, including the path/name of the data files, traditional LCS learning parameters, and on/off switches for the many new features that have been incorporated into the ExSTraCS framework. While most run parameters in ExSTraCS include a default value, a handful of parameters must be specified within the configuration file. These include (1) `trainFile`, the path/filename for the training dataset, (2) `testFile`, the path/filename for the testing dataset (`None` can be specified if no testing data is available or desired to be included in the analysis), (3) `outFileName`, the path/base-filename for the five standard output files, and (4) `outEKFileName`, the path/base-filename for the EK file output by ExSTraCS during pre-processing. Names can be given without paths if respective files are in or will be placed in the working directory. Included with ExSTraCS is an example configuration file named `ExSTraCS_Configuration_File_Minimum.txt`. To get ExSTraCS running on the small simulated genetics training and testing datasets included with the software, leave this configuration file as is. To run ExSTraCS on the included datasets navigate from the command line to the folder where `ExSTraCS_Main.py` has been saved and type the following:

```
python ./ExSTraCS_Main.py
ExSTraCS_Configuration_File_Minimum.txt
```

This is the command for running ExSTraCS. Notice that the only argument required by ExSTraCS is the file path/name for a properly formatted configuration file.

Also included with ExSTraCS are two additional example configuration files. The first, `ExSTraCS_Configuration_File_Complete.txt` includes all available ExSTraCS run paramters and their default values. Some may find it convenient to use/edit this configuration file format so that they have greater control over algorithm parameters, and/or to have a record of parameters used in an analysis via a copy of a complete configuration file. Alternatively, the second example named `ExSTraCS_Configuration_File_Recommended.txt` includes both the minimum required run parameters, and parameters that (1) may dramatically impact performance given different dataset characteristics (2) are convenient for data formatting, or (3) give users access to key optional features. We expect that most users would find it convenient to use/edit this configuration file format.

Running the 6-bit Multiplexer Analysis

In order to run ExSTraCS on our example 6-bit multiplexer we will edit the `ExSTraCS_Configuration_File_Complete.txt` file. As we will largely rely on default values, we will only make a few key updates to this file (itemized below). See the ExSTraCS users guide (included with the software) for a detailed review of all run parameters, their function, and expected impact on algorithm performance.

- `trainFile` is assigned the name of the 6-bit multiplexer training data file `6Multiplexer_Data_Complete.txt`, saved in the working directory.
- `testFile` is assigned `None`.
- `outFileName` is arbitrarily assigned `ExampleRun`.
- `outEKFileName` is assigned `ExampleRun`.
- `learningIterations` is assigned `5000.10000`.
- `N`, which specifies the maximum rule population size, is assigned `500`.
- `p_spec`, which impacts the proportion of specified attributes in newly covered rules, is kept at the default of `0.5`.
- `nu`, which dictates the importance of high accuracy in the calculation of rule fitness, is assigned `10`.

The last four parameters in this list are really the only run parameters that a typical user might have to adjust when running ExSTraCS. We will discuss them in a bit more detail in a moment. First, in order to run this 6-bit multiplexer analysis simply type the ExSTraCS run command as before, this time specifying the edited configuration file.

```
python ./ExSTraCS_Main.py
ExSTraCS_Configuration_File_Complete.txt
```

Key Run Parameters

As alluded, there are only a handful of run parameters that have the potential to dramatically alter ExSTraCS performance, depending on the size and complexity of the dataset/problem. While users can certainly rely on default parameters to get started, we quickly review the impact of setting these key parameters below.

`learningIterations:` This parameter is used to specify two things. (1) The maximum number of learning iterations and (2) when to evaluate ExSTraCS at specified 'checkpoints'. The user can specify any number of learning checkpoints, however the iteration numbers should be increasing up to some maximum number of iterations, and individual values should be separated by a period. For larger datasets, or more complex problems, the user may wish to increase the number of learning iterations. Alternatively, if the user wants the algorithm to run for less time the number of learning iterations should be decreased. Note that for this 6-bit multiplexer problem we are running ExSTraCS for a total of 10,000 iterations, but pausing after only 5000 iterations for a complete evaluation of the rule population.

`N:` This parameter specifies the maximum population size that ExSTraCS is allowed to reach before the deletion mechanism turns on and maintains this maximum number. The value of N can have a dramatic influence on ExSTraCS performance. If N is too small, ExSTraCS can't properly explore a given search space, or maintain 'good' rules. If N is too large, ExSTraCS will take longer to run, and the resulting rule population will likely be much bigger than necessary, and potentially harder to interpret.

`p_spec:` This parameter indicates the probability that a given attribute will be specified (vs. generalized) during covering. Generalizing an attribute is equivalent to adding '#' in traditional LCS rule representations. Setting this value appropriately can play a critical role in successful LCS learning. By default, ExSTraCS uses a default value of 0.5. In datasets with a small number of attributes (i.e. $<= 20$) it is fine to leave this probability set to the 'high' default value of 0.5. Roughly speaking, as the number of attributes in the dataset increases, the value of p_spec should decrease. If p_spec is set too high for a given dataset, wildly over specific rules will be generated in covering, that have little to no chance of generalizing to other instances. On the other hand if p_spec is set too low covering may not initially 'cover' much of the search space, leading to lower initial rule diversity and a reduced chance that the best solution will be found. Ideally we would want to set p_spec to be equal to or just larger than the optimal average rule specificity required for the problem domain at hand. In most real-world problems this knowledge would not be available. New features in ExSTraCS 2.0 eliminates the need for this parameter all together, and greatly improves algorithm scalability.

`nu:` This parameter specifies the power to which accuracy is raised in calculating rule fitness. By default, ExSTraCS employs a default value of 1 which we have found to be most effective on noisy problems/data. Traditionally, LCS algorithms have employed a value of 10, which is particularly effective in 'clean' problems with little to no noise (i.e. it is possible to achieve a testing accuracy of 100%, or close to this value). In this 6-bit multiplexer problem we have set this parameter to 10, as is applied in most LCS algorithms.

ExSTraCS Output

ExSTraCS outputs a handful of files upon completion. These outputs can be analyzed to evaluate performance and provide a window inside the rule population and the patterns of association captured within. Using the 6-bit multiplexer example we will explore the ExSTraCS output and see how a solution distributed over a rule population can be interpretable. We examine each output file separately below.

Population Statistics

Many users may simply be interested in the training and/or testing accuracy of the evolved rule population. This is the classification or prediction accuracy of the solution.

The population statistics file (extension '[Iteration]_PopStats.txt') summarizes global performance of ExSTraCS, and provides global statistics to help characterize which attributes were found to be most important to making accurate predictions. Most importantly this file provides training accuracy, testing accuracy, global run time, coverage (i.e. the proportion of instance in the dataset upon which a prediction could be made) and the number of unique rules in the final population (i.e. macro population size). After 10,000 learning iterations ExSTraCS yields a training accuracy of 1.0, a run time of 0.07 minutes, a coverage of 1.0, and a macro population size of 95. In other words, ExSTraCS has learned to predict all training instance classes with 100% accuracy. Inspecting the file for only 5,000 iterations demonstrates that ExSTraCS had obtained 100% accuracy within 0.04 minutes on this fairly simple toy problem.

Next, this file includes three summary statistics introduced in [7] that can be used for knowledge discovery to identify attributes that were of particular importance in making class predictions. These statistics include the specificity sum, the accuracy sum, and the attribute tracking global sum. For each statistic a sum is calculated for every attribute in the training data. Attributes that consistently have the highest sums for these three metrics are likely to be most important for making accurate predictions. For our 6-bit multiplexer example we observe that the two address bit attributes consistently yield the highest scores for all three metrics. This is clearly in-line with the multiplexer solution, that requires optimal rules within which the address bits are always specified, along with one other register bit. In problems where only a subset of attributes are predictive, these metrics can be applied as feature selection to identify attributes that the algorithm identified as important vs. attributes that could be ignored.

Statistical significance values (i.e. p-values) can be assigned to most statistics in this file through the use of permutation testing as described in [7].

Rule Population

For users interested in knowledge discovery, or using the evolved rule population as a prediction engine, this output file (extension [Iteration]_RulePop.txt) includes all information about individual rules evolved in the population. Essentially this file outputs the resulting solution or 'model', that is distributed over a population of simple IF:THEN rules.

Any rule can be translated into a human readable expression such as 'If A0 is 0 and A1 is 1 and R1 is 1, THEN Class = 1', which would be a rule describing the 6-bit multiplexer pattern seen in Figure 2. The rule population can be explored through manual rule inspection. For every rule in this file, a number of rule parameters are stored for each including the rule condition, which ExSTraCS breaks up into 'Specified' and 'Condition', or the attribute column location and the attribute state specified, respectively. This is part of ExSTraCS's more efficient and flexible rule representation [1]. Also, each rule stores a class prediction (referred to as 'Phenotype'), rule fitness, and rule numerosity. The most effective way to identify the most important rules in the population as part of manual interpretation, is to rank rules by decreasing numerosity. This can be accomplished quickly in software like Microsoft Excel. Rules with the largest numerosity are typically most important. Doing this for our 6-bit multiplexer problem we quickly observe that all 8 optimal rules have been identified within the 14 rules with the largest numerosities out of all 95 rules in the population. These optimal rules include:

- (1,1,#,#,#,0) $\rightarrow$ 0
- (0,1,#,0,#,#) $\rightarrow$ 0
- (1,0,#,#,1,#) $\rightarrow$ 1
- (1,0,#,#,0,#) $\rightarrow$ 0
- (1,1,#,#,#,1) $\rightarrow$ 1
- (0,1,#,1,#,#) $\rightarrow$ 1
- (0,0,0,#,#,#) $\rightarrow$ 0
- (0,0,1,#,#,#) $\rightarrow$ 1

The rule population can also be visualized as a heatmap as described in [7], clustering rules vs. attributes, to reveal attribute interactions, and potentially heterogeneous relationships captured by rules in [P]. This can be particularly useful in noisy problems, or in problems where only some attributes are predictive.

Co-occurrence

Similar to the attribute sum scores provided by the population statistics file, attribute co-occurence sums are given in the Co-occurrence file (extension '[Iteration]_CO.txt').

This file ranks top pairs of attributes that are co-specified in rules across [P]. Permutation testing can be also be applied as described in [7] to assign p-values to these sums. For our 6-bit multiplexer example, the top co-occuring attribute pair is A0 and A1, which captures the most important interacting attribute pair (i.e. the address bits that should be co-specified within optimal rules).

Attribute Tracking Scores

Another output that can be applied to knowledge discovery is the attribute tracking score file (extension '[Iteration]_AttTrack.txt'). As described in [6], hierarchical clustering can be performed on instances, and attributes within this file to identify groups of instances with similar patterns of attributes with high attribute tracking scores in order to identify potentially heterogeneous instance subgroups and better characterize relationships between attributes predictive of class. Similar to the rule population visualization, these scores can be visualized as a heatmap, wherein instances vs. attributes are illustrated, each arranged via hierarchical clustering.

Learning Tracking

Different from the previously described output files, the learning tracking file (extension 'LearnTrack.txt') is only output once per run of ExSTraCS on a given dataset. This file includes all of the estimated learning performance updates, output throughout ExSTraCS learning. This file can be used to graph learning progress over time. Examining this file for our 6-bit multiplexer problem, we observe that 100% training accuracy is achieved after about 2350 learning iterations.

Conclusions

ExSTraCS has adapted and expanded the Michigan-style LCS algorithm to the needs of investigators dealing with real-world supervised learning problems. This software seeks to take the flexible and powerful LCS algorithm framework and make it user-friendly, and transparent such that classification and data mining on complex domains can be performed with confidence and understanding.

In this article we have used a somewhat simple problem to illustrate how the training, evaluation, and interpretation of an LCS algorithm has been made more approachable in the context of the ExSTraCS software. However, this example only begins to scratch the surface in terms of what ExSTraCS is capable of, and how it's output may be utilized as a competitive prediction machine, and a rich resource for knowledge discovery. ExSTraCS is under active development and improvement, to further enhance performance, interpretability, and flexibility to different types of data and analysis. ExSTraCS 2.0 adds a rule specification limit that dramatically improves ExSTraCS scalability and eliminates one of the major LCS run parameters. Over the next year we also plan to (1) add the ability for ExSTraCS to learn on data with continuous endpoints, referred to as quantitative trait analysis, (2) further improve overall learning speed and performance, and (4) provide a graphical user interface (GUI) for ExSTraCS to facilitate use and incorporate live learning visualizations. We encourage user feedback, application to new problem domains, collaborative or independent development of ExSTraCS, and competitive comparison of this new LCS framework to other cutting edge machine learning strategies.

References

[1] Jaume Bacardit and Natalio Krasnogor. A mixed discrete-continuous attribute list representation for large scale classification domains. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 1155–1162. ACM, 2009.

[2] Ester Bernadó-Mansilla and Josep M. Garrell-Guiu. Accuracy-based learning classifier system: models, analysis and applications to classification tasks. *Evolutionary Computation*, 11(3):209–238, 2003.

[3] Jie Tan, Jason Moore, and Ryan Urbanowicz. Rapid rule compaction strategies for global knowledge discovery in a supervised learning classifier system. In *Advances in Artificial Life, ECAL*, volume 12, pages 110–117, 2013.

[4] Ryan Urbanowicz, Gediminas Bertasius, and Jason Moore. An extended Michigan-style learning classifier system for flexible supervised learning, classification, and data mining. *Parallel Problem Solving from Nature–PPSN XIII*, page In press, 2014.

[5] Ryan Urbanowicz and Jason Moore. Addressing scalability with a rule specificity limit in a Michigan-style supervised learnign classifier system for classification, prediction, and knowledge discovery. *In Review*, 2014.

[6] Ryan Urbanowicz, Ambrose Granizo-Mackenzie, and Jason Moore. Instance-linked attribute tracking and feedback for Michigan-style supervised learning classifier systems. In *Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference*, pages 927–934. ACM, 2012.

[7] Ryan J Urbanowicz, Ambrose Granizo-Mackenzie, and Jason H Moore. An analysis pipeline with statistical and visualization-guided knowledge discovery for Michigan-style learning classifier systems. *Computational Intelligence Magazine, IEEE*, 7(4):35–45, 2012.

[8] Ryan J Urbanowicz, Delaney Granizo-Mackenzie, and Jason H Moore. Using expert knowledge to guide covering and mutation in a Michigan-style learning classifier system to detect epistasis and heterogeneity. In *Parallel Problem Solving from Nature-PPSN XII*, pages 266–275. Springer, 2012.

[9] Stewart W. Wilson. Classifier fitness based on accuracy. *Evolutionary computation*, 3(2):149–175, 1995.

About the author

Ryan Urbanowicz is a post-doctoral research associate at Dartmouth College where he received his PhD in genetics in 2012. He also holds a B.Sc. and M.Eng in Biological Engineering from Cornell University. His main research interests include bioinformatics, data mining, machine learning, evolutionary algorithms, epidemiology, and learning classifier systems. On the topic of learning classifier systems he has received two best paper awards, written a widely cited review of the field, and served as organizer for the international workshop for the past four years.

Homepage: www.ryanurbanowicz.com
Email: ryan.j.urbanowicz@dartmouth.edu

Racing Line Optimization @ Race Optimal

Ricky Vesel — Race Optimal (http://www.raceoptimal.com/) — rickyv@raceoptimal.com

Race Optimal (www.raceoptimal.com) is a project aimed at creating an effective and fully automated process for racing line optimization in order to produce a database of racing lines for circuits around the world. The ideal racing line is defined as the trajectory around a track that allows a given vehicle to traverse the circuit in the minimum time. In practice, it is an abstraction that varies with track, environmental conditions, vehicle type and condition, competitive traffic, and other factors. A certain extent of the talent possessed by the elite driver is the ability to perceive this optimal path and its variations, as well as to navigate it as quickly as possible. Until now, automated methods have struggled to match the performance of human-generated or human-guided racing lines [1]. This article describes the optimization engine and representation scheme used at Race Optimal to compute realistic and high performing racing lines for a wide variety of vehicle types.

Despite their abstract existence in the world of sport, there is significant use for concretely defined racing lines. Racing video games are quite popular, and require high quality racing lines in order to provide challenging AI competitors [1]. Racing lines have also a role in drivers' education, where they are used to develop visualization skills, as well as to provide a comparison of a driver's current approach with a potentially superior one. GPS data overlays onto track maps are currently used as driver aids and a logical extension of this approach is to display the generated optimal racing lines for comparison as well. An optimization process that encompasses vehicle characteristics can further be used for race preparation for example to choose a downforce configuration or transmission setup, as well as to quickly familiarize a driver with a new circuit.

When tackling the problem of racing line optimization, different simplifying assumptions have been applied. A 2D kinematic approach dictates that a vehicle's speed is governed by

$$v_{max} = \sqrt{\frac{r\mu(mg + F_{DF})}{m}} \tag{1}$$

where r is radius of curvature, μ is tire friction coefficient, m is vehicle mass, and F_{DF} is aerodynamic downforce. Maximizing the radius of curvature everywhere on the path will then maximize allowable vehicle speed everywhere. However, a shorter path, though having a smaller radius of curvature, may still take less time to traverse due to the shorter distance. Thus, one assumption applied in racing line optimization is that the optimal path will be a combination of the maximum curvature path (MCP) and the shortest distance path (SP). Producing the optimal linear combination of these paths is the accomplished by Braghin et al. [2]. This approach can be extended by dividing the track into subsections where the optimal tradeoffs between MCP and SP are optimized independently [3]. However, not all the approaches assume that the optimal racing line is a combination of the MCP and SP; in fact, other authors explored alternative methods of generating the racing line geometry, for example Euler spirals [4] or Bézier curves [5]. A simple point-by-point representation of the racing line may be also applied [3].

Once a racing line representation scheme is established, a fitness function is required to evaluate the candidate solutions. Fitness may simply be the measurement of path length or total curvature, but in order to capture the variation of the ideal racing line for different vehicles, some considerations about vehicle dynamics and power level are necessary. In [3], a robot driver is used in a high quality simulation to evaluate candidate racing lines. The present work also uses a computer simulation, but one that takes into account a much more limited model of vehicle dynamics in order to run as quickly as possible while still capturing the essence of several vehicle types.

A framework for racing line optimization should

1. Allow a high degree of geometric flexibility in racing line representation
2. Produce a racing line that appears realistic and smooth, without kinks or unnecessary undulations
3. Produce visually appealing and convincingly accurate racing lines for several vehicle types
4. Make full utilization of the available track surface, meeting all apexes and borders where necessary
5. Require minimal setup procedure allowing for the analysis of a large number of tracks and vehicles, with a fully automated solution process

By applying unique methods of geometric representation, optimization, and physics simulation, we believe these goals have been achieved. The methodology is described in the following sections.

Racing Line Representation

One of the biggest challenges was creating a racing line representation scheme that satisfies the above criteria, particularly (1), (2), and (4). Experimentation with different approaches revealed that all of the approaches mentioned above fail to meet criteria (1) alone. To elaborate, a high degree of flexibility means that the racing line should be able to take on any conceivable shape that is physically realistic for a vehicle traversing a racetrack. Use of a predefined geometric shape, such as an Euler spiral, while interesting, is obviously too restrictive to satisfy this aim.

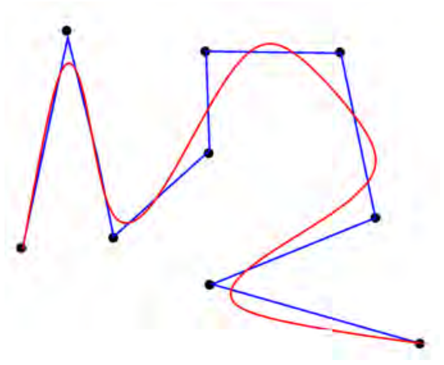


Fig. 1: Example of a Bézier curve, with control points (black) and the resulting curve (red) [5].

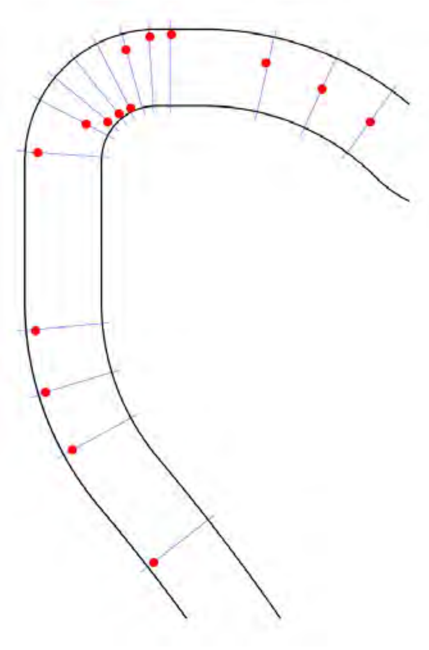


Fig. 2: Sample control point locations. The blue lines represent the possible locations of the attached control points (red). Adapted from [5].

This requirement should also obviously disallow a simple point-by-point construction of the racing line, with every point free to move independently; the unpredictable jumps from point to point would result in unphysical vehicle behavior and/or difficulty calculating the actual trajectory. It quickly becomes clear that a control point approach combined with some sort of smoothing function is necessary.

The authors in [5] utilize a Bézier curve (Figure 1), and this approach comes closest to satisfying our requirements. The control points, however, are restricted to the line segment perpendicular to the track on which they are placed, having a length of 120% of the track width, as shown in Figure 2. However, because Bézier curves typically do not intersect their control points, depending on the point density, there is no guarantee that this scheme will allow the racing line to reach the edges of the track, failing Criteria 5. Furthermore, limiting the control point positions to a line segment could result in an unacceptable likelihood that the fastest geometry might only be inadequately approximated by the limited set of potential Bézier curves available in this scheme, failing Criteria 1. Thus, in the current work, the control points are allowed to move more freely, within a circle of radius equal to four times the average track width, originating at initial user defined locations. An example is shown in Figure 3.

As part of fulfilling Criteria 3 to produce visually appealing racing lines, the line must smoothly connect to itself. This is important not only for the aesthetic quality, but because if that requirement is not enforced, the line could begin a circuit on a different path than with which it concludes, which is illogical in the context of lapping a racetrack. Initially the working solution was to forge a connection between the starting and ending points with an intermediate Bézier curve, which resulted in significant added complexity. Eventually, a more satisfactory solution was arrived upon, which utilizes a periodic smoothing spline [6].

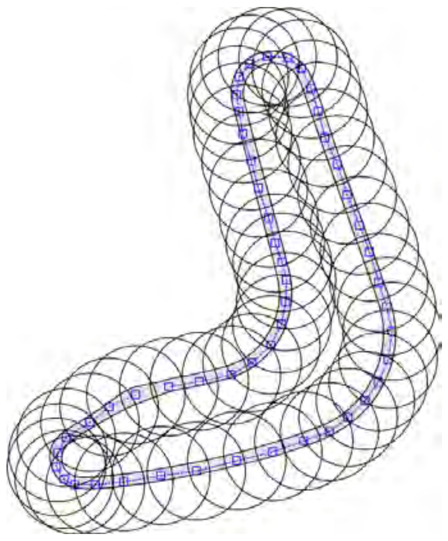


Fig. 3: Sample control point distribution (blue) and the circles representing their possible locations (black). Each control point is located at the center of its circle.

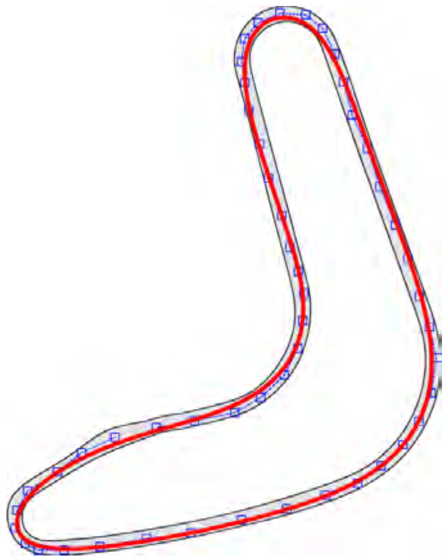


Fig. 4: Control points (blue) and the resulting periodic smoothing spline (red) using a smoothing factor of 0.25.

Periodic smoothing splines allow for a parameterized degree of smoothing, and automatically create a nicely connected curve which satisfies the following "periodic" end conditions over the interval $[x_1, x_n]$ [6]:

$$f(x_1) = f(x_n)$$

$$f'(x_1) = f'(x_n)$$

$$f''(x_1) = f''(x_n)$$

To obtain the racing line resulting from a set of control points, a parameter is introduced such that the x and y coordinate data are treated separately as a function of that parameter.

For example, control points $P_k = (x_k, y_k)$, $k \in \{1, \dots, n\}$, are separated into $P(x,k) = (t_k, x_k)$ and $P(y,k) = (t_k, y_k)$, with $t_k \in \{1, 2, \dots, n\}$. The coefficients of the periodic smoothing spline are then found for the x and y data separately, after which they are plotted together against t to form the smoothly connected racing line. The present work uses smoothing parameter $p = 0.25$. An example of the resulting (not optimized) racing line is shown in Figure 4.

Geometric Constraints

Because we use a control point scheme that allows for a highly variable racing line shape, some method must be employed to restrict the resulting line to the confines of the track. The first step in this process is to detect if the racing line intersects the start/finish line within the first 10% of its segments. Any line that fails this test is rejected automatically as invalid, thus ensuring that every candidate at least begins in an allowable position. The remaining task is to detect which points fall outside the bounds of the circuit, and apply an appropriate penalty. This was initially accomplished by counting the intersections between the borders and the racing line. Since it is known that the candidate begins inside the track, after one intersection it will be out of bounds, and will remain so until another intersection, and so on. Even after porting this functionality to C code, this process was unacceptably slow. A racing line typically has about 1000 points per mile, and track borders roughly the same point density. Evaluation often required tens of millions of intersection tests per solution.

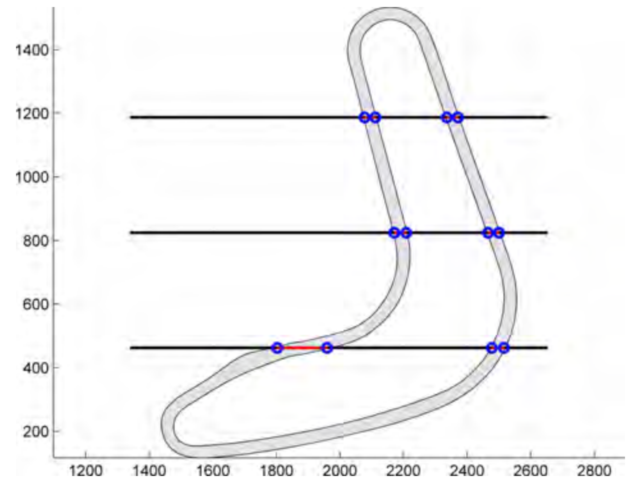


Fig. 5: Example of the horizontal slice method. Typically 10,000 or more horizontal lines are drawn at equal intervals (black) and the intersections with the borders are calculated (blue). The resulting valid x-intervals are shown in red. Three lines are shown here for demonstration.

A new method was devised to eliminate this bottleneck in the evaluation process, allowing for out-of-bounds points to be found very quickly, after a small initial cost at setup. In this method, several thousand horizontal lines are superimposed over the track borders, and the points of intersection with the borders are determined (Figure 5). A lookup table is produced that contains, for each line, the x-coordinates that contain the in-bounds sections of track. For a given point on the racing line, the table row with the closest y-coordinate can be quickly ascertained. That row can then be scanned to determine if the racing line point's x-coordinate is valid in any of the x-intervals for that row. Table 1 contains a sample of the lookup table corresponding to Figure 5.

Fitness Evaluation

Since this project requires the production of optimal racing lines for several vehicle types, vehicle simulation must obviously come into play to capture these differences. And because a flexible geometric scheme is utilized, there is no compelling reason to restrict oneself to SP or MCP solutions. This frees the process from as many assumptions about the optimal geometry as possible.

Tab. 1: Sample of in-bounds lookup table corresponding to the slices drawn in Figure 5.

y-coordinate	$x_{left,1}$	$x_{right,1}$	$x_{left,2}$	$x_{right,2}$
462.4	1802.6	1959.3	2477.1	2516.1
825.0	2170.7	2207.7	2465.3	2499.9
1187.6	2077.2	2111.1	2335.4	2370.9

Thus, vehicle simulation lap time is the core of the fitness function. The simulation process is outlined as follows:

1. Test for intersection with the start-finish line
2. Calculate the radius of curvature at every point, using

$$r = \left| \frac{(x'^2 + y'^2)^{3/2}}{x'y'' - y'x''} \right|$$

3. Calculate the maximum allowable velocity at every point, limiting to vehicle top speed
4. Beginning at the slowest point, iterate forward, limiting acceleration to that allowed by excess grip, vehicle power level, and aerodynamic drag
5. Iterate backward, limiting deceleration to that allowed by excess grip and aerodynamic drag
6. Enforce smooth input transitions
7. Calculate lap time
8. Count the out-of-bounds points and apply the appropriate penalty

Several items require further explication, beginning with (3). Maximum velocity is determined from the kinematic equation for centripetal force:

$$F_c = \frac{mv^2}{r} \tag{2}$$

This force is equal to the level of cornering force available,

$$F_c = \mu(mg + F_{DF}) \tag{3}$$

where F_{DF} is the aerodynamic downforce. From the well-known equation for aerodynamic lift, downforce can be modeled by $F_{DF} = 1/2\rho v^2 SC_L$. For a ground vehicle, however, the terms other than velocity can be grouped in a single parameter, k_{DF}, where $F_{DF} = k_{DF}v^2$. By setting Equation 2 equal to Equation 3 and plugging in this result, we can solve for velocity:

$$v = \sqrt{\frac{r\mu mg}{m - r\mu k_{DF}}}$$

Since the term in the denominator approaches zero as r_{max} approaches $m/(\mu k_{DF})$, this implies that for $r \geq r_{max}$, a vehicle with downforce can travel at an unlimited speed. Thus, when r is above r_{max}, the maximum velocity is taken to be the vehicle top speed.

Steps (4) and (5) translate the maximum allowable speed based on radius into realistic vehicle speed based on simplified vehicle dynamics. To limit the acceleration in the forward direction, the point at which speed is minimum is selected. The algorithm then looks to the velocity at the next point and computes the desired acceleration. The actual acceleration (and throttle input) is then restricted to that allowed by tire grip, engine power, and aerodynamic drag. After processing the entire path this way, the process is repeated in reverse, treating braking zones as acceleration zones in the backward direction. For braking, however, aerodynamic drag provides a contribution rather than a cost. Figure 7 demonstrates the output of steps 3-5 for the Mid-Ohio Sports Car Course, one of the circuits featured at RaceOptimal.com.

Initially this was the extent of the vehicle simulation. However, some Race Optimal users pointed out the unrealistically fast input transitions, as well as rapid input corrections mid-corner that would likely lead to instability. Since the vehicle simulations are published on Race Optimal in video form, these shortcomings needed to be addressed. A smoothing process was developed that limits the rate of input transition depending on the cornering load, where maximum cornering load requires the slowest rate of throttle or brake transition. The details of the smoothing algorithm are beyond the scope of this article, but an example of the results are shown in Figure 7 and Figure 8. Including this limit in the optimization also improved the shape of the final racing line, since the simulation no longer had the luxury of traversing a line in a way that requires instantaneous input corrections.

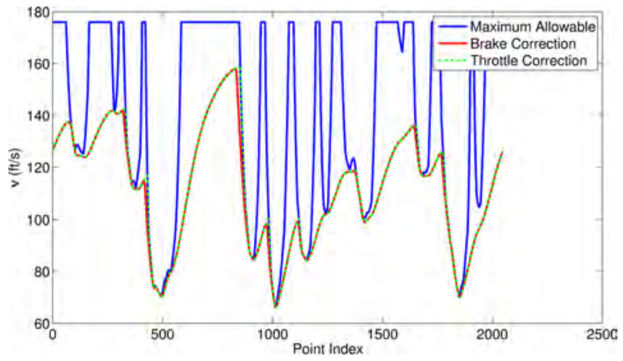


Fig. 6: Progression of simulated vehicle velocity, starting with the maximum allowable and showing corrections for acceleration and braking zones. Simulation is for a Mazda Miata at the Mid-Ohio Sports Car Course.

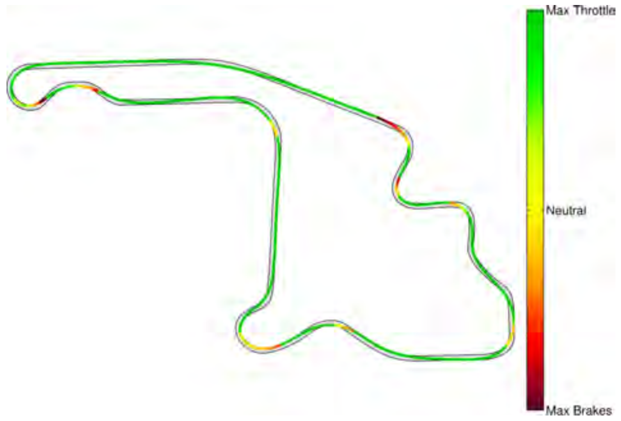


Fig. 7: Throttle and brake for Mazda Miata at Mid-Ohio Sports Car Course without smoothing correction.

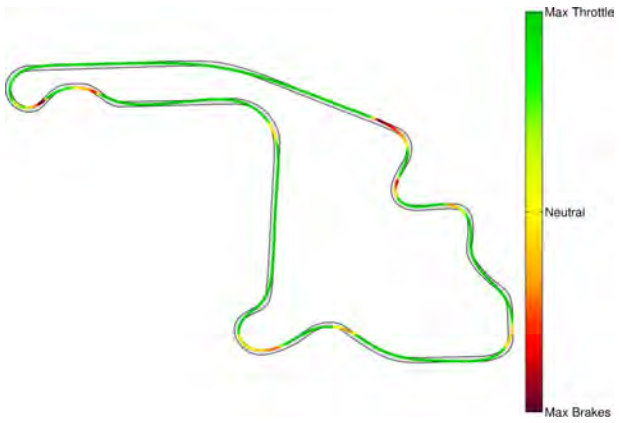


Fig. 8: Throttle and brake for Mazda Miata at Mid-Ohio Sports Car Course with smoothing correction.

Genetic Algorithm

We experimented with several approaches, including particle-swarm optimization and genetic algorithms with various parameters. The current approach is a standard genetic algorithm with a supplemental subprocesses called "healing" used to improve performance. The optimization uses binary encoding and a population size of 75 members, operating on a set of control point locations. The control point positions are calculated by an offset from their origin defined in radial coordinates, with 14 bits each encoding r and Θ. Selection from the parent population for breeding is made using the roulette wheel approach, where the likelihood of selection is proportional to candidate fitness. Bit crossover probability between mating genomes is 75%, and mutation probability varies with generation, but is at most 0.2%. Each generation, a child solution set is created equal in size to the parent generation. A simple replace-worst strategy is utilized, where the worst 90% of parent solutions are replaced by the best 90% of child solutions.

The maximum number of generations is set to 15000, but the optimization can end as early as 10000 if no substantial lap time improvement is being obtained. In order to combat premature convergence, after 2000 generations the population is re-randomized using the current best solution as the seed.

This approach provides generally good results, but not good enough to satisfy Criteria 3, which requires visually appealing and convincingly accurate racing lines. The optimization would frequently miss apexes by a small to medium distance, and would do this consistently at certain turns on some tracks, despite the fact that clipping the apex resulted in a quicker lap. Although the difference in lap time was rarely more than one or two tenths of a second, such errors are noticeable in the resulting racing lines, and undermine the credibility of the results. It was eventually determined that this problem was caused by the solutions becoming overly sensitive to out-of-bounds points at the entrances and exits of some turns. The racing line converged on the border at the entrance or exit before the apex, and by that time moving closer to the apex pushed the other sectors out of bounds. Since out-of-bounds points are penalized stiffly in order to prevent them from appearing in the final solution, the result was that the optimization was unable to meet the racing line with the apex of some turns.

In order to address this shortcoming, a “healing” sub-process was developed that is applied to a certain percentage of offspring solutions. To heal a racing line, control points closest to concentrations of out-of-bounds points are incrementally shifted toward the in-bounds direction in an attempt to bring the out-of-bound points back onto the track. This process involves recalculating the racing line many times for a given solution, and therefore increases the computational burden substantially. It was found that limiting the likelihood of healing was beneficial both to processing time and final solution quality, as healing every solution tended to interfere too much with the genetic algorithm itself. Although the optimization still occasionally produces a missed apex, the frequency of this occurring is greatly reduced with the incorporation of the healing sub-process.

Results

As an example, we applied our optimization scheme to the Mid-Ohio Sports Car Course. In particular, in addition to the normal simulation-based optimization, we also performed optimizations seeking the SP and MCP solutions—the latter maximizing the sum of curvature as the fitness function. Figure 9 shows the MCP and SP solutions compared with the simulation-based optimization in a section of track called the Keyhole. The black-filled area represents the possible linear combinations of the MCP and SP solutions. Figure 10 shows the same results, but for the section of track made up of Turns 6-10, a series of connected esses. It can be seen that over most of this section the MCP and SP solutions are very similar. The simulation-based racing line enters Turn 6 within the MCP-SP area, but thereafter follows a path that deviates substantially from what would be allowed by any linear combination of the two.

Lap times for the three optimizations are shown in Table 2. The simulation-based result achieved a lap time of 1’43.42. The National Auto Sport Association (NASA) currently lists the track record (eligible laps occur during a timed points race) for Spec Miata vehicles at Mid-Ohio as 1’45.655[1]. This is over two seconds slower than the optimal lap time achieved in the simulation; investigations are underway to determine if this discrepancy is a result of optimistic vehicle parameters or possibly other inaccuracies in the simulation. Nevertheless, the simulation-based result beat the MCP solution by over 4 seconds, and the SP line by over 5 seconds. However, no optimization of the tradeoff between MCP and SP paths was performed. The time elapsed through Turns 6-10 can be measured for a better comparison, since there is little variation between the MCP and SP lines, and the MCP line alone is a plausible route through that sequence. The sector times are shown in Table 2, where the simulation-based solution is 0.67 seconds quicker than the MCP.

Bearing in mind the fourth criteria, to capture the differences in racing line for different vehicles, the same section from Mid-Ohio is shown in Figure 11, via a screen capture from raceoptimal.com. The four vehicle types are the Mazda Miata, Porsche 911 GT3 RS 4.0, a generic motorcycle-type vehicle, and a theoretical F1 car.

[1] http://www.nasamidwest.com

The optimization clearly produces different paths for the four vehicles. The entire track must also be considered in order to assess the accomplishment of the criteria listed in the introduction, and this is shown in Figure 12 for the Mazda Miata. The result is a fluid racing line that touches the borders in the expected places, and takes a line very similar to what one can observe in professional onboard footage. Thus, the initial five criteria for an automated process of realistic racing line optimization have been satisfied.

Conclusions

A racing line optimization procedure has been developed that produces plausible real-world results that are visually pleasing and can capture the behavior of different vehicle types. The procedure does not rely on assumptions about the geometry of the ideal racing line. It is hoped that future authors will free themselves from the restrictions of predetermined geometry or racing line characterization when undertaking the challenge of racing line optimization.

As of writing, 62 circuits with 145 individual configurations have been analyzed on RaceOptimal.com, with unique racing lines and video simulations for each of the four vehicles. Thus, the usefulness of the present method as a means of developing a racing line database has been established.

Future Work

The current track geometry is modeled in two dimensions, meaning that elevation changes and track camber are not considered, accordingly, future work will include 3D considerations. Also, a more sophisticated vehicle model can be used in order to more accurately capture the interaction between the racing line and vehicle dynamics, as well as to include variable vehicle parameters, such as gearing, downforce configuration, and tire hardness in the optimization process.

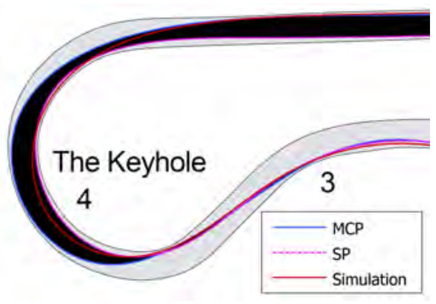


Fig. 9: The Keyhole section of Mid-Ohio Sports Car Course, showing the MCP, SP, and simulation based optimization results. The black region represents the track accessible by a linear combination of the MCP and SP solutions. The red line is the optimized path for a Mazda Miata. The racing line is the vehicle centerline, so a gap exists between the racing line and the track borders even at the nearest point.

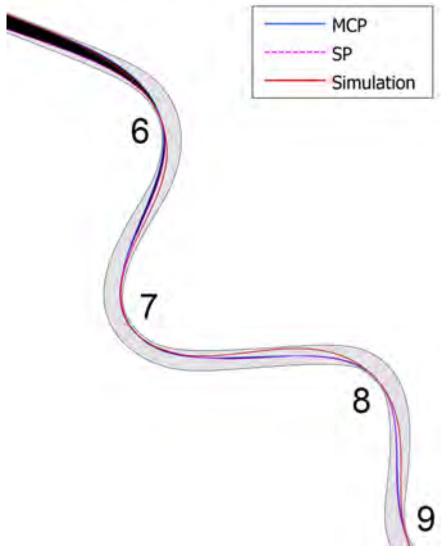


Fig. 10: Series of esses beginning with Madness from the Mid-Ohio Sports Car Course, showing the MCP, SP, and simulation based results. Clearly, the simulation based path requires track areas well outside the bounds of the MCP and SP solutions.

Tab. 2: Lap and sector times for the three optimizations.

Sector	MCP Time	SP Time	Simulation-Based Time
Full Track	1'47.67	1'48.86	1'43.42
"Madness" Section	19.21	19.57	18.54

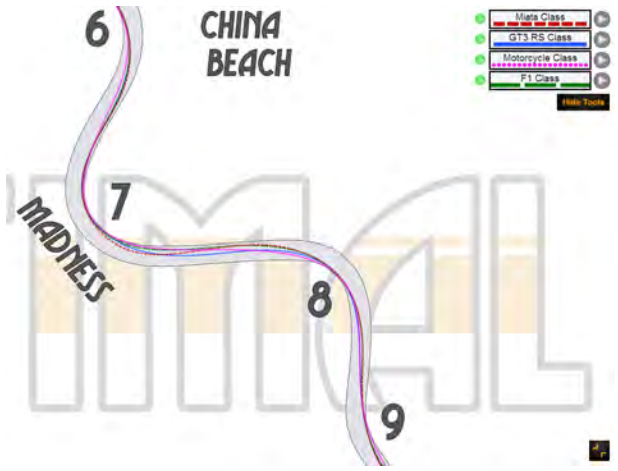


Fig. 11: Sample of the final racing line map for Turns 6-9 at Mid-Ohio Sports Car Course —- Club Circuit. The individual lines can be toggled on and off.

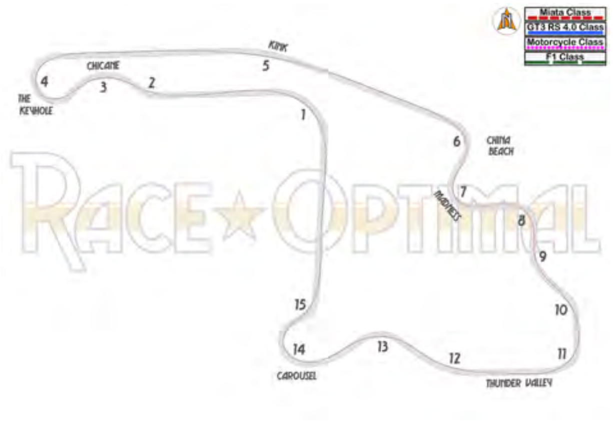


Fig. 12: Sample of the final racing line map for the Mazda Miata at Mid-Ohio.

References

[1] Stefano Lecchi. *Artificial intelligence in racing games*. In CIG'09: Proceedings of the 5th international conference on Computational Intelligence and Games, pages 1–1, Piscataway, NJ, USA, 2009. IEEE Press.

[2] F. Braghin, F. Cheli, S. Melzi, and E. Sabbioni. *Race driver model*. Comput. Struct., 86(13-14):1503–1516, 2008.

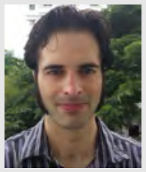
[3] L. Cardamone, D. Loiacono, P.L. Lanzi, and A.P. Bardelli. *Searching for the optimal racing line using genetic algorithms*. In Computational Intelligence and Games (CIG), 2010 IEEE Symposium on, pages 388–394, aug. 2010.

[4] Y. Xiong. *Racing Line Optimization*. Thesis. Massachusetts Institute of Technology, 2010.

[5] M. Botta, V. Gautieri, D. Loiacono, and P.L. Lanzi. *Evolving the Optimal Racing Line in a High-End Racing Game*. In Computational Intelligence and Games (CIG), 2012 IEEE Symposium on, pages 108–115, aug. 2012.

[6] N. Y. Graham. *Smoothing With Periodic Cubic Splines*. The Bell System Technical Journal 62.1 (1983): 101-10.

About the author



Ricky Vesel received his MS in Aerospace Engineering from The Ohio State University, where he specialized in wind turbine optimization. He became interested in applying similar techniques in searching for the ideal racing line, and solidified the concept over the course of a six week solo motorcycle trip. Race Optimal was created as a result. He currently lives in Vietnam and is an avid table tennis player.

Homepage: http://www.raceoptimal.com/
Email: rickyv@raceoptimal.com

K-Expressions Visualisation

Alwyn V. Husselmann — Institute of Natural & Mathematical Sciences, Massey University, Auckland — A.V.Husselmann@massey.ac.nz

Visualisation is important for gaining a qualitative understanding of how algorithms operate [1, 2]. Visualization techniques have been used to shed light on 3D voxel sets [3], vector fields [4], as well as spatial data structures [5], lattice gases [6], and several evolutionary algorithms.

In this brief article, we present a very fast and simple algorithm for visualising large population of lengthy program trees represented as a set of Karva expressions. The algorithm assumes that the population of candidates is a collection of *Karva*-expressions [16, 10] which can be individually expressed as the following genotype:

$$\star{-}{+}{+}\mathtt{babaadc}$$

The symbols $\star$, $+$, $-$ and $/$ are the typical mathematical operators taking two operands, and the symbols a, b, c and d are some arbitrary constants. The sequence of symbols represents a genotype whose interpretation (the phenotype) is shown in Fig. 1. The tree is constructed by taking the first symbol to be the root, and its arguments are filled from left to right, level by level, advancing through the expression one symbol at a time. Symbols that are not included in the phenotype (c and d) are known as “introns” [16], but are still considered in the genomes and by the genetic operators.

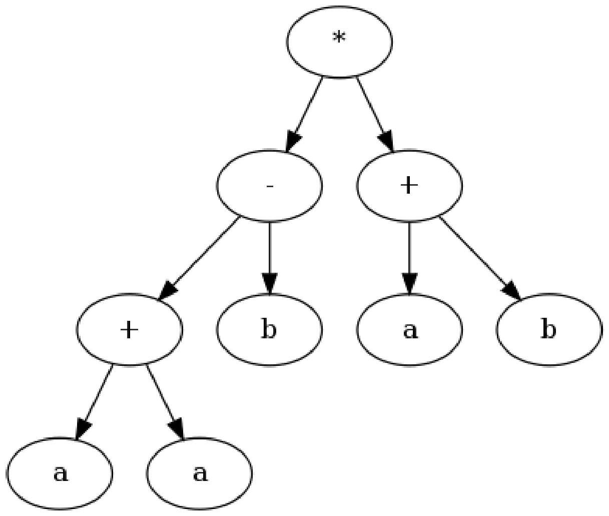


Fig. 1: The abstract syntax tree representing the *karva*-expression $\star{-}{+}{+}\mathtt{babaadc}$. The representation contains no d or c, as these do not make part of the phenotype.

GraphViz Visualisation

The algorithm is a geometric version of the Firefly Algorithm [12] designed for program space [2]. It is "geometric" in the sense of Moraglio's efforts towards geometrically unifying evolutionary algorithms in his PhD thesis [13].

A *GraphViz* tool known as *sfdp*, which uses an algorithm based on the work of Fruchterman and Reingold [14], was used to visualise a population of candidates. In essence, a population of expressions up to a length of 64 symbols were condensed into a tree, where chromosomes would share a graph node if they have it in common at the same expression symbol index. Nodes are then linked from root to leaf, which allows one to easily see where a chromosome deviates from others, and also what group of symbols is most likely to be seen near the top of the tree (where the function symbols are). All root symbols are linked to one manually placed root symbol, which is not present in any expression head. This is simply to ensure that there is one tree, and not four, as would be the case if there were four symbols possible in the head section of a set of *k*-expressions.

Renderings of a population of 1024 candidates (each with chromosome length 64) using the above method involved generated the images shown in Figures 2, 3 and 4 and reproduced with kind permission from Springer Science & Business Media; the images were originally published in [2].

Each of these images were rendered at different time intervals. The colours represent a distance from the root of the tree to the individual cells. Each cell contains a symbol of the expressions as indicated above. Figure 2 shows a freshly initialised population of individuals, whereas Figure 3 shows (generation 500) a population with more agreement over which symbols to use in which indices. Finally, Figure 4 shows generation 1000, where convergence has been achieved more or less. There is much more agreement and less divergent strands.

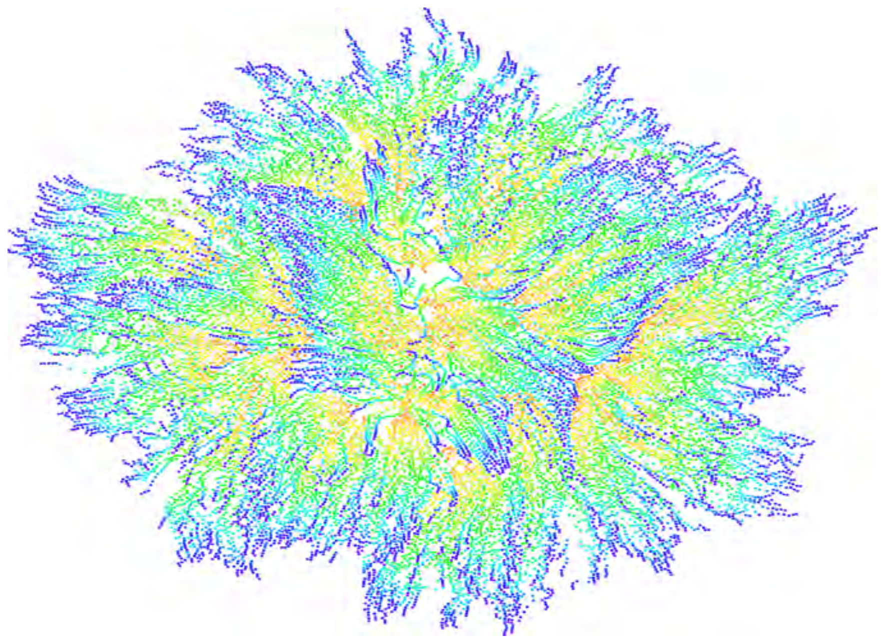


Fig. 2: A 2-dimensional visualisation of the very first generation of candidates. As can be seen in comparison to Figures 3 and 4, there is little agreement over which symbol to use for which expression symbol index.

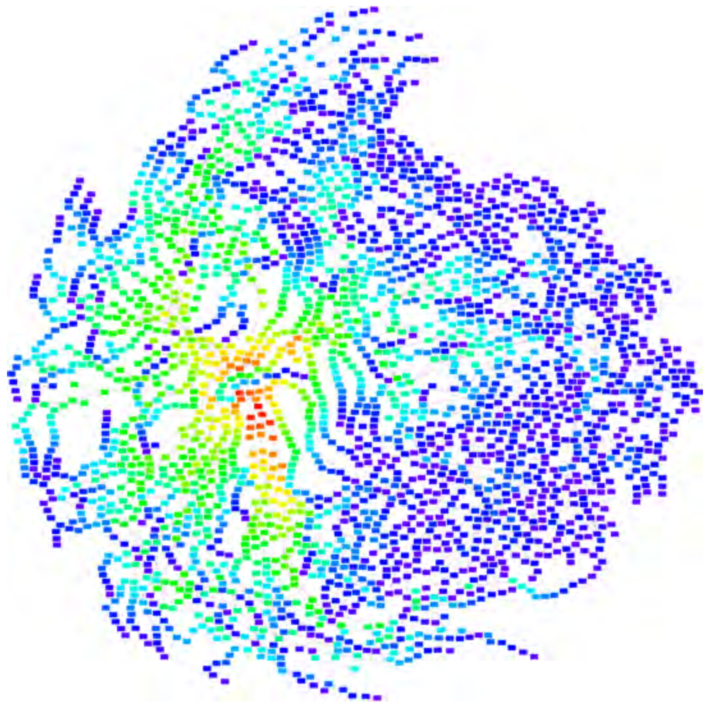


Fig. 3: Visualisation of generation 500 (symbolic regression population).

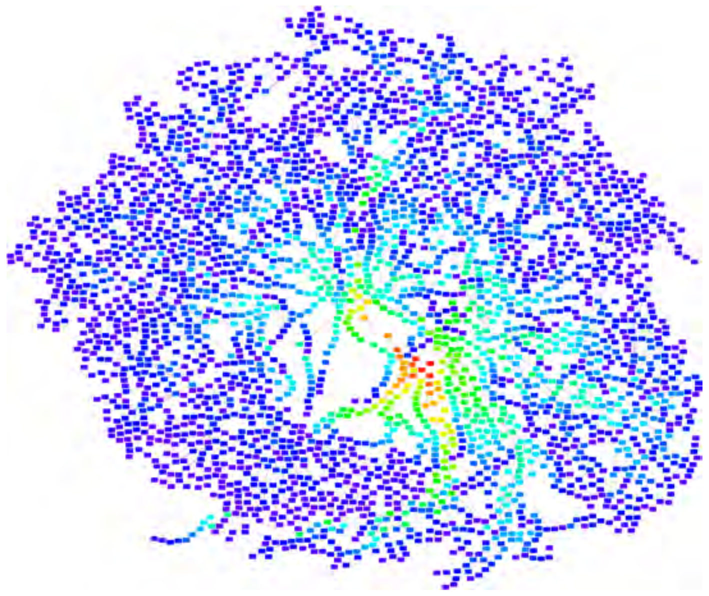


Fig. 4: Visualisation of generation 1000 (symbolic regression population).

References

[1] Husselmann, A.V.: Data-parallel Structural Optimisation in Agent-based Modelling. PhD thesis, Massey University (2014)

[2] Husselmann, A.V., Hawick, K.A.: Geometric firefly algorithms on graphical processing units. In: Cuckoo Search and Firefly Algorithm. Springer (2014) 245–269

[3] Hawick, K.A.: 3D visualisation of simulation model voxel hyperbricks and the cubes program. Technical Report CSTN-082, Computer Science, Massey University, Albany, North Shore 102-904, Auckland, New Zealand (October 2010)

[4] Husselmann, A.V., Hawick, K.A.: 3D vector-field data processing and visualisation on graphical processing units. In: Proc. Int. Conf. Signal and Image Processing (SIP 2012). Number CSTN-140, Honolulu, USA, IASTED (20-22 August 2012) 92–98

[5] Husselmann, A.V., Hawick, K.A.: Spatial data structures, sorting and GPU parallelism for situated-agent simulation and visualisation. In: Proc. Int. Conf. on Modelling, Simulation and Visualization Methods (MSV'12), Las Vegas, USA, CSREA (16-19 July 2012) 14–20

[6] Hawick, K.: Visualising multi-phase lattice gas fluid layering simulations. In: Proc. International Conference on Modeling, Simulation and Visualization Methods (MSV'11), Las Vegas, USA, CSREA (18-21 July 2011) 3–9

[7] McDermott, J. In: Visualising Evolutionary Search Spaces. The ACM Special Interest Group on Genetic and Evolutionary Computation (2014)

[8] Bille, P.: A survey on tree edit distance and related problems. Theoretical Computer Science **337**(1–3) (2005) 217 – 239

[9] Ferreira, C.: Gene expression programming: A new adaptive algorithm for solving problems. Complex Systems **13**(2) (2001) 87–129

[10] Ferreira, C.: Gene Expression Programming. 2 edn. Volume 21 of Studies in Computational Intelligence. Springer-Verlag, Berlin Heidelberg (2006) ISBN 3-540-32796-7.

[11] Pelikán, M., Kvasnicka, V., Pospichal, J., Koza, J.R., Deb, K., Dorigo, M., Fogel, D.B., Garzon, M., Iba, H., Riolo, R.L.: Read's linear codes and genetic programming. In: Genetic Programming 1997: Proceedings of the Second Annual Conference. (1997) 268

[12] Yang, X.S.: Firefly algorithms for multimodal optimization. In: Stochastic Algorithms: Foundations and Applications, SAGA. (2009)

[13] Moraglio, A.: Towards a Geometric Unification of Evolutionary Algorithms. PhD thesis, Computer Science and Electronic Engineering, University of Essex (2007)

[14] Fruchterman, T.M., Reingold, E.M.: Graph drawing by force-directed placement. Software: Practice and experience **21**(11) (1991) 1129–1164

About the author

Alwyn Husselmann recently completed his Ph.D. in Computer Science in 2014 at Massey University, which was supervised by Prof. Ken Hawick and Prof. Chris Scogings. His research shed light on evolutionary computation in the context of agent-based modeling, involving parallel computing and multi-stage programming. He is currently lecturing at the same institute, and investigating automatic platform code generation for combinatorial metaheuristic optimisers.

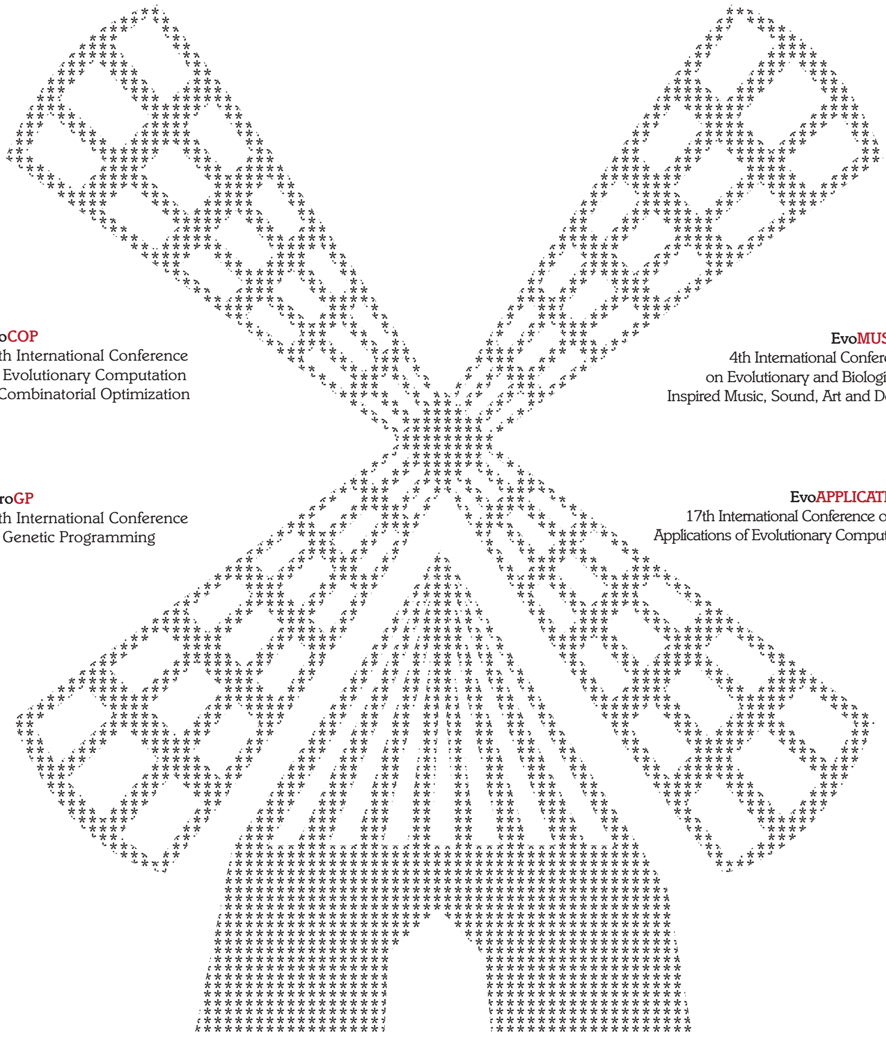
Homepage: http://www.massey.ac.nz/ avhussel/
Email: A.V.Husselmann@massey.ac.nz

evo*2015
8-10 April
Copenhagen - Denmark
www.evostar.org

Submission deadline
15 November 2014

EVO* coordinator
Jennifer Willies, Edinburgh Napier University, UK

Local Chairs
Paolo Burelli, Aalborg University
Julian Togelius, IT University of Copenhagen



EvoCOP
15th International Conference on Evolutionary Computation in Combinatorial Optimization

EvoMUSART
4th International Conference on Evolutionary and Biologically Inspired Music, Sound, Art and Design

EuroGP
18th International Conference on Genetic Programming

EvoAPPLICATIONS
17th International Conference on the Applications of Evolutionary Computation

Conferences Chairs: • Penousal Machado • Malcolm Heywood • James McDermott • Gabriela Ochoa • Francisco Chicano • Colin Johnson • Adrián Carballal • João Correia • Antonio Mora
Evo* Publicity Chairs: • Mauro Castelli • Pablo García Sánchez

2014 Women@GECCO Workshop

Carola Doerr and Gabriela Ochoa

The Women@GECCO workshop series started in 2013 with the aim of generating and supporting academic, professional, and social opportunities for women in evolutionary computation. The second edition, led by Una-May O'Reilly in collaboration with Anna Esparcia, Aniko Ekart, and Gabriela Ochoa, was held this year in Vancouver. Anne Auger and Carola Doerr ran an associated event: a work-life balance panel featuring the contribution of both female and male colleagues, who shared their experience in dividing their time between working hours and time for personal projects. Elena Popovici, the local chair, provided invaluable support and organised the provision of child-care facilities at the conference.

The workshop took place on Sunday July 12, at 10:40 – 12:30. The program featured an invited speaker, Prof. Jasmina Arivofic, who has been a faculty member of the Department of Economics, Simon Fraser University, B.C. Canada, since 1993. Jasmina climbed the ranks from assistant, to associate to full professor and serves as Director of Centre for Research on Adaptive Behavior in Economics. She conducts research in agent-based computational economics, with emphasis in understanding human learning and adaptive behavior using evolutionary simulations and genetic algorithms. Jasmina provided an engaging talk and spoke both about her work and personal journey in a male dominated field. She added to the workshop discussion sharing her experience with womens' groups in her field. In particular, she affirmed the benefits of mentorship, a topic that was discussed during the workshop. We have decided to run a mentorship program where interested women in the field (on all levels of an academic career) can subscribe. Carola and Emilia Tantar volunteered to organize this mentorship program. Interested members of the community should get in touch with them, both if you are interested in mentoring someone or if you want to be mentored by a more senior person. Carola and Emilia will do their best to find a good match between mentors and mentees.

During the rest of the workshop we also discussed challenges women face at conferences and the work place such as difficult interactions, being outnumbered, networking, and integrating work and life. This discussion was chaired by Una-May and Anne who shared some of their experiences.

The panel on work-life matters was held in the evening of Sunday, July 12, after the workshop and tutorial sessions. We are very happy to have had Jürgen Brancke, Emma Hart, Gabriela Ochoa, Una-May O'Reilly, Elena Popovici, Marc Schoenauer, and Nur Zincir-Heywood as panelists. The panelists shared different aspects of what makes a good work-life-balance for them. Elena pointed out that the topic is often discussed with a single focus on how to combine a successful academic career with having kids, but that for her it is also important to find ways to combine your job-related ambitions with your hobbies.

The panelists agree that it is both advantageous but at the same time a dangerous feature of our job that we can outsource many activities to our personal time, e.g., reading papers while waiting for the doctor's appointment. Several panelists report that reducing the working time to 80% or less as well as outsourcing of housekeeping work has helped them freeing up some time for non-work related projects. Marc shares some data on the "academic funnel": while the number of females in undergraduate courses is quite acceptable in many fields, the proportion of women drastically decreases with every step of an academic career (PhD, PostDoc, Professor level). The panel also discusses that there are huge differences between the different countries. We had a very lively discussion with several impulses from the audience so that we have decided to run a similar panel next year.

We count as accomplishments this year the design of new T-Shirts (as seen in the photo), the inclusion of our workshop description as part of the GECCO proceedings, and integration with the conference registration, the provision of childcare, the successful work-life panel, and the local invited speaker (serving as an important role model). We need to continue working to achieve a mentoring scheme, outreach activities (specially at high school level), and organising gatherings outside the realms of the GECCO conference.

An important outcome of the workshop was the distribution of work and roles to coordinate our future activities. The design and distribution of a survey to gather information related to women in evolutionary computation is in hands of Emma Hart, Amanda Whitlock, and Una-May; Leigh Sheneman volunteered to work with Una-May in the design of T-shirts; a mentoring scheme will be organised by Carola and Emilia; while Anna and Gabriela will organise an invited speaker for our forthcoming event in Madrid (GECCO 2015). We still need volunteers to work on fundraising and outreach activities, and to help in the organisation of the next work-life panel. If you are interested in supporting the workshop, please reach out to one of the organizers! In summary, we had an enjoyable, well-attended, and up lifting event, in which we welcomed new members, got to know each other better, and shared some of our experiences and concerns both among us and with our appreciated male colleagues.

Acknowledgments

Anne Auger and Carola Doerr would like to thank Timo Kötzing from the University of Jena for taking notes during the panel discussion.

Carola Doerr is a CNRS researcher at the University Pierre et Marie Curie, Paris, France. Her background is in mathematics (Diploma in 2007 from Kiel University, Germany) and Computer Science (PhD 2011 from the Max Planck Institute for Informatics and Saarland University, Saarbrücken, Germany). From Dec. 2007 to Nov. 2009, Carola has worked as a business consultant for McKinsey & Company, mainly in the area of network optimization. Carola's main research interest is in the theory of randomized algorithms, both in the design of efficient algorithms as well as in randomized query complexities ("black-box complexity"). She has contributed to the field of evolutionary computation also through results on the runtime analysis of evolutionary algorithms and drift analysis, as well as through the development of search heuristics for solving geometric discrepancy problems.

Homepage: http://www-ia.lip6.fr/ doerr/
Email: Carola.Doerr@mpi-inf.mpg.de

Gabriela Ochoa is a Lecturer in Computing Science at the University of Stirling, Scotland. Her research interests lie in the foundations and application of evolutionary algorithms and heuristic search methods, with emphasis on autonomous (self-*) search, hyper-heuristics, fitness landscape analysis, and applications to combinatorial optimisation, healthcare, and software engineering. She has published over 80 scholarly papers and serves various program committees. She is associate editor of Evolutionary Computation (MIT Press), was involved in founding the Self-* Search track in 2011, and served as the tutorial chair at GECCO in 2012, 2013. She proposed the first Cross-domain Heuristic Search Challenge (CHeSC 2011) and is involved in organising EvoCOP 2014, 2015, and FOGA 2015

Homepage: http://www.cs.stir.ac.uk/ goc/
Email: gabriela.ochoa@cs.stir.ac.uk

2014 Symposium on Search-Based Software Engineering - Event Report

Nadia Alshahwan

The city of Fortaleza, Brazil hosted the sixth edition of the Symposium on Search-Based Software Engineering (SBSE) between the 26th and 29th of August. Brazil was chosen to host the symposium because of its strong and growing SBSE community.

The organizers provided attendees with a truly amazing experience by holding the event in five-star resort and making every effort to make the event enjoyable, educational and interesting.

This year the symposium broke the record for highest number of submission, albeit by a small margin, which is impressive considering it is a standalone edition as opposed to the FSE-collocated edition of odd years.

Every day the symposium kicked off with a keynote or an invited talk from a leader in the field. Professor Mark Harman gave a talk on the first day that explored the future directions of SBSE in areas such as genetic improvements and non-functional enhancement.

The second day began with a keynote from Professor Mauro Pezze who discussed redundancy in software and its applications and how SBSE can be used to identify redundant functions. The keynote speaker on the third day was Professor Marc Schoenauer who gave a very interesting talk about programming by ranking.

The three days of the symposium were packed with interesting talks that discuss applications of SBSE in a wide range of software engineering activities such as system design, requirements engineering, testing and the production of documentation. One thing to note was that with all the temptations just a few steps away from the conference room (i.e. the sun and beach), it was clear that the attendees were more committed to learning more about the current trends in SBSE.

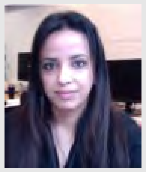
Overall the symposium was a fun and enlightening experience. I am sure many participants came back thinking about how to apply some of the ideas to their work. I know I spent the long flight back to London fantasizing about how genetic programming can produce customized similarity functions.

Challenge Track

The challenge track at SSBSE this year was probably the most exciting part of the symposium. Four very interesting papers were presented over two sessions. At the end of the second session the audience voted for the winner. The topics covered genetic improvements, fault prediction and GUI crashing. The winning paper was Babel Pidgin: SBSE Can Grow and Graft Entirely New Functionality into a Real World System by Mark Harman, Yue Jia, and William B. Langdon.

Graduate Track

Shadi Ghaith was the graduate track winner by receiving the highest total scores from the reviewers. This was only fitting as Shadi kindly volunteered to cover for various members of his team and ended up presenting three papers in addition to his own.



Nadia Alshahwan is research associate at the CREST Centre at University College London. She received her PhD from UCL in 2012. This year she served as the graduate track chair at SSBS. Her research interests include genetic programming and compression based similarity metrics.

Homepage: http://www0.cs.ucl.ac.uk/staff/N.Alshahwan/
Email: nadia.alshahwan@ucl.ac.uk

Evolving Multimodal Behavior Through Modular Multiobjective Neuroevolution

Dissertation by Jacob Schrum

Intelligent organisms do not simply perform one task, but exhibit multiple distinct modes of behavior. For instance, humans can swim, climb, write, solve problems, and play sports. To be fully autonomous and robust, it would be advantageous for artificial agents, both in physical and virtual worlds, to exhibit a similar diversity of behaviors. Artificial evolution, in particular neuroevolution [3, 4], is known to be capable of discovering complex agent behavior. This dissertation expands on existing neuroevolution methods, specifically NEAT (Neuro-Evolution of Augmenting Topologies [7]), to make the discovery of multiple modes of behavior possible. More specifically, it proposes four extensions: (1) multiobjective evolution, (2) sensors that are split up according to context, (3) modular neural network structures, and (4) fitness-based shaping. All of these technical contributions are incorporated into the software framework of Modular Multiobjective NEAT (MM-NEAT), which can be downloaded here.

First, multiobjective optimization is used to assure that different objectives associated with different modes of behavior each receive direct focus, using the Non-dominated Sorting Genetic Algorithm II [2].

Second, sensors are designed to allow multiple different interpretations of objects in the environment. If a sensor reading can be interpreted in several different ways depending on context, then it is a conflict sensor. Learning to use such general sensors is difficult. In contrast, split sensors assign individual sensor readings to different sensors depending on some human-specified context, such as whether the sensed agent is currently a threat or not. Split sensors can bias evolutionary search in a useful way, but a human designer may not know how to construct useful split sensors for an arbitrary domain. Therefore, ways of automatically splitting up a domain across different modes of behavior, without expert knowledge, are developed in the dissertation.

The third contribution allows evolution to discover what modes of behavior to use, and how to use them. Evolving networks are given ways of representing multiple policies explicitly via modular architectures. Specifically, each of several output modules defines a separate mode of behavior. Each module has a preference neuron, which indicates the network's preference for using a particular module on a given time step. By letting the module with the highest preference output control the agent, evolution can determine when to use a particular module in addition to the behavior of each module. Additionally, evolution can discover the number of modules to use via several variants of Module Mutation [6], a mutation operation that adds a new output module to the network. If a user wants to specify how to use specific network modules, a topology similar to that in Multitask Learning [1] can be used. However, the results show that in complex domains, it is better to let evolution discover its own module usage with the help of preference neurons.

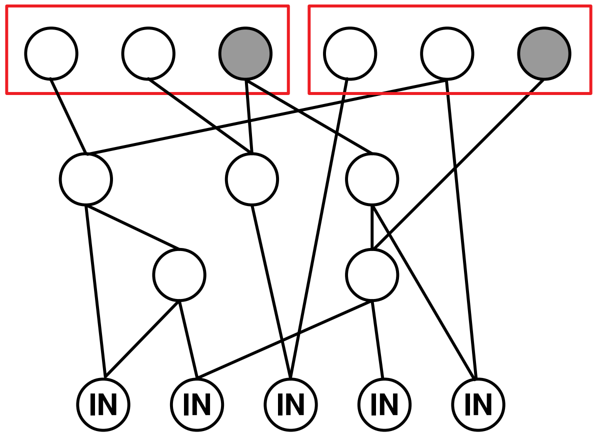


Fig. 1: Network with two modules, each consisting of two neurons to define behavior, and a gray preference neuron.

The fourth technical contribution of the dissertation is Targeting Unachieved Goals (TUG [5]), a way of dynamically adjusting which objectives are active in order to lead the population toward the most promising areas of the search space. TUG turns off objectives in which the average performance of the population surpasses numeric goals. Objectives can be turned back on if performance decreases again, but if all goals are achieved, then all objectives are reactivated, and all goal values are increased. As a result, the population is gradually pushed to reach better scores in all objectives.

In addition to the technical contributions, the dissertation provides a way of classifying domains based on how tasks are divided within them, and recommendations on which methods to apply in each type of domain. First, isolated tasks are completely independent from each other, but are performed serially by a single agent that must perform well in all of them. Second, interleaved tasks alternate back and forth in such a way that an agent can prepare for one task while in another. Interleaved tasks are clearly delineated, but not independent, because the actions in one task can affect the possible outcomes in other tasks. Third, in blended tasks there is no longer a clear border between tasks, resulting in blended situations where an agent must deal with multiple tasks simultaneously.

Example domains of each type are used to evaluate the methods developed for the dissertation, but the most challenging example domain is Ms. Pac-Man, a popular classic arcade game with blended tasks (Ms. Pac-Man must deal with both threat and edible ghosts at the same time). However, the methods developed for this dissertation allow Ms. Pac-Man to evolve multimodal behavior appropriate for handling these blended tasks. In fact, the best evolved task division is unexpected, because it dedicates a module to luring ghosts near power pills before eating them, which is a powerful play strategy that leads to high scores surpassing those of previously evaluated methods in this domain. Specifically, the multimodal networks evolved in this dissertation achieve higher scores than Ant Colony Optimization, Monte-Carlo Tree Search, and several forms of Genetic Programming.

In sum, the results in this dissertation demonstrate that complex multimodal behavior can be evolved automatically, resulting in robust and intelligent agents.

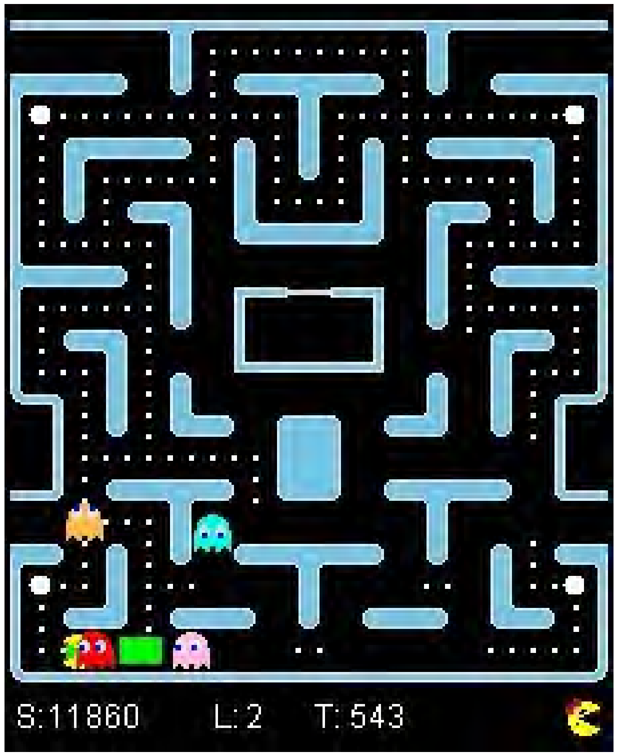


Fig. 2: Ms. Pac-Man lures ghosts near a power pill before eating them.

References

[1] R. A. Caruana. Multitask Learning: A Knowledge-based Source of Inductive Bias. In *Proceedings of the 10th International Conference on Machine Learning*, pages 41–48, 1993.

[2] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

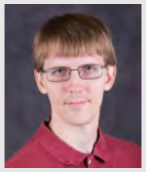
[3] D. Floreano, P. Dürr, and C. Mattiussi. Neuroevolution: from architectures to learning. *Evolutionary Intelligence*, 1(1):47–62, 2008.

[4] J. Lehman and R. Miikkulainen. Neuroevolution. *Scholarpedia*, 8(6):30977, 2013.

[5] J. Schrum and R. Miikkulainen. Evolving Agent Behavior In Multiobjective Domains Using Fitness-Based Shaping. In *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, pages 439–446. ACM, 2010.

[6] J. Schrum and R. Miikkulainen. Evolving Multimodal Behavior With Modular Neural Networks in Ms. Pac-Man. In *Proceedings of the 16th Annual Conference on Genetic and Evolutionary Computation*, pages 325–332. ACM, 2014.

[7] K. O. Stanley and R. Miikkulainen. Evolving Neural Networks Through Augmenting Topologies. *Evolutionary Computation*, 10(2):99–127, 2002.



Jacob Schrum received his B.S. degree in 2006 from Southwestern University in Georgetown, Texas, where he triple-majored in Computer Science, Math and German, graduating with honors in both Computer Science and German. He received his M.S. degree in Computer Science from the University of Texas at Austin in 2009, and his Ph.D. in 2014 from the same institution, under the supervision of Risto Miikkulainen. He has now returned to Southwestern University as an Assistant Professor of Computer Science. His research focuses on automatically learning intelligent agent behavior in complex domains such as video games, particularly using neuroevolution.

Email: schrum2@southwestern.edu
Homepage: WWW
Dissertation: WWW

Data-parallel Structural Optimisation in Agent-based Models

Dissertation by Alwyn V. Husselmann

Agent-based models offer valuable simulations of various phenomena found in disciplines such as ecology [1], microbiology [2] and social science [3]. Even the spatial distribution of crime can be simulated [4] using this method.Representing discrete entities in a system as autonomous, interactive and situated in some manner, gives rise to the notion of "bottom-up" modelling, as opposed to "top-down" approaches commonly done using partial differential equations [5]. While not considered a replacement for these, agent-based modelling (ABM) is another potent computational science tool.

Several facets of ABM make it relevant in the realms of evolutionary optimisation. Particularly, a well-researched area within ABM is that of parameter calibration. The typical process of calibrating the parameters of an ABM to more accurately reflect a natural phenomenon under scrutiny is a very time consuming process. It was not long before this process was reinterpreted as an optimisation problem under the genetic algorithm [6]. Other past attempts showed that objective functions were understandably difficult to formulate in these situations. While reasonably successful, research effort dwindled under the excessive computational strain of evaluation and continuous averaging necessary, in order to eliminate stochastic variation, as is ubiquitous in ABM.

The fortunate development of technologies such as CUDA [7] has breathed life into many projects. For those unfamiliar, CUDA is a software architecture which streamlines the use of NVIDIA graphics processing units for scientific computing. CUDA presents an excellent opportunity to drive the study of optimisation further in the context of ABM, and indeed many other evolutionary computation applications. Where only small agent-based models could be calibrated using metaheuristics, it is now much faster and larger systems clearly within reach.

An aspect of optimisation not normally considered in the context of ABM is that of combinatorial optimisation, or the structural optimisation of an agent-based model. Very few studies have been conducted on this. Junges and Klügl investigated this under the term of behaviour optimisation [8] and developed a methodology for building agent-based models guided by machine learning. Privošnik in 2002 evolved agents operated by finite state machines which solved the Ant Hill problem [9], and van Berkel [10] generated models in the style of genetic programming with predefined building blocks.

With high performance computing at hand, the author of this thesis set out to streamline structural (behavioural) optimisation in ABM. A problem that presented itself very early on in the simple formulation of a genetic programming (GP) algorithm was the choice of terminals and nonterminals, a difficult issue so far frustratingly unsolved [11]. In the formulation of an agent-based model, it is common for experts in the domain to be familiar with a significant portion of a model. It was therefore reasonable to seek a method of reducing the search space based on this assumption. It was not immediately obvious how to accomplish this.

Domain-specific languages have in the past been successful in simplifying the development of code for those who are not familiar with programming [12]. Languages typically do not offer a built-in pre-processor style optimiser, but the use of a language to aid in the search for a method to both expose and allow optimisation provided a natural method for limiting search space by forcing the user to identify what part of their model is "uncertain". An easy method of creating domain-specific languages is done by Multi-stage Programming (MSP), a programming paradigm developed by Walid Taha [13, 14]. This made possible a language developed by DeVito and colleagues at Stanford University [15]. Terra is considered the low-level counterpart to the Lua scripting language, built on LuaJIT and LLVM. It is a compiled language which allows run-time compiled code generation. In this thesis, the Terra language was used to create a new agent-based modelling language called MOL. An example of this language is shown in Listing 1.

For brevity, not all the features of this language can be detailed here. However, of particular note is line 19. This line provides an objective function, and a block of code, given by the operator, as starting point. The objective function cell_count is in fact a macro, defined above the code in pure Terra. Due to Terra's use of LLVM, it is also possible to link arbitrary libraries to this language, and use them at run-time with compiled calls. Using Clang through Terra allows one to write C code within the same file, and use this directly within this modeling language. Through compilers written in Terra, MOL is currently compiled without modification to both CUDA code, as well as single-threaded code.

```
mol
    if  me > 0 then

       defvar I = compute_illuminance
       defvar state = get_my_state
       defvar myspecies = get_my_species
       defvar activated_counter = get_activated_state_counter

       if  state == ACTIVATED then
           if  (randomfloat) < GAMMA then
               increment_activated_state_counter
               go_to_resting
           else
               if  randomfloat < BETA * I then
                   go_to_inhibited
               end
           end
       else
           select recombination to minimise(cell_count)
               if  state == INHIBITED then
                   if  randomfloat < DELTA then go_to_resting end
               else
                   if  state == RESTING then

                       if  randomfloat < (get_split_probability) then
                           make_split
                       else
                           if  randomfloat < (ALPHA*I) then go_to_activated end
                       end
                   end
               end
           end
       end

       defvar dir = get_kawasaki_move
       move dir
    end
end
```

Listing 1: A Photobioreactor model in MOL, built to model the growth kinetics and optimise yield times.

The optimisation algorithm used in this language is a very simple evolutionary optimiser operating on K-expressions of the Gene Expression Programming algorithm [16]. The optimiser operates directly on syntax trees in Lua tables, freely translating between strings of K-expressions and trees for applying evolutionary operators. Without run-time code generation, it would have been much more computationally expensive to evaluate each candidate model.

In summary, this language has successfully evolved finite state machines for a photobioreactor agent-based model, and has also solved the Santa Fe Ant Trail problem. It performs well, and it is anticipated that other platforms such as multi-core processors and MPI would be useful in future for accelerating simulations based on available hardware. It is hoped that the language will be made available in the near future for the public to use.

References

[1] Grimm, V., Revilla, E., Berger, U., Jeltsch, F., Mooij, W.M., Railsback, S.F., Thulke, H.H., Weiner, J., Wiegand, T., DeAngelis, D.L.: Pattern-oriented modeling of agent-based complex systems: Lessons from ecology. Science **310** (2005) 987–991

[2] Ferrer, J., Prats, C., López, D.: Individual-based modelling: An essential tool for microbiology. J Biol Phys **34** (2008) 19–37

[3] Epstein, J.M.: Agent-based computational models and generative social science. Generative Social Science: Studies in Agent-Based Computational Modeling (1999) 4–46

[4] Birks, D., Townsley, M., Stewart, A.: Generative explanations of crime: Using simulation to test criminological theory. Criminology **50** (2012) 221–254

[5] Macal, C.M., North, M.J.: Tutorial on agent-based modeling and simulation part 2: How to model with agents. In: Proc. 2006 Winter Simulation Conference, Monterey, CA, USA. (3-6 December 2006) 73–83 ISBN 1-4244-0501-7/06.

[6] Calvez, B., Hutzler, G.: Automatic tuning of agent-based models using genetic algorithms. In: Proceedings of the 6th International Workshop on Multi-Agent Based Simulation (MABS 2005). (2005) 41–57

[7] NVIDIA: CUDA C Programming Guide. 5.0 edn. (July 2013)

[8] Junges, R., Klügl, F.: Evaluation of techniques for a learning-driven modeling methodology in multiagent simulation. In Dix, J., Witteveen, C., eds.: Multiagent System Technologies. Volume 6251 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2010) 185–196

[9] Privošnik, M., Marolt, M., Kavčič, A., Divjak, S.: Construction of cooperative behavior in multi-agent systems. In: Proceedings of the 2nd International Conference on Simulation, Modeling and optimization (ICOSMO 2002), Skiathos, Greece, World Scientific and Engineering Academy and Society (2002) 1451–1453

[10] van Berkel, S., Turi, D., Pruteanu, A., Dulman, S.: Automatic discovery of algorithms for multi-agent systems. In: Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference companion. (July 2012) 337–334

[11] O'Neill, M., Vanneschi, L., Gustafson, S., Banzhaf, W.: Open issues in genetic programming. Genetic Programming and Evolvable Machines **11** (2010) 339–363

[12] Franchi, E.: A domain specific language approach for agent-based social network modeling. In: IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. (2012) 607–612

[13] Taha, W., Sheard, T.: Multi-stage programming with explicit annotations. In: ACM SIGPLAN Notices. Volume 32., ACM (1997) 203–217

[14] Taha, W.: A gentle introduction to multi-stage programming. In: Domain-Specific Program Generation. Springer (2004) 30–50

[15] DeVito, Z., Hegarty, J., Aiken, A., Hanrahan, P., Vitek, J.: Terra: a multi-stage language for high-performance computing. In: PLDI. (2013) 105–116

[16] Ferreira, C.: Gene expression programming: A new adaptive algorithm for solving problems. Complex Systems **13**(2) (2001) 87–129

Alwyn Husselmann recently completed his Ph.D. in Computer Science in 2014 at Massey University, which was supervised by Prof. Ken Hawick and Prof. Chris Scogings. His research shed light on evolutionary computation in the context of agent-based modeling, involving parallel computing and multi-stage programming. He is currently lecturing at the same institute, and investigating automatic platform code generation for combinatorial metaheuristic optimisers.

Dissertation: WWW
Homepage: http://www.massey.ac.nz/ avhussel/
Email: A.V.Husselmann@massey.ac.nz

Calls and Calendar

January 2015

Learning and Intelligent OptimizatioN Conference (LION9)

January 12-16, 2015, Lille, France

Homepage: http://www.lifl.fr/LION9/

The large variety of heuristic algorithms for hard optimization problems raises numerous interesting and challenging issues. Practitioners are confronted with the burden of selecting the most appropriate method, in many cases through an expensive algorithm configuration and parameter tuning process, and subject to a steep learning curve. Scientists seek theoretical insights and demand a sound experimental methodology for evaluating algorithms and assessing strengths and weaknesses. A necessary prerequisite for this effort is a clear separation between the algorithm and the experimenter, who, in too many cases, is "in the loop" as a crucial intelligent learning component. Both issues are related to designing and engineering ways of "learning" about the performance of different techniques, and ways of using past experience about the algorithm behavior to improve performance in the future. Intelligent learning schemes for mining the knowledge obtained from different runs or during a single run can improve the algorithm development and design process and simplify the applications of high-performance optimization methods. Combinations of algorithms can further improve the robustness and performance of the individual components provided that sufficient knowledge of the relationship between problem instance characteristics and algorithm performance is obtained.

This meeting, which continues the successful series of LION events (see LION 5 in Rome, LION 6 in Paris, LION 7 in Catania, and LION 8 in Gainesville), is exploring the intersections and uncharted territories between machine learning, artificial intelligence, mathematical programming and algorithms for hard optimization problems. The main purpose of the event is to bring together experts from these areas to discuss new ideas and methods, challenges and opportunities in various application areas, general trends and specific developments.

Conference Organizers:

Clarisse Dhaenens
Laetitia Jourdan
Marie-Eléonore Marmion

Important Dates

Conference:	January 12-16, 2015

FOGA XIII – Foundation of Genetic Algorithms
Call for Participation

January 17-20, 2015, Aberystwyth, Wales, UK

Homepage: http://www.sigevo.org/foga-2015

FOGA 2015 takes place in Aberystwyth, Wales (UK) January 17-20, 2015. FOGA is the premier event on the theoretical foundations of evolutionary computation and all kinds of randomised search heuristics, including but not limited to evolutionary algorithms, ant colony optimisation, artificial immune systems and particle swarm optimisation. Accepted papers will be published in post-conference proceedings by ACM Press.

Submission deadline was 31/08/2014. After rigorous peer reviewing 16 papers have been accepted for presentation and publication. The list of accepted papers is online: http://foga2015.dcs.aber.ac.uk/papers.html

We are inviting researchers to attend FOGA and join us in lively discussions about the presented work. The deadline for standard registration is on 30/11/2014. Late registration is possible, although at a higher cost. Registration is done online: http://foga2015.dcs.aber.ac.uk/registration.html

Registration includes admission to all sessions on all four days; coffee breaks on all four days; lunch on the three days without an excursion; participation in the excursion including Welsh tea; conference dinner; access to pre-proceedings; printed post-proceedings.

Accepted papers

- Sandra Astete-Morales, Marie-Liesse Cauwet, Olivier Teytaud: Evolution Strategies with Additive Noise: A Convergence Rate Lower Bound.
- Golnaz Badkobeh, Per Kristian Lehre, Dirk Sudholt: Black-box Complexity of Parallel Search with Distributed Populations.
- Keki Burjorjee: Implicit Concurrency in Evolutionary Computation.
- Marie-Liesse Cauwet, Shih-Yuan Chiu, Kuo-Min Lin, David Saint-Pierre, Fabien Teytaud, Olivier Teytaud, Shi-Jim Yen: Parallel Evolutionary Algorithms Performing Pairwise Comparisons.
- Duc-Cuong Dang, Per Kristian Lehre: Efficient Optimisation of Noisy Fitness Functions with Population-based Evolutionary Algorithms.
- Mathys C. Du Plessis, Andries P. Engelbrecht, Andre Calitz: Self-Adapting the Brownian Radius in a Differential Evolution Algorithm for Dynamic Environments.
- Thomas Jansen: On the Black-Box Complexity of Example Functions: The Real Jump Function.
- Timo Kötzing, Andrei Lissovoi, Carsten Witt: (1+1) EA on Generalized Dynamic OneMax.
- Oswin Krause, Christian Igel: A More Efficient Rank-one Covariance Matrix Update for Evolution Strategies.
- Johannes Lengler, Nick Spooner: Fixed Budget Performance of the (1+1)-EA on Linear Functions.
- Alan Lockett: Insights From Adversarial Fitness Functions.
- Luigi Malagò, Giovanni Pistone: Information Geometry of Gaussian Distributions in View of Stochastic Optimization.
- Richard Mealing, Jonathan Shapiro: Convergence of Strategies in Simple Co-Adapting Games.
- Adam Prügel-Bennett, Jonathan Rowe, Jonathan Shapiro: Run-Time Analysis of Population-Based Evolutionary Algorithm in Noisy Environments.
- Eric Scott, Kenneth De Jong: Understanding Simple Asynchronous Evolutionary Algorithms.
- Renato Tinos, Darrell Whitley, Francisco Chicano: Partition Crossover for Pseudo-Boolean Optimization.

More information about FOGA can be found at its web site: http://www.sigevo.org/foga-2015

Organizers

Jun He	Aberystwyth University, Wales, UK
Thomas Jansen	Aberystwyth University, Wales, UK
Gabriela Ochoa	University of Stirling, Scotland, UK
Christine Zarges	University of Birmingham, England, UK

April 2015

Evostar 2015 - EuroGP, EvoCOP, EvoBIO and EvoWorkshops

April 8-10, 2015, Copenhagen, Denmark

Submission deadline: November 25, 2014

Homepage: www.evostar.org

EvoStar comprises of five co-located conferences run each spring at different locations throughout Europe. These events arose out of workshops originally developed by EvoNet, the Network of Excellence in Evolutionary Computing, established by the Information Societies Technology Programme of the European Commission, and they represent a continuity of research collaboration stretching back nearly 20 years.

The five conferences include:

- EuroGP 18th European Conference on Genetic Programming
- EvoBIO 12th European Conference on Evolutionary Computation, Machine Learning and Data Mining in Computational Biology
- EvoCOP 15th European Conference on Evolutionary Computation in Combinatorial Optimisation
- EvoMUSART 4rd International Conference on Evolutionary and Biologically Inspired Music, Sound, Art and Design
- EvoApplications 16th European Conference on the Applications of Evolutionary and bio-inspired Computation including the following tracks
  - EvoCOMNET Application of Nature-inspired Techniques for Communication Networks and other Parallel and Distributed Systems
  - EvoCOMPLEX Applications of algorithms and complex systems
  - EvoENERGY Evolutionary Algorithms in Energy Applications
  - EvoFIN Track on Evolutionary Computation in Finance and Economics
  - EvoGAMES Bio-inspired Algorithms in Games
  - EvoHOT Bio-Inspired Heuristics for Design Automation
  - EvoIASP Evolutionary computation in image analysis, signal processing and pattern recognition
  - EvoINDUSTRY The application of Nature-Inspired Techniques in industrial settings
  - EvoNUM Bio-inspired algorithms for continuous parameter optimisation
  - EvoPAR Parallel and distributed Infrastructures
  - EvoRISK Computational Intelligence for Risk Management, Security and Defense Applications
  - EvoROBOT Evolutionary Computation in Robotics
  - EvoSTOC Evolutionary Algorithms in Stochastic and Dynamic Environments

Featuring the latest in theoretical and applied research, EVO* topics include recent genetic programming challenges, evolutionary and other meta-heuristic approaches for combinatorial optimisation, evolutionary algorithms, machine learning and data mining techniques in the biosciences, in numerical optimisation, in music and art domains, in image analysis and signal processing, in hardware optimisation and in a wide range of applications to scientific, industrial, financial and other real-world problems.

EVO* Poster

You can download the EVO* poster advertisement in PDF format here

EVO* Call for Papers

You can access the call for papers of all the EVO* conferences here.

EVO* Coordinator:

Jennifer Willies, Napier University, United Kingdom
j.willies@napier.ac.uk

General Chairs:

Penousal Machado, Malcom Heywood, James McDermott, Gabriela Ochoa, Francisco Chicano, Colin Johnson, Adrian Carballai, João Correia, Antonio Mora

Local Chair:

Paolo Burelli, Aalborg University
Julian Togelius, IT University of Copenhagen

Publicity Chair:

Mauro Castelli & Paolo García Sánchez

Important Dates

Submission Deadline:	25 November 2014
Notification:	07 January 2015
Camera-ready:	21 January 2015
Conference:	8-10 April 2015

May 2015

2015 IEEE Congress on Evolutionary Computation (CEC 2015)

May 25-28, 2015, Sendai, Japan

Homepage: http://sites.ieee.org/cec2015/

Deadline December 19, 2014

The annual IEEE CEC is one of the leading events in the field of evolutionary computation. It covers all topics in evolutionary computation including: Ant colony optimization, Artificial immune systems, Coevolutionary systems, Cultural algorithms, Differential evolution, Estimation of distribution algorithms, Evolutionary programming, Evolution strategies, Genetic algorithms, Genetic programming, Heuristics, metaheuristics and hyper-heuristics, Interactive evolutionary computation, Learning classifier systems, Memetic, multi-meme and hybrid algorithms, Molecular and quantum computing, Multi-objective evolutionary algorithms, Parallel and distributed algorithms, Particle swarm optimization, Theory and Implementation, Adaptive dynamic programming and reinforcement learning, Coevolution and collective behavior, Convergence, scalability and complexity analysis, Evolutionary computation theory, Representation and operators, Self-adaptation in evolutionary computation, Optimization, Numerical optimization, Discrete and combinatorial optimization, Multiobjective optimization.

IEEE CEC 2015 will feature a world-class conference that aims to bring together researchers and practitioners in the field of evolutionary computation and computational intelligence from all around the globe. Technical exchanges within the research community will encompass keynote lectures, regular and special sessions, tutorials, and competitions as well as poster presentations. In addition, participants will be treated to a series of social functions, receptions, and networking to establish new connections and foster everlasting friendship among fellow counterparts.

Important Dates:

- Competition Proposals Due: September 26, 2014
- Tutorial Proposals Due: January 9, 2015
- Special Session Proposals Due: October 31, 2014
- Paper Submission Due: December 19, 2014

July 2015

GECCO 2015 - Genetic and Evolutionary Computation Conference

July 11-15, 2015, Madrid, Spain

Homepage: http://www.sigevo.org/gecco-2015

Abstract submission deadline: January 21, 2015

Full paper submission deadline: February 4, 2015

The Genetic and Evolutionary Computation Conference (GECCO-2015) will present the latest high-quality results in the growing field of genetic and evolutionary computation.

Topics include: genetic algorithms, genetic programming, evolution strategies, evolutionary programming, real-world applications, learning classifier systems and other genetics-based machine learning, evolvable hardware, artificial life, adaptive behavior, ant colony optimization, swarm intelligence, biological applications, evolutionary robotics, coevolution, artificial immune systems, and more.

Workshop Submission Deadlines

Workshop Submission Deadline	March 28, 2014
Decision Notification	April 15, 2014
Camera-ready Submission	April 25, 2014
Conference	July 12-16, 2014

Organizers

General Chair:	Anna I Esparcia-Alcázar
Editor-in-Chief:	Sara Silva
Local Chairs:	J. Ignacio (Iñaki) Hidalgo
	Luis Hernández-Yáñez
Publicity Chair:	A. Şima Etaner Uyar
Tutorials Chair:	Anabela Simões
Workshops Chair:	Gisele Pappa
Competitions Chair:	Mike Preuss
Social Media Chair:	Pablo García-Sánchez
Business Committee:	Jürgen Branke
	Pier Luca Lanzi
EC in Practice Chairs:	Thomas Bartz-Beielstein
	Jörn Mehnen

Venue

Madrid, the capital of Spain, is a cosmopolitan city that combines the infrastructures and status as an economic, financial, administrative and service centre, with a large cultural and artistic heritage, a legacy of centuries of exciting history. Madrid has one of the most important historic centres of all the great European cities. The historic centre, also known as the "Madrid of Los Austrias" (in reference to the Hapsburg monarchs), and the spectacular Plaza Mayor square are a living example of the nascent splendour of the city in the 16th and 17th centuries. Art and culture play a key role in Madrid's cultural life. The capital has over 60 museums which cover every field of human knowledge. Highlights include the Prado Museum, the Thyssen-Bornemisza Museum and the Reina Sofía National Art Centre, dedicated to contemporary Spanish art. Madrid's extensive and beautifully maintained parks and gardens —-like the Retiro park, formerly the recreational estate to the Spanish monarchs, the Casa de Campo and the Juan Carlos I park—- offer inhabitants and visitors the chance to enjoy the sunshine, stroll, row on their lakes or feed the squirrels, in one of the greenest capitals in Europe. But if there is one thing that sets Madrid apart, it must be its deep and infectious passion for life that finds its outlet in the friendly and open character of its inhabitants. Concerts, exhibitions, ballets, a select theatre offer, the latest film releases, the opportunity to enjoy a wide range of the best Spanish and international gastronomy, to savour the charms of its bars and taverns.

The conference will be held at The Meliá Castilla hotel, which is considered one of the most emblematic hotels in Madrid, with an appealing blend of perfectly balanced classic and contemporary styles, where peace and urban life are accomplices to delight the guests. The excellent location, a few minutes from Paseo de la Castellana, 15 minutes from Barajas airport, near Chamartín train station and the Real Madrid Santiago Bernabéu football stadium, makes the Meliá Castilla the perfect choice for your visit to Madrid.

More Information

Visit www.sigevo.org/gecco-2015 for information about electronic submission procedures, formatting details, student travel grants, the latest list of tutorials and workshop, late-breaking abstracts, and more.

GECCO is sponsored by the Association for Computing Machinery Special Interest Group for Genetic and Evolutionary Computation.

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 solicit 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 editor@sigevolution.org.

All the issues of SIGEVOlution are also available online at www.sigevolution.org.

Notice to Contributing Authors to SIG Newsletters