Exploring the Evolution of Meta Parametric Models

ABSTRACT
Parametric associative logic can describe complex design scenarios but are typically non-trivial and time consuming to develop. Optimization is being widely applied in many fields to find high-performing solutions to objective design needs, and this is being extended further to include user input to satisfy subjective preferences. However, whilst conventional optimization approaches can set good parameters for a model, they cannot currently improve the underlying logic defined by the associative topology of the model, leaving it limited to predefined domain of designs.

This work looks at the application of Cartesian Genetic Programming (CGP) as a method for allowing the automatic generation, combination and modification of valid parametric models, including topology. This has value as it allows for a much greater range of solutions, and potentially computational “creativity,” as it can develop unique and surprising solutions. However, the application of a genome-based definition and evolutionary optimization, respectively, to describe parametric models and develop better models for a problem, introduce many unknowns into the model generation process. This paper explains CGP as applied to parametric design and investigates the difference between using mating, mutating and both strategies together as a way of combining aspects of parent models, under selection by a genetic algorithm under random, objective and user (Interactive GA) preferences. We look into how this effects the resultant overiterated interaction in relation to both the geometry and the parametric model.
INTRODUCTION

Computational design, specifically parametric associative design, has become widely adopted. This method has many strengths, leveraging a formal but flexible relational geometry definition to capture design logic, to drive advanced form generation and support analysis and optimisation. Parametric design is often used in the early concept stage, when many options are considered over a short time in order to explore the design and solution space, with parametric variability speeding up exploration significantly. Despite its popular use during the exploratory phase parametric associative models (PM) are not easy to reconfigure, especially if a change has not been designed into the model from the start (Davis et al. 2011). To rephrase this more technically: parametric changes (modifying variables) are trivial, whereas associative changes (modifying the topology) of a parametric associative model is difficult.

The relative complexity of manipulating parameters vs. associations can be demonstrated if we consider changing one at random; if a parametric value is changed it is likely that the model will still function, but if a component link or component type is changed randomly then it is likely to break the model. One solution to algorithmically create and change valid topological definitions is applying Cartesian Genetic Programming (CGP), which is a Genetic Programming method that uses evolutionary algorithms to algorithmically create functional and optimal designs. CGP was originally developed for designing circuits, specifically logic gate–based chips (Miller et al. 2000). The main innovation of CGP is to develop an effective schema that enables the “genome” of the design—an encoded numerical represenation that is easy to manipulate computationally and that defines a design when processed into a “phenotype”—to encompass both topological and functional elements of a directed acyclic graph (DAG). In the case of an electrical circuit, the DAGs define the circuit with electrical messages that pass from inputs through a potentially large set of logic gates and then produce outputs that define which type of logic gate is used at each node, while the topology defines how they are connected. This represents a system where the messages go in one direction and there is no feedback or recursion, hence it is directed acyclic.
CGP has been shown to be successfully extended to parametric modelling (Harding and Shepherd 2016). The reason for this is that parametric modelling, as used in packages such as Grasshopper, GenerativeComponents and Dynamo BIM, essentially defines a graph like a representation of dataflows that must be a DAG. If we replace the logic gates with components and the circuits with data connections, then the two systems operate the same way. This is a novel approach and is potentially very powerful at enabling optimisation or machine learning to operate on parametric models, both at the parametric and accusative levels. However, the complexity of automated model generation by CGP is little understood. As a result, this paper investigates the relationships between the CGP representation, the PM and the resultant geometry, especially regarding the effects of iteratively combining models, which is a principal mechanism for evolutionary approaches.

**METHOD**

The field of CGP is relatively new and experience in applying it effectivity is growing with the research. Whilst there is not yet a consensus about effective approaches to its application, as there is in say, genetic algorithms, most current applications of CGP follow broadly similar methods, and the current authority on this method is Miller (2011). For this study a relatively standard implementation of the CGP schema was used for defining a parametric model on a fixed grid (hence Cartesian) of components. In our case three arrays were used to define a complete parametric model: (i) a 2D “Function” array (F) defining the component types from a collection of possible components, (ii) a 3D “Topology” array (T) that determines appropriate links between component inputs and outputs, and (iii) a 1D “Metrics” array (M) of the values of the input parameters (Figure 2). The use of CGP is similar to its application in circuit programming, but an important difference is that the data is “typed.” Whereas circuits simply transfer binary on-off messages, a parametric system uses a range of incompatible data types, such as numbers, characters and geometry (points, lines, surfaces, etc.) as inputs, which complicates the generation. As such there generation constraint with DAGs, in that nodes/component inputs must be connected to the right upstream outputs to allow the function of the component to work. For example, the definition of a line must take two points, and if one input is a number and the other is not, then the line cannot be defined adequately. Harding and Sheppard (2016) discuss this issue in more detail.

With this formal but flexible definition of a parametric model consistent with the CGP schema, it is possible to undertake an evolutionary process to generate, change and mix different models together. The CGP generation works by initially assigning random values for F, T and M. It is possible to constrain the values of F and T so that all inputs to the components have the right type. Here we can see that the components or functions used in this process are important; they are defined by the user and can theoretically be any parametric component. For any component to actually have a chance of working, all inputs of a chosen component must be represented in other component outputs, or the metric value M.

Beyond initial generation, to most simple approach to changing a model is to “mutate” it by randomly changing some values in some or all of F, T or M. Doing this to M is the same as changing parameters in a typical genetic optimization. Changing T or F generates a new logic in the model, and assuming only a few elements are changed, can result in small but significant effects on the results in output geometry (Figure 3).
We can also "mate" multiple models by mixing the values in T, F and M of two (or more) models to make a new derivative model. The approach applied here uses a quite typical splitting method, where the genome of both models is cut into pieces and then alternate pieces are used to generate a new genome (as demonstrated in Figure 4). It is worth noting that this splitting is realised by the CGP by using a fixed grid size of elements. However there are other techniques that can remove this limitation. This simplifies the process, as the genome remains the same size and the location of individual genes/values in the genome/matrix are comparable across models and discernible in the phenotype/generated parametric model. In our case, a single variable position crossover point was used.

To explore the effects of mutation and mating on the development of solutions, a custom lightweight parametric system was developed in JavaScript with a limited range of components. As Harding (2014) elaborates, the choice of components used essentially determines the pallet of geometry available to the algorithm and thus directs the type of results achieved. To ensure a level of focus on the design, a relatively limited set of components is preferable to produce meaningful relevant designs, which are comparable between themselves. For this study a component set was used for CGP, oriented towards the production of single-story modernist plans, nicknamed the "Farnsworth set." It includes basic geometry, but also special components to define the actual "built elements," namely: rectangle floor,
rectangle roof, solid wall, glazed wall. An automatically generated example is shown in Figure 5. The system was set up to support genetic evolutionary processes, using a “roulette wheel” based selection process. As part of this exploration we compared both objective fitness values and subjective fitness (user input preference scores) inputted via the web browser.

To ascertain the potential of generated models, we explored useful objective metrics to measure models, noting that there is an important difference between the parametric model and the geometry model. Since the sample design is not intended for a particular programmatic requirement, other attributes were calculated as indicators to determine whether the resulting model is useful as a design artifact. Indicators include the number of geometry elements created (number of points, lines, etc.) and the type and variety of elements (diversity of geometry vs. a few repeating elements, e.g., all walls, all floors, etc.). Similarly, evaluation metrics were developed for the parametric model to determine model fitness, and these were based on existing approaches used to analyse computer programs (Davis et al. 2013). This was to get values to measure how “correct” and how well the model was working, independent of its actual geometry output. The main metrics were: the statistical variance of components (especially the number of “built element” components); the “efficiency” in terms of the percentage of components used; and metrics for the complexity of the model (cyclo-matic complexity). A detailed list and explanation of the metrics used can be found in the appendix.

We then set up an experimental approach to look at the effect of an evolutionary process as applied to a pool of parametric models in order to understand how the iterated application of a genetic algorithm, using the CGP schema to manipulate the designs, affects the models. An initial fixed number of models
is randomly generated, based on the predefined CGP schema/ genome. The key properties of the CGP schema as mentioned above are components to use, grid size and number of numerical inputs. The GA is initialized, which also requires inputs such as the size of the generation and how many generations to run the algorithm for. The GA can optimize the models based on one of three approaches: (i) completely random, (ii) objective model fitness metrics, and (iii) subjective user preference. The GA can be run to a user-defined evolutionary method (mate/mutate/both) to create a new generation of CGP models. A small example of this process is shown in Figure 6. Note that unlike typical heredity trees, models are ordered on the x-axis left to right based on their objective performance values, so that we can see how this changes over time (Figure 8).

As outlined this was applied while looking at three key CGP generation approaches: mutation only, mating only and mating then mutation. The intent was to identify any interesting effects of the methods and see if it was possible to develop any identifiable “best” approach. For objective fitness, the metric was the amount of geometry generated by the solution, while the subjective fitness was based on the voting preference of one user marking a model from 1 to 10, based on personal preference; in the latter case, turning the study into the use of an interactive evolutionary algorithm.

The system is publically available online and can be tried at: http://www.metadesignlab.com/demo/arcadia-2017

**INITIAL RESULTS**

The results are complex and still being analysed more deeply for insight. However, some interesting initial findings have been found. The current analysis looks at comparing all the aforementioned CGP combination methods with all of the different GA optimisation criteria.

Looking broadly at the models generated, linear correlations were found between component use efficiency and geometry output, irrespective of the generation criteria (random, objective or subjective/user driven). However, there is an unintuitive lack of correlation between parametric model complexity and output geometry complexity. It was also shown that models developed by CGP have higher complexity properties than the human-produced counterparts shown in Davis (2013), implying an ability solve problems radically differently but also produce solutions that are not easily readable by humans. It was also shown that by looking at large numbers of randomly generated models, one could show whether there were relationships or not between different model metrics (Figure 7).

The discussed metrics were applied to provide insight in order to understand the effect of model creation over the whole duration of an evolutionary processes, looking at the effect of selection (subjective vs. objective) and the method of making new models (mating, mutating or both). It was found that subjective human selection was much more effective in converging the design, whereas objective selection always produced a much wider
FURTHER WORK

This study is by all means not complete and further analysis of current results are the first priority. However, even based on the initial findings, areas that require further research/exploration have been identified, along with a few limitations and potential new directions.

One of the main areas that requires deeper consideration is how to handle model complexity. As described in the findings, the computer can generate much more complex networks.
The evolutionary tree of an automated design session using a genetic algorithm, with 30 generations containing 30 models per generation. The models are shown contained in grey boxes with red lines showing hereditary links from direct mating using CGP. The oldest models are shown at the top, with "children" and other descendants ordered going down. We observe that using only mating, the diversity in geometry output (horizontal axis) is broad and tends towards large outputs relative to the initial generated models, but is not convergent than typical manually built models, which leads to two issues. First, the more complex models become more inefficient (they have more failed or "junk" components and do not create more complexity in geometry) and second, these models become increasingly illegible.

To address these two issues, we could examine how humans create associative models and develop certain network criteria to "guide" the generative process in order to create better working complex models, reducing junk components. We could also develop additional logic that filters and translates these generated networks into more legible components and connections, which could be further modified by a designer.

Another area that requires further investigation is the evolutionary process using mating. While mating allowed for much more geometrically diverse solutions to be generated, it often resulted in inefficient CGP models, which reduced the geometric complexity. This issue is a consequence of incompatible "input" types that get passed on during slicing and recombination of different typologies. An experiment that explores the effect of different types of "crossover" could be set up to determine which approach or approaches are most effective to be used in CGP parametric models.

Lastly, the techniques discussed in this study could be further enhanced by exploring additional metrics and associating them with the models created in order for the computer to make better judgements about which models are the "fittest." More metrics that "describe" the visual quality of the designs would...
help reduce the gap in convergence between subjective selection and objective selection.

CONCLUSIONS
The application of CGP to parametric modelling demonstrates an approach that is able to control widely used design systems to automatically generate models with a much greater amount of variability and potential as compared to conventional optimizations based on solely input values. However, this will only be useful if these methods are controllable, directable and human readable. Initial findings highlight the lack of knowledge and understanding in applying computational methods to control parametric modelling and the need for further analysis and work. The complexity of these models requires new ways to gain insights into the tractability and effectiveness of applying CGP in this context. It has been shown that some metrics are able to identify and potentially see the resultant evolution of these models in productive ways for designers. Encouraging is that the results of such methods are made more useful with the introduction of human input, pointing to productive methods of human-machine interaction.

More work is required to develop better underlying methods for combining models productively, in a similar way as a human designer would, as well as interfaces that allow the human designer more efficient and practical ways of directing this powerful method to make the designs that are actually relevant and desirable by the designer.

ACKNOWLEDGEMENTS
We would like to acknowledge the support of SUTD and the SUTD MIT International Design Center

REFERENCES


APPENDIX
A graphical explanation of the metrics used to measure model and output geometry is shown on the opposite page.

IMAGE CREDITS
All figures copyright of the authors.

Sam Conrad Joyce is an Assistant Professor in Architecture and Sustainable Design at the Singapore University of Technology and Design. Prior to this he was an Associate at Foster + Partners, in the Applied Research and Development group leading structural integration. Before this he held a role as Design Systems Analyst at Buro Happold working on geometrical, structural and master-planning projects. His group the Meta Design Lab focuses on data rich systems for effective design space exploration and decision making. To enable this his work synthesizes and applies new techniques in distributed and scalable computing, multi-objective optimization, big data analytics, artificial intelligence and web based visualization.

Nazim Ibrahim has more than seven years of experience as a researcher. His work focuses on the use of Parametric tools and Computational workflows in the exploration and evaluation of various design related issues. He earned his Masters in Architecture from NUS where he explored the use of parametric tools and digital fabrication technologies at various stages of design. He firmly believes that computation design should be employed extensively to resolve the complex issues faced by design and architecture. He currently works as a Research assistant in Meta Design Lab, exploring web frameworks in creating generative and parametric design environments.
Visual explanation of the various metrics captured for this study, measuring both for the parametric model itself and the output geometry that it produces.