BUILDING INTELLIGENCE THROUGH GENERATIVE DESIGN

Structural analysis and optimisation informed by material performance

RYAN JOHAN[1], MICHAEL CHERNYAVSKY[2],
ALESSANDRA FABBRI[3], NICOLE GARDNER[4],
M. HANK HAEUSLER[5] and YANNIS ZAVOLEAS[6]
[1,3,4,6]*UNSW*
[1,3,4,6]*{r.Johan|a.fabbri|n.gardner|y.zavoleas}@unsw.edu.au*
[2]*Aurecon Sydney*
[2]*Michael.Chernyavsky@aurecongroup.com*
[5]*UNSW & CAFA Visual Innovation Institute Beijing*
[5]*m.haeusler@unsw.edu.au*

Abstract. Generative design (GD) is the process of defining high-level goals and constraints and then using computation to automatically explore a range of solutions that meet the desired requirements. Generative processes are intelligent ways to fast-track early design stages. The outcomes are analyzed simultaneously to inform decisions for architects and engineers. Whilst material properties have been defined as a driving agent within generative systems to calculate structure, material performance or structural capacity are not linked with early decision-making. In response, this paper sets a constrained approach upon traditional and non-traditional materials to validate the feasibility of structures. A GD tool is developed within Grasshopper using C-sharp, Karamaba3D, Galapagos and various engineering formulas. The result is a script, which prioritizes the structural qualities of material as a driving factor within generative systems and facilitates communication across different expertise.

Keywords. Intelligent systems; generative design; material properties; structural analysis; evolutionary algorithms.

1. Introduction: Building intelligence with generative tools

Generative Design (GD) defines processes that incorporate a series of specific principles ranging from genetic algorithms to topology optimization. The combination of these systems drives form generation. Generally, integrative design processes emphasize natural morphogenesis in which the pattern formation is contiguous to the process of materialization (Baharlou 2015). In analogy, a form may be the result of interactions between internal components and external forces (Kwinter 2008). Baharlou and Kwinter both describe unique components acting as separate classes to be synthesized into the generative design tool. These classes interact with each other within the virtual environment driven by a unique

Intelligent & Informed, Proceedings of the 24th International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA) 2019, Volume 1, 371-380. © 2019 and published by the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA), Hong Kong.

set of rules and algorithms making a complete intelligent system. Due to the nature of this system, the results can be extremely superfluous depending on how their operations are set and are informed with data; thus, the proper generative computational framework includes both mechanisms to generate possibilities and constraints to limit the range of possibilities (Holland 1999). When developed properly, this intelligent system approach may inform early design decisions.

In response, this paper follows a constrained method to set a generative framework, as one both loaded by algorithmic intelligence and informed by material data. A generative system is created to optimize design solutions based on traditional and non-traditional materials. By using 'action research' methodology, this system is developed iteratively. Each iteration "learns" by the one created before to inform the next one, building up the system's intelligence.

2. Generative Design informed by material constraints

The purpose is to investigate the possibility of integrating material-based constraints as a driving factor within generative frameworks to inform design. Different structural patterns are explored using real-time computational form-finding. The development of the generative system is investigated through constraint-generating procedures. This aids in understanding the variety of constraints being explored as well as the possibility to simultaneously link these constraints to increasingly informed engineered solutions. Becoming familiar with the bottom-up character of emergent systems is useful to manage complexity. When transferred into generative design, investigating the mechanical properties of materials and their formation within structural patterns allows form to emerge based on this class of rules. The rules are gathered by investigating data specific to the geometrical behaviour, the desired material, and the structural analysis of formulas during optimization. As a case study for structural formation, a planar truss system is used. An algorithmic pattern and the material properties of form inform generative design towards a more intelligent design process. Generative design is built upon itself by following rules and algorithms as its limitations. An action research methodology is adopted, whereat implicit is the idea that one begins a cycle of posing questions, gathering data, reflecting, and deciding on a course of action (Ferrance 2000). A GD system builds up its own intelligence by informing solutions based on collected datasets, further tested, analysed and improved upon. Informing design with sectional and material properties from the start challenges the process, but produces more feasible outcomes.

While this method maintains a generative computational framework to create all possibilities, it maintains specific constraints and/or limitations. The mechanisms of this system are bound to material properties, physical forces and construction constraints. The material properties can characterize geometrical behaviour mechanisms. In addition, motion behaviour mechanisms can perform as consequence for each parameter, where if the desired conditions are not being met, then the responsible mechanism will release an appropriate response to change the system's behaviour. Due to its nature, this topic entails a strong foundation of background data on the optimised material's mechanical properties. As there are many rules and algorithms engineers use for material and force calculations, it is

essential to choose the most appropriate ones to respond to real scenarios.

To create the planar space truss system C-sharp was used. A structural analysis plugin named Karamba3D was also used to extract the forces needed for the analysis algorithms. The mechanical properties of steel, wood and bamboo were used in the material properties component from Karamba3D and were to be compared during optimization. A utilization of safety formula would then be used to assess the generative design outcomes against all the internal properties. As this methodology is constantly tested, analyzed and validated, each iteration of the GD system could be built progressively, that is, informed by the previous one while informing the next through more constraints and limitations.

3. Background: Agent-based systems intelligence and limitations

GD approaches challenge physical limitations within a virtual environment set as a functioning system. A functioning system is one that operates through levels of development, feedback and redevelopment (Bertalanffy 1969) in search of balance. Similarly, a form may result from behavioural systemic approaches by interrelating material and spatial conditions managed by agents (Weinstock 2004). A behavioural system with highly complex rules is better understood as an agent-based one, setting the framework for a generative design workflow. The agent-based generative system inspires material-driven processes, by drawing from bottom-up approaches and by taking advantage of all the design properties to build it up. In architecture, creating a generative model would be associated with different methods to apprehend its unique complexity also with a logic that was not known from the start; however, in engineering, a generative model would closely encompass with a more physical set of properties and rules. One of the features of complex systems is emergent properties, which can be obtained through Constrained Generating Procedures (CGP's) (Holland 1999). CGP's systems prove to be a strong driver with agent-based systems, whereby each agent can be defined by a unique set of constraints as desired into a bottom-up process. As each agent has a unique set of data and rules, they are critical in defining the outcome which should be both generative, meanwhile within specific constraints. This real-time interaction relies on the agents' data structure; the agents perceive the environment within the coding system, as well as other agents and based on their defined ontology compute the proper response to any stimuli.

In 2013, Martin Tamke, David Stasiuk and Mette Ramsgard-Thomsen set a generative model through agents informing design, but due to fabrication constraints, it was compromised. Specifically, they led a material-based concept of a "growing architecture" able to sense and dynamically adapt to its environment. Even though the concept met with the aim of a generative system, fabrication was limited to a top-down approach where each strut was made of a tightly packed bundle of variable-sized rattan elements, clashing with the bottom-up approach for GD (Tamke, Stasiuk and Ramsgard-Thomsen 2013). Moreover, in 2013, Ehsan Bahatlou and Achim Menges witnessed the same limitation concerning fabrication in another study. They investigated integrating material and construction constraints to plate-like structures, a mechanism that was limited to the plane's geometry and concluded that "it is also discernible that the lack of

construction mechanisms (which naturally has been used in the plate structure), along with insufficient construction constraints caused the initial result to be far from what was expected" (Baharlou and Menges 2013). Despite such limitations, a structure's performance is commonly seen by engineers as directly proportional to its physical properties. The mistake is often made to base GD on geometry only, rather than include environment's tectonic behaviour, material properties, or other physical constraints (Pugnale et al 2011) as driving factors to emergence. However, the emerging structural phenomena of materials must also be constrained by a new agent that describes their mechanical properties forming the total parameters that respond to dynamic, and variable contextual forces (Kolarevic 2003), as a way to appropriate material into structures.

4. Case study: Calculating a truss system through material

A truss system has been selected to develop the GD system. The aim was to create a generative script for a structure that could optimize traditional and non-traditional material use. The framework was driven by four stages; the first being the component for the generative truss system; secondly, the structural analysis; thirdly, the optimization algorithm, and; finally, the evolutionary solver.

4.1. C-SHARP GENERATIVE COMPONENT

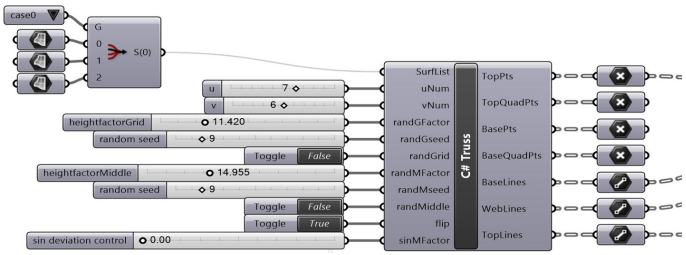

Figure 1. Planar Space Truss System C-sharp component.

As data enters the system, intelligence is built as long as the generative component is highly diverse and parametric from the outset. This way the computer can automatically change all the variables whilst simultaneously analyzing them. Creating a generative space truss system entails a robust and complex structure. A C-sharp component within the Grasshopper environment was chosen to create the system. There were three main reasons as to why C-sharp was used; the first was because by using a raw coding language such as C-sharp, the ability to 'loop' within the same set of data and perform the same function multiple times was crucial to create the generative nature needed. Grasshopper could not have been used because it is a linear coding system which can only execute a function once. The second reason was for speed and efficiency. Grasshopper exponentially requires more computing power with each component added, which resultantly increases the time to complete each calculation. Besides, C-sharp programming allows for a faster and more flexible system, as the computer's efficiency could

be saved for optimization. Finally, the truss analysis often assumes that loads are applied to the joints and not at intermediate points along the members. This means every truss member is then subjected to pure compression or tension forces only, whereas shear, bending and other more-complex forces would not have been considered. Thus, C-sharp ensured staying within the scope.

4.2. STRUCTURAL ANALYSIS

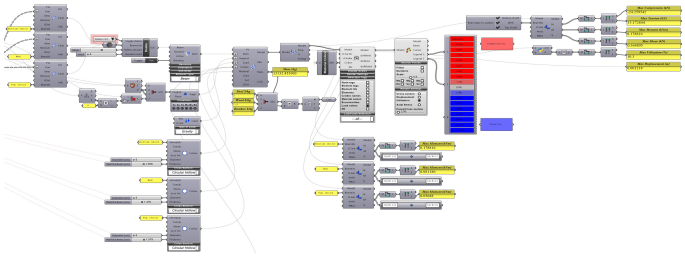

Figure 2. Karamba3D setup within Grasshopper.

The Grasshopper plugin Karamba3D was used in acquiring the data needed to execute structural calculations. Karamba3D is an engineering tool for structural analysis, which provides accurate data about spatial trusses, frames and shells. As such, it was able to compute the results for all the various forces being applied to each beam, dependent on the load types.

4.2.1. Normal force

The main forces extracted from the structural analysis plugin were the resultant normal forces. The normal forces are the components of the contact force that is perpendicular to the object, the truss beams in this case. These forces were critical to develop the optimization formula and complete the next step.

4.3. OPTIMISATION FORMULA

The normal forces from Karamba analysis set a Factory of Safety (FoS) or Utilization of Safety (UoS) algorithm used by the evolutionary solver.

4.3.1. Utilization of Safety

In engineering, a UoS is a number range which expresses how much stronger a structural system needs to be to withstand the load it will be undertaking. The UoS is defined by the following formula:

Utilization of Safety = Design Stress / Yield Stress

4.3.2. Design Stress

Design Stress is the load the object is required to withstand. This is where the resultant normal forces from Karamba were used. The design stress is calculated in Grasshopper using the following equation:

Design Stress = Applied Load / Area Cross Section

4.3.3. Yield Stress

Yield Stress is the material property showing the maximum stress up to which a body undergoes elastic deformation. The yielding point determines the limits of performance for mechanical components, since it represents the upper limit to forces that can be applied without permanent deformation. The following calculation was done within Grasshopper and used the Yield Stress for steel, which was already in Karamba3D's database.

4.3.4. 'If' Statement

To be able to start the optimization stage, the resulting Utilization of Safety algorithm needed to be run through an 'if' statement. This was critical to filter the results that fell within the UoS threshold by those that fell outside of it. More specifically the way the "if" statement works is that if the resulting UoS falls within the set range of 0.5 and 0.8, then the resulting number equals to 1 (one). However, if the resulting number fell either above or below that range, then the resulting number would be a 'false' one, which in computer terms is 0 (zero). This way, any outputs that resulted in 1 would be categorized as 'safe', whereas those that equalled to 0 would be considered as 'unsafe'.

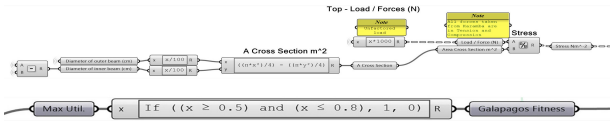

Figure 3. Yield Stress formula using the resultant normal forces (top), and; original 'if' Statement rule (down).

4.4. EVOLUTIONARY ALGORITHM

The geometry for the truss system is iterated from the external optimization loop so that the optimum solution can be discovered. To achieve this looping an evolutionary/genetic algorithm was used. There are a range of evolutionary algorithms that are available within Grasshopper, including tools such as Galapagos, Goat and Octopus. Each solver has its strengths and limitations in terms of flexibility and adaptability for its intended use. The user would decide which algorithm is adequate for each optimization problem based on the number of objectives to be fulfilled, or the necessity of finding a global or a local optimum.

4.4.1. Galapagos

In this research, the Galapagos evolutionary solver was considered adequate for the main optimization algorithm. The external input required from the solver is the genes that form the genome and the fitness value. The possible assigned genes are floating point numbers that can accept values between two numerical boundaries.

4.4.2. Solver Settings

In the Galapagos editor some further adjusting from the user is necessary before optimization commences: firstly, whether the fitness value that is being optimized for should be a minimum or a maximum; then, the number of outcomes that form a population as well as the multiplication factor for the first populations being optimized. Furthermore, decisions should be made concerning the percentage of designs of a population that can be transferred in the next generation, as well as the inbreeding percentage. The value of -100% is used for fully zoophilic and the value of 100% for fully incestuous. The algorithm terminates when the maximum assigned duration defined from the user is reached, or when the number of consecutive generations without finding an improved solution exceeds the maximum stagnant limit. Through testing of the planar truss system, some of the solver settings were changed to ensure a more reliable workflow. The multiplication factor was increased from 2x to 3x, to ensure Galapagos was exploring a wide-solution space, to begin with. The original inbreeding variable of +75% was also changed to +25% to a more zoophilic approach, ensuring greater deviation each time the solver jumped to create a new population of results.

4.4.3. Penalties

The original 'if' statement which filtered results within or outside the UoS threshold was changed to be able to optimize for a minimum mass. A penalty value had to be present in the output of the Grasshopper 'if' component since these values would be used next as a fitness for Galapagos. The first change was that instead of the 'if' statement returning a 1 if the result fell within the UoS threshold, it would now return the resulting mass.

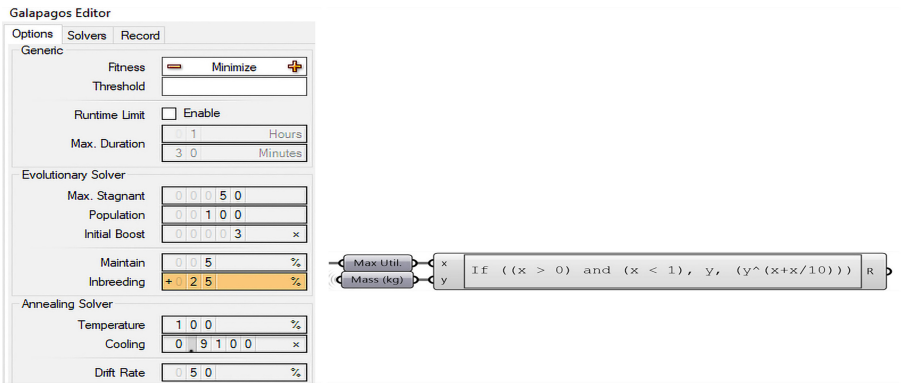

Figure 4. Galapagos solver settings (left), the 'if' Statement with Penalty Factor (right).

However, if the result fell outside the range, they could no longer give a 0, because Galapagos would consider all those results to be most optimal as the mass it is solving for is 0. Therefore, for all the structures that fell outside of the UoS threshold, a mass penalty was implemented to ensure they were always exponentially greater than the correct results that fell within the threshold. This

ensured Galapagos was always optimizing for the correct results.

5. Generative framework

The GD design framework has produced promising yet conceptual results. The ability to be able to optimise both traditional and non-traditional materials such as steel, wood and bamboo has shown that the resulting planar space truss system can be optimised to adhere between the same utilisation of safety threshold regardless of the material chosen. This has resulted in bamboo structures that can be far more optimal in terms of their total mass and cost compared to steel.

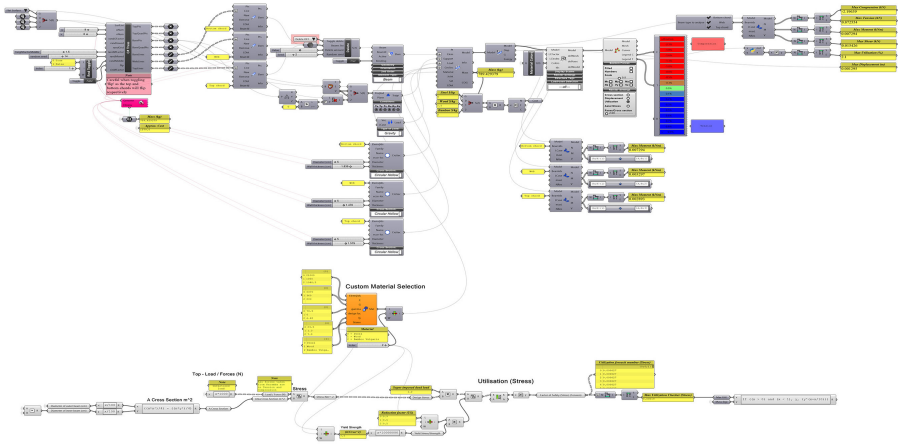

Figure 5. Complete generative framework (Ryan Johann, in collaboration with Aurecon).

6. Evaluation of research project

The project being presented produced a material-driven generative solver as an intelligent design process informed by data and behavioral inputs, progressively producing robust solutions that for this case study adhere to a utilization of safety. By comparing traditional and non-traditional structural materials, this system has distinguished the use of a hypothetical framework for generating iterations during the early stage development of a structural system without necessarily conflicting with desired design directions, all while driven by a material.

Unlike research of structural analysis lacking material inputs in a typical generative design method, the results agreed with outcomes that would have been expected by using traditional materials or techniques. The expectation of a bamboo structure to withstand the same loads of a steel system while still adhering to the same utilization of safety only becomes more apparent with the use of the material-driven generative method.

However, even though early-stage performance evaluations is becoming more valued as the power of generative computation grows, this method and its outcomes still encounter much resistance. The proposed generative process

requires a substantial investment of time and collaboration between different disciplines and will inherently need to redesign the way in which the design process is perceived. This would regard engineers, architects, computational designers and material scientists to imply a new framework in which each aspect of their contribution would drive the other, and without a digital collaborative platform to accommodate for this workflow, generative design methods such as the one proposed cannot be widely implemented.

Moreover, in the current alpha stage of development, the outcomes cannot be deemed as proposed due to the limited number of structural analysis formulas integrated. A utilization of safety is only one generalized formula to deem how suitable the resulting structure could be in terms of safety measurements. The results would still need to be further analyzed in a reputable structural analysis software such as SAP2000 & ETABS. This is not to say a generative design method could not withhold a quality assurance to a professional standard, because due to the framework's nature, new design conditions and materials can be easily integrated to further refine the performance, allowing an exploration of structural possibilities which would have otherwise been ignored.

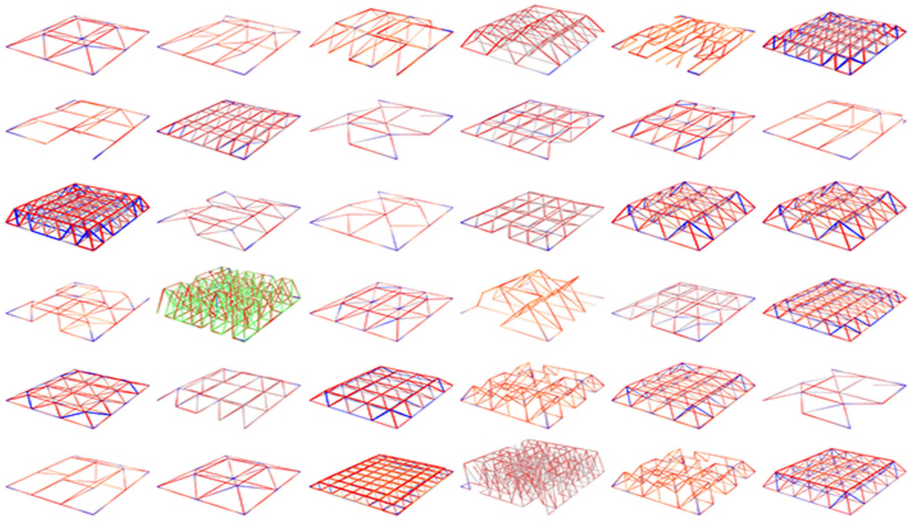

Figure 6. Various optimization results for planar space truss systems.

7. Conclusion

This paper has suggested a generative framework to optimize traditional and non-traditional materials by applying rules and algorithms as a set of constraints resulting in novel outcomes. Variations of a structural geometry can be produced through 'fast-track' iterations, as an inspiration and start for further design development by real-world configurations.

The example of the planar space truss system being optimized by different

materials addresses the capabilities of exploring new design options through a generative approach, veering from traditional analysis tools. The ability to inform the subsequent iterations by former ones and to progressively build up the intelligence of the generative process by evaluating the solutions creates a bidirectional relationship between scripting potential, parametric environments and traditional analyses for the development stage. It follows that the design process of a structural system may be applied to traditional as well as non-traditional building materials, as they may both be described by the same design requirements and processes to inform the same script.

Given the flexibility and principle of the generative framework, the overarching approach of this constrained system should be further considered within the greater 'generative' and 'analysis' ecosystem. By starting with a material-based approach, the framework is already in place for migrating design concepts into analysis and development. Collaboration between industries needs to be promoted to further develop integrated generative design workflows, whereby the system can become more robust through self-evaluation, information feeding and reflection, prompting towards more quality assurance for each of the professions involved in the building industry. A more constrained approach to the generative design framework presents with more feasible results, along with a pathway to advance communication between architects and engineers.

References

Baharlou, E. and Menges, A.: 2013, Behavioural Prototyping: An Approach to Agent-Based Computational Design Driven by Fabrication Characteristics and Material Constraints, *Rethinking Prototyping, Proceedings of the Design Modelling Symposium*, Universitat der Kunste, Berlin, 291-304.

Bertalanffy, L.: 1968, *General System Theory: Foundations, Development, Applications*, Braziller Inc., New York NY.

Ferrance, E.: 2000, *Action Research: Themes in Education*, LAB/Brown University, Providence RI.

Holland, J.H.: 1999, *Emergence: From Chaos to Order*, Perseus Books, Cambridge MA.

B. Kolarevic (ed.): 2003, *Architecture in the Digital Age Design and Manufacturing*, Spon Press, New York NY.

Kwinter, S.: 2008, *Far from Equilibrium: Essays on Technology and Design Culture*, Actar, Barcelona & New York.

Pugnale, A. and Sassone, M.: 2014, Structural Reciprocity: Critical Overview and Promising Research/Design Issues, *Nexus Network Journal*, **16/1**, 9-35.

Tamke, M., Stasiuk, D. and Ramsgard-Thomsen, M.: 2013, The Rise: Material Behaviour in Generative Design, *Adaptive Architecture: Proceedings of 33rd Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA 2013)*, 379-388.

Weinstock, M. 2004, Morphogenesis and the mathematics of emergence, *in* M. Hensel, A. Menges and M. Weinstock (eds.), *AD vol.74*, Wiley, London, 10-17.