Proceedings of the International Association for Shell and Spatial Structures (IASS)
Symposium 2015, Amsterdam
Future Visions
17 - 20 August 2015, Amsterdam, The Netherlands

Grammar-based generation of equilibrium structures through graphic statics

Juney Lee*[a], Corentin FIVET[a], Caitlin MUELLER[a]

*, [a] Massachusetts Institute of Technology
77 Massachusetts Avenue, Room 5-418, Cambridge, MA 02139, USA
juney@mit.edu

Abstract

This paper proposes a grammar-based structural design methodology using graphic statics. By combining shape grammars with graphic statics, this method enables the designer to: 1) rapidly generate unique, yet functional structures that fall outside of the expected solution space, 2) explore various design spaces unbiasedly, and 3) customize the combination of grammar rules or design objectives for unique formulation of the problem. Design tests presented in this paper will show the powerful new potential of combining computational graphic statics with shape grammars, and demonstrate the possibility for exploring richer and broader design spaces with much more trial, and less error.

Keywords: grammatical design, shape grammars, graphic statics, conceptual structural design

1. Introduction

Most commonly used parametric tools in architectural design provide extensive geometric freedom in absence of performance, while engineering analysis software mandates pre-determined forms before it can perform any numerical analysis. Digital models generated by architects typically have to be re-modelled by an engineer in a file format that is appropriate for numerical analysis software. This trial-and-error process is not only time intensive, but it also limits exploration beyond the design space filled with conventional solutions. While the rapidly advancing capabilities of computational tools have enabled architects to generate almost any form, and engineers to analyze any structure, it has not stimulated the designers to explore new structural forms. More meaningful investment of the computational resources that are available today may be in investigating new structural possibilities, rather than developing better ways of optimizing what may be inherently bad forms.

There is a need for computational design tools that can not only generate forms, but simultaneously process structural logic and rules, so that the outcome does not need to be constantly remodeled and checked with numerical analysis software. In order to explore a wide range of diverse design alternatives, computational power along with controlled randomness and intelligent optimization can be used to help the designer in unbiasedly exploring alternative solutions that are unexpected, visually interesting and yet performatively adequate.

2. Background

In computational analysis and optimization of structures, the objective function is mathematically formulated and parameters are numerically defined. This means that the design space contains all possible solutions to a given problem. While sophisticated optimization algorithms are capable of finding the optimal solution within this parameter-based design space, the result is still limited by the design space itself. During earlier stages of design, a parameter-based design space does not contain the wide range of design possibilities that the designer may want to explore (Mueller [10]). Optimization driven tools are highly dependent on initial geometry that was chosen, and typically converge on one or very few optimal solutions without providing much design diversity.

2.1. Grammar-based design

In order to broaden the design space, a grammar-based approach can be used in place of the conventional parameter-based design paradigm. Grammar-based design, or more commonly known as *shape grammars*, is a set of allowable shape transformations that can be used to define a design language, through which form generations can be automated based on a desired logic, style or objective (Stiny and Gips [12]). It has been used frequently in architectural context to not only analyze design styles and languages, but also generate new ones. William Mitchell hinted at its potential applicability to other fields such as structural design, by incorporating functional attributes and structural criteria to grammar rules in the form of Functional Grammars (Mitchell [8]).

2.2. Structural grammars

Shape grammars have been applied in engineering, most notably by Shea and Cagan, as a method called *shape annealing* [11]. Because this method uses shape transformations which are entirely geometric, a numerical analysis is required after every operation. Also, the transformations are guided by a stochastic optimization algorithm, which means that unless the transformation improves the overall performance, it will keep iterating until one is found. While successful in generating unexpected solutions, the shape annealing method is optimization driven and seeks to converge on a single optimal solution. Shape annealing is ultimately resource-intensive, and the resulting diversity is limited. Alternatively, shape grammars can be used to explore trans-typology structures, by randomly mix-and-matching elements of different typologies (Mueller [10]). A wide range of unexpected yet structurally feasible solutions can be found using a small set of grammar rules. However, previous research in this area was limited to typologies that are specific mainly to bridge structures only.

2.3. Graphic statics

Graphic statics is a graphical method of calculating forces for discrete structures under axial loads (Culmann [5]). It is based on construction of two reciprocal diagrams (Cremona [4]): the form diagram representing the actual geometry of the structure, and the force diagram that represents the internal forces through vectors. Because forces are graphically represented using vectors, no numerical analysis is required to calculate the forces. When combined with modern day computation, graphic statics can become a powerful design tool by automating the drawing process, and enabling real-time interaction between the reciprocal diagrams. Most notable examples include Active Statics (Greenwold and Allen [7]), eQuilibrium (Van Mele *et al.* [13]), and Constraint-based Graphic Statics (Fivet and Zastavni [6]).

2.4. Combining grammars and graphic statics

Shape grammars and graphic statics have been explored previously in the field of creative structural design, but never in combination. When shape grammars and graphic statics are combined, several key benefits emerge. First, geometric rules can have direct relationship with corresponding force diagrams so that any geometric transformation results in equilibrium. Because local and global equilibrium are always guaranteed, randomness can be introduced during the generation process to increase diversity of solutions. Second, because force diagrams are constructed for every transformation, there is no need for further numerical analysis. Lastly, the rules have no boundary-specific parameters, which enables the methodology to be applied to a wide range of design problems. By harnessing the intelligent, generative power of shape grammars, and the computational graphic statics that can transform forces into physical forms, architecture and structure can be integrated more seamlessly during conceptual design.

3. Methodology

3.1. Elements

The proposed methodology operates on three types of computational classes: 1) a *Force* class that is 2D vector, with a type parameter, direction and amplitude; 2) a *Node* class that includes a coordinate, state parameter (active or not-active), type parameter, and a list of forces; and 3) an Assembly class that includes a list of *Node* classes, list of members, the overall system state, and other information about the entire structure. The three elements are graphically summarized in Figure 1.

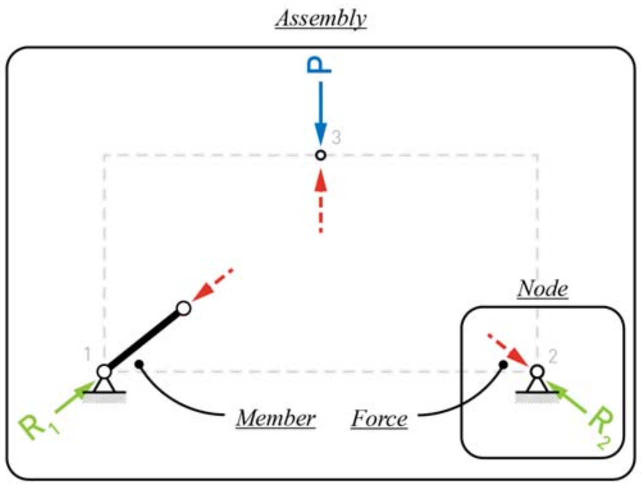


Figure 1: Conceptual overview of integrating shape grammar and graphic statics.

3.2. Concept 1: temporary forces

For a *Node* to be in a state of static and rotational equilibrium, the sum of the force vectors applied to it must equal zero. In graphic statics, equilibrium is verified when the force vectors form a closed polygon (Culmann [5]; Cremona [4]; Allen and Zalewski [2]). When the geometry of the structure is already known (Figure 2a), the force polygon construction for a *Node* is relatively simple, and the equilibrium can be easily verified. However, when the structure is not yet known, the *Node* must always be equilibrated with a temporary *Force*, shown in dotted red arrow (Figure 2b).

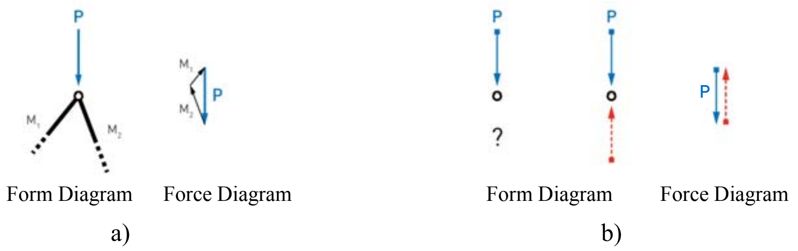


Figure 2: a) Equilibrium of a typical *Node* with applied load and members; and b) the use of a temporary *Force* (shown in red, dotted arrows) to temporarily equilibrate a *Node*.

Similarly, if a structure is assumed to be in global equilibrium, the sum of the external force vectors (applied loads and reactions) and the moments must equal zero (Figure 3a). This means that whatever the geometry of the structure ends up being, the sum of all internal force vectors (and hence all temporary *Forces*) and the moments must also equal zero, as shown in Figure 3b.

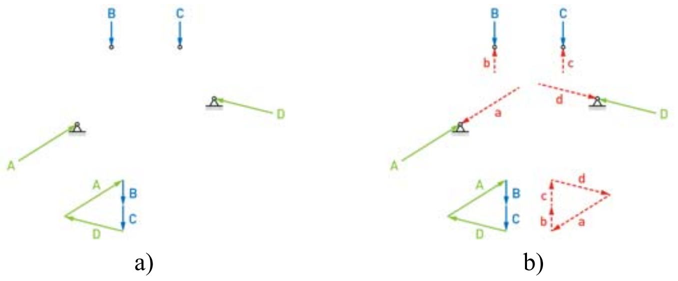


Figure 3: a) Equilibrium of external *Forces* (applied loads shown in blue, reactions in yellow) mandates: and b) equilibrium of internal temporary *Forces* (shown in red, dotted arrows).

3.3. Concept 2: graphic statics based rules

Because temporary *Forces* are used to enforce equilibrium both globally and locally, the temporary *Forces* can then be used to generate the geometry of the structure that always satisfy equilibrium. Specific geometric generations or transformations can be formulated as a rule that acts on temporary *Forces*. Figure 4 shows three examples of simple rules that can generate *Members* at *Node* 1 using the temporary *Forces*.

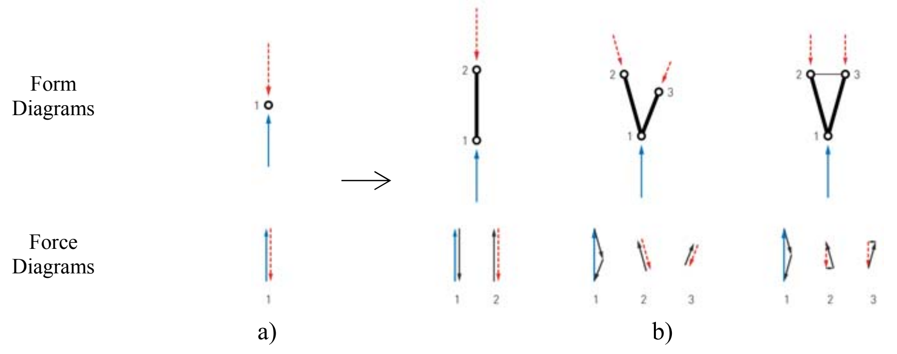


Figure 4: a) a free *Node* in equilibrium; and b) three simple rules showing how equilibrated structure can be generated using the temporary *Forces*.

3.4. Constraints

Generative grammars can be a powerful tool in discovering new structural forms. However, without intelligent constraints, the rules may be too broad and generate forms that have limited practical feasibility. In addition, the grammar rules can potentially be applied recursively, or repeated without an end. The following strategies are used to set constraints.

- *Setting reasonable local bounds, such as minimum and maximum angles or lengths.*
- *Global termination conditions, such as loop count and recursion control mechanisms.*

4. Tool setup

The proposed methodology automatically and randomly generates designs through series of rule applications. The conceptual overview of the computational setup is illustrated in Figure 5. Generally, the Grammar Engine is responsible for choosing rules, deciding where to apply them and updating the geometry. Graphical Computation Engine functions as the structural blueprint behind all procedures to be performed by the grammar engine.

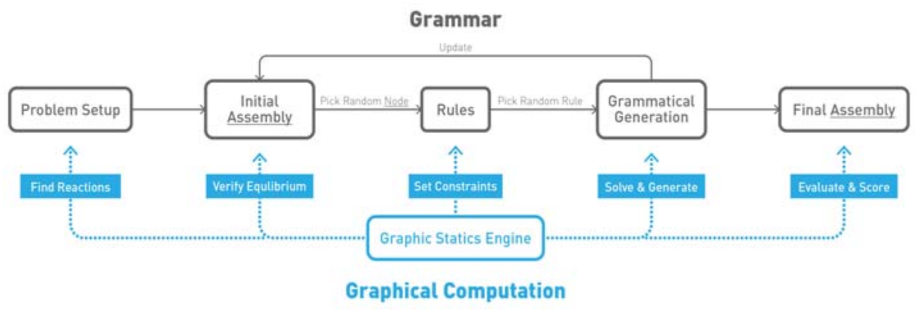
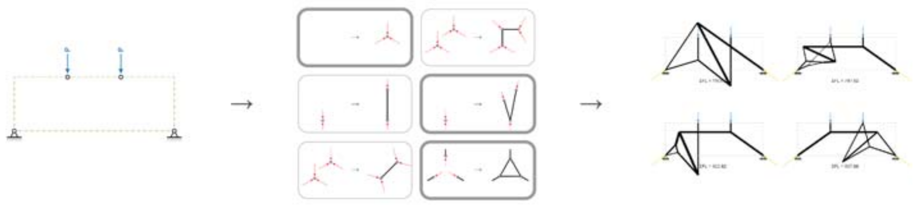


Figure 5: Conceptual overview of integrating shape grammar and graphic statics.

4.1. Workflow

Unlike most conventional engineering tools, this methodology begins without a starting geometry. First, the user sets up the problem by defining the applied loads, reactions and the locations of those forces (Figure 6a). The user then chooses the rules to apply, and weights for each rule which defines how likely it is that a rule will be randomly selected to be applied (Figure 6b). Finally, the user defines how many options to produce. Results can be further diversified by modifying the following global parameters: 1) minimum number of rule applications for each generation; 2) rule sensitivity towards the beginning, the middle or the end of the generation cycle; 3) termination conditions; and 4) random seed (Figure 6c).



a) **Step 1**: Setup problem, create initial Assembly.

b) **Step 2**: Pick rules to apply, and set rule weights.

c) **Step 3**: Generate iterations, and sort based on a desired performance metric.

Figure 6: User workflow of the proposed methodology.

4.2. Rules

The eight rules used for generating designs in this paper, are summarized in Figure 7. All parameters incorporate structural logic and knowledge, and is always verified by construction of force diagrams. While the parameters are randomly determined, it is constrained by user-defined lower and upper bounds to control that randomness.

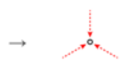
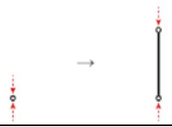
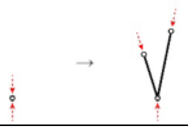
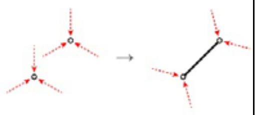
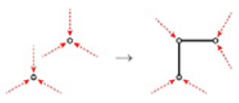
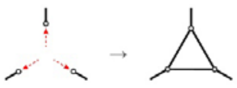
Rule 0 START generation.	**Rule 1** Create a *Node*.	**Rule 2** Extend a *Node*.	**Rule 3** Split a *Node*.
	◦ Length range: [L_{min}, L_{max}] ◦ Angle range 1: [θ_{min}, θ_{max}]	◦ Length range: [L_{min}, L_{max}]	◦ Length range 1: [L_{min}, L_{max}] ◦ Angle range 1: [θ_{min}, θ_{max}]
No geometric operations.			
Rule 4 Connect two *Nodes*.	**Rule 5** Extend & connect two *Nodes*.	**Rule 6** Close structure.	**Rule 7** END generation.
◦ Search range: [R_{min}, R_{max}] ◦ Force Factor: F	◦ Search radius: [R_{min}, R_{max}]		
			*No geometric operations.*

Figure 7: Summary of rules and parameters.

4.3. Example problem

Figure 8 shows a step-by-step generation sequence of one possible design for a simple problem.

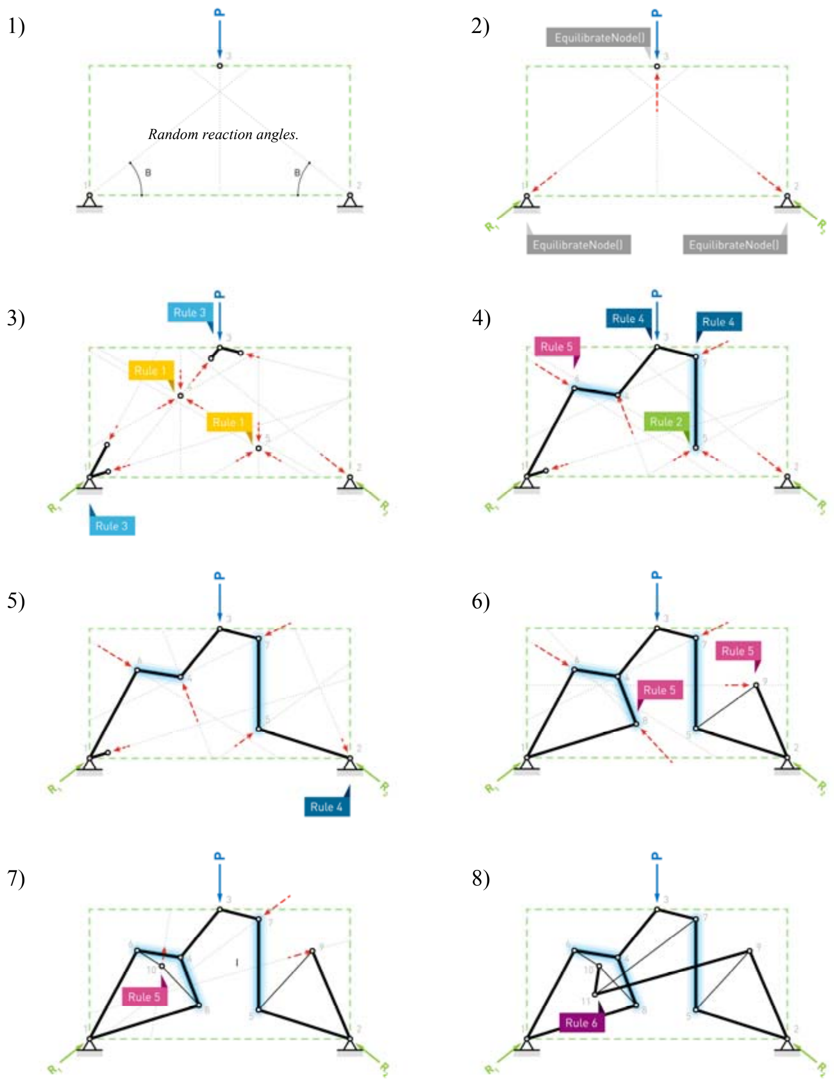


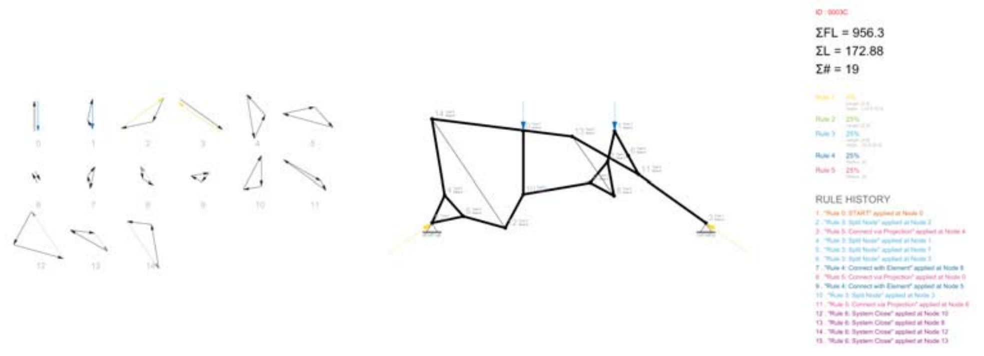
Figure 8: An example of an automatic random generation sequence.

5. Results

In this section, the proposed methodology is tested on the several example problems to demonstrate how this approach can be used to generate a wide range of diverse, discrete planar structures.

5.1. Implementation

The proposed methodology was implanted as an interactive tool, using IronPython, Rhinoceros and Grasshopper. Each iteration instantaneously generates: 1) corresponding force diagrams for every node (Figure 9a), 2) a form diagram with clear labels (Figure 9b), and 3) rule history, information and evaluation metrics for the current solution (Figure 9c). Visual representation of the forces, the evaluation metric, and the rule history which summarizes how the structure was derived, enables clearer understanding of the structure and informs better design decisions more quickly. The rule history, which records all the parameters that were used to generate the current iteration, is an important feature that enables reproducibility of the same iteration during later stages in design, when more information about the boundary conditions and the project in general, may be available. This blueprint can also be used to develop more detailed versions of the design, and allow creative breeding using genetic algorithms.


ID : 0003C

ΣFL = 956.3
ΣL = 172.88
Σ# = 19

Rule 1 [illegible]
Rule 2 25%
Rule 3 25%
Rule 4 25%
Rule 5 25%

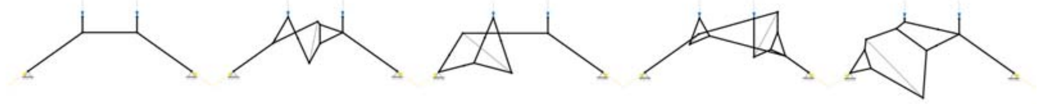
RULE HISTORY
1 . "Rule 0: START" applied at Node 0
2 . "Rule 3: Split Node" applied at Node 2
3 . "Rule 5: Connect via Projection" applied at Node 4
4 . "Rule 3: Split Node" applied at Node 1
5 . "Rule 3: Split Node" applied at Node 1
6 . "Rule 3: Split Node" applied at Node 3
7 . "Rule 4: Connect with Element" applied at Node 8
8 . "Rule 5: Connect via Projection" applied at Node 0
9 . "Rule 4: Connect with Element" applied at Node 5
10 . "Rule 3: Split Node" applied at Node 3
11 . "Rule 5: Connect via Projection" applied at Node 6
12 . "Rule 6: System Close" applied at Node 10
13 . "Rule 6: System Close" applied at Node 8
14 . "Rule 6: System Close" applied at Node 12
15 . "Rule 6: System Close" applied at Node 13

a) b) c)

Figure 9: Screenshot of an example problem in Rhinoceros.

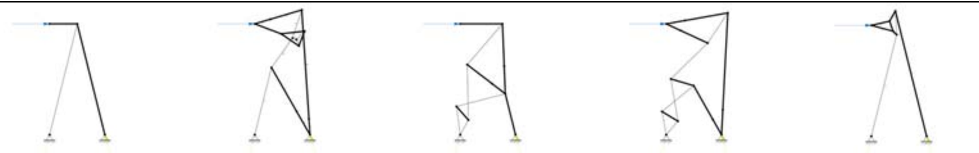
5.2. Design tests

Figure 10 shows the application of the tool on various design scenarios. Designs shown in Figure 10 are high-performing solutions with regularized geometry (first design on the left for each row) and the top four designs from 40 iterations, based on the total volume of structural material, or equivalently the total load path. Assuming constant internal stress at its optimal or final iteration state, the total volume or load path can be calculated by the simple formula: $\Sigma|P|\cdot L$, where P is the internal force of a member, and L is the length of that member (Baker *et al.* [3]). Using graphic statics, $\Sigma|P|\cdot L$ can be computed easily by multiplying the length of the member in the form diagram, and the length of the corresponding force vector which is provided by the force diagrams. The designs randomly produced through grammatical exploration exhibit significant diversity, which may often be desirable even at the sacrifice of a small amount of efficiency.



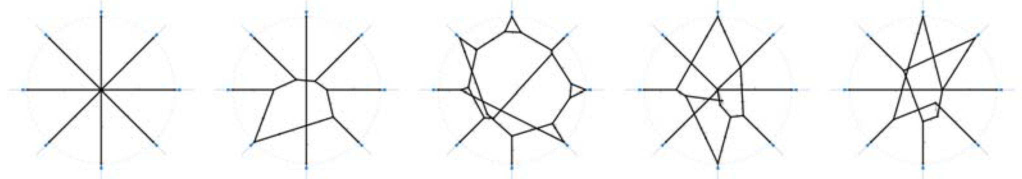
1.00	**1.42**	**1.45**	**1.53**	**1.80**
ΣPL = 465.00	ΣPL = 658.61	ΣPL = 673.61	ΣPL = 712.42	ΣPL = 838.03

a) a span-like structure



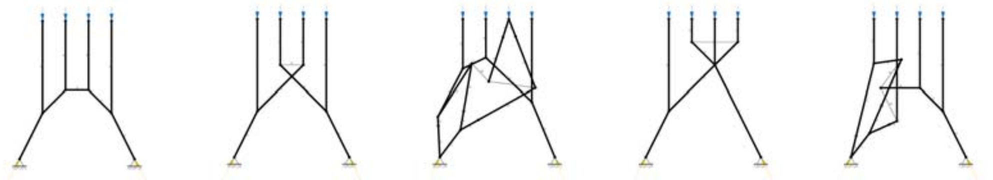
1.00	**1.08**	**1.09**	**1.15**	**1.17**
ΣPL = 1350.00	ΣPL = 1452.60	ΣPL = 1465.11	ΣPL = 1554.21	ΣPL = 1581.21

b) a cantilever-like structure



1.00	**1.01**	**1.02**	**1.04**	**1.04**
ΣPL = 800.00	ΣPL = 805.00	ΣPL = 817.57	ΣPL = 832.59	ΣPL = 835.51

c) a radial, compression structure



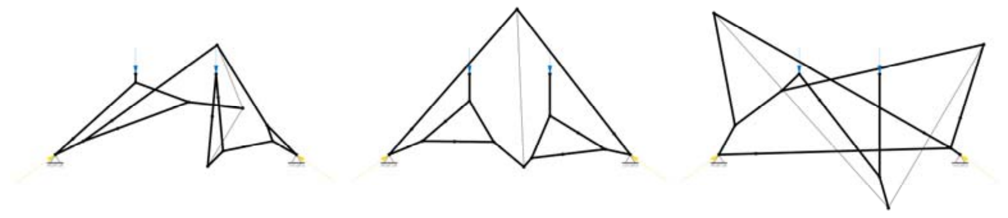
1.00	**1.07**	**1.12**	**1.14**	**1.21**
ΣPL = 435.00	ΣPL = 465.00	ΣPL = 487.85	ΣPL = 495.00	ΣPL = 524.79

d) a vertical, wall-like structure

Figure 10: Application of the methodology on four different design scenarios. Normalized metric relative to the funicular solution is shown in bold.

5.3. Exploration of parameters

Figure 11 shows the effect that rule parameter variations can have on the results. For the three options shown for design scenario same as in Figure 10-a), the only parameter that was changed was the lower and upper bounds for the split rule. Similarly, modifying the parameters for other rules will result in drastically diverse designs.



Split rule angle range: 10°-20° Split rule angle range: 20°-40° Split rule angle range:40°-60°

Figure 11: Effect of changing rule parameters on the results.

5.4. Practical applications

While the proposed tool allows fast exploration of design possibilities during conceptual design, the results will need to be interpreted by the architect and the engineer in order to develop the design with more detail and rigor during later stages in design. Figure 12 illustrates how three results selected by the designer can be developed into realistic, yet significantly different and unexpected roof structures.

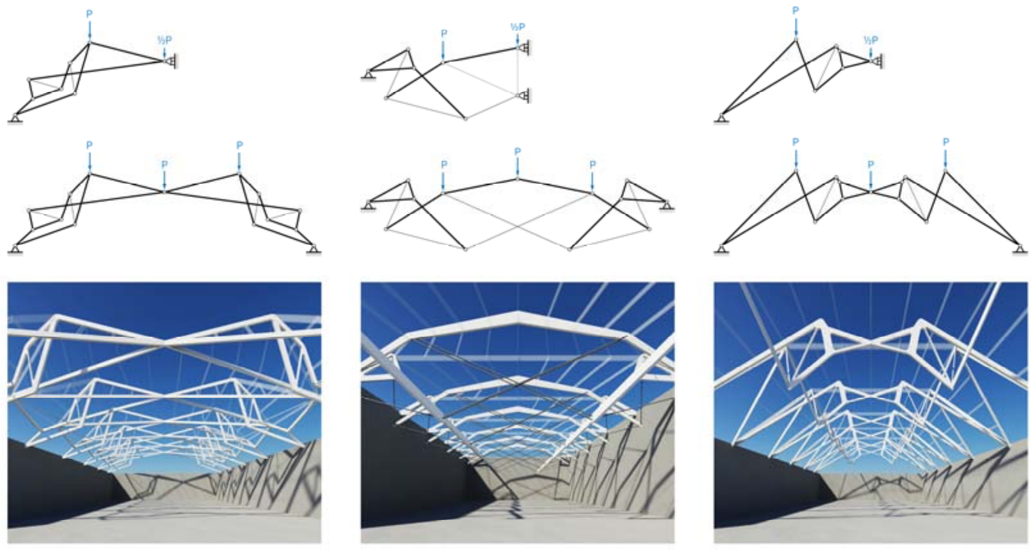


Figure 12: Sample designs with symmetry enforced, resulting in more practically applicable designs.

6. Conclusions

6.1. Contributions

This paper introduced a grammar-based design methodology, as an alternative to the conventional parametric design paradigm, which is limited in variety and often lead to expected solutions. The following specific contributions were presented:

6.1.1. More trial and less error

By incorporating forces during the form generation process, the resulting designs are guaranteed to be in equilibrium. Therefore, no further numerical analysis is required. Reduced coordination time between architects and engineers allows exploration of better and more interesting ideas faster. While most numerical analysis tools provide quick feedback on performance, they do not inform the designer with any guidance for improving the design. On the other hand, graphic statics instantly generates clear visualization of forces which enables the designers to get a clearer understanding of the structure's internal forces. As a result, the designer's intuition of the relationship between form and forces is improved, and better decisions will be made more quickly as the project progresses.

6.1.2. Unbiased exploration of diverse design alternatives

With automated generation by the computer, which is guided by the design goals input by the human designer, diverse solutions can be generated that simply would not be conceivable manually by a human designer with a pencil or a mouse. In addition, the automated generation of multiple designs at once not only increases the creative capacity of the designer, but also leads to new insights and better understanding of the design problem itself.

6.1.3. Generative graphic statics: beyond reciprocity

The reciprocal relationship between form and forces in graphic statics, means that one has to be created before the other can be drawn. Therefore, most computational graphic statics tools only work with pre-set problems, functioning mostly as an interactive analysis or visualization tool. By combining graphic statics with shape grammars, the form finding capabilities of graphic statics can be used to *generate* equilibrium structures. Most previous work done on shape grammars require a shape to preexist before any rule can be applied. However, the rules presented in this paper are based on *Nodes* or points, and are not dependent on any preceding shapes or conditions. Therefore, the methodology is flexible enough to be applied to a variety of design problems, and is able to generate structures without any prescribed typologies or preferences.

6.2. Future work

Although this paper was a successful first attempt of implementing this new methodology, there are several important directions for future work. First, global parameters could be improved to gain better control of overall generation process, including more intelligent ways in which the rules are chosen and where they are applied. Secondly, more detailed or material-specific constraints, buckling lengths, and minimizing overlapping members could be incorporated. Also, because all designs shown in this paper are also statically equilibrated only for the defined load case, it will be important to develop a procedure to check for possible mechanisms and local instabilities. Lastly, while this paper focused on rules based on the form diagram, rules can also be developed for the force diagram (Akbarzadeh *et al.* [1]), which will further enrich the design possibilities.

6.3. Closing remark

Overall, this new methodology demonstrates the validity in combining and applying shape grammars and graphic statics together to various engineering design problems. The general versatility and customizability of the tool, and the speed at which it can generate unconventional and yet statically equilibrated structures, greatly improves possibilities for creative yet performance-focused explorations during early stages of conceptual structural design.

References

[1] Akbarzadeh M., Van Mele T. and Block P., Compression-only Form finding through Finite Subdivision of the Force Polygon, in *Proceedings of the International Association for Shell and Spatial Structures (IASS) Symposium*, 2014.

[2] Allen E. and Zalewski W., *Form and Forces: Designing Efficient, Expressive Structures*. John Wiley and Sons, New York, 2009.

[3] Baker W., Beghini L., Arkadiusz M., Carrion J. and Beghini A., Maxwell's Reciprocal Diagrams and Discrete Michell Frames. *Structural and Multidisciplinary Optimization*. Springer, 2013.

[4] Cremona L., *Graphical Statics: Two Treatises on the Graphical Calculus and Reciprocal Figures in Graphical Statics*. Claredon Press, Oxford, 1890.

[5] Culmann K., *Die Craphische Statik*. Verlag Meyer & Zeller, Zurich, 1864.

[6] Fivet C., and Zastavni D., Constraint Based Graphic Statics: New Paradigm of Computer-Aided Structural Equilibrium Design. *Journal of the International Association for Shell and Spatial Structures, 2013*; **54**; 271-280.

[7] Greenwold S. and Allen E., Active Statics, 2003. http://acg.media.mit.edu/people/simong/statics/Start.html

[8] Lee J., *Grammatical Design with Graphic Statics: Rule-based Generation of Diverse Equilibrium Structures*. MIT Master of Engineering Thesis, 2015

[9] Mitchell W., Functional Grammars: An Introduction. *Reality and Virtual Reality (ACADIA),* 1991; 167-176.

[10] Mueller C.T., *Computational Exploration of the Structural Design Space.* MIT Doctorate Dissertation, 2014.

[11] Shea K., and Cagan J., Languages and semantics of grammatical discrete structures. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing,* 1999; **13**; 241-251.

[12] Stiny G. and Gips J., Shape Grammars and Generative Specification of Painting and Sculpture. *Information Processing,* 1972; **71**; 1460-1465.

[13] Van Mele T., Block P., Ernst C. and Ballo L., eQUILIBRIUM: An interactive, graphic statics-based learning platform for structural design, 2009-2014. http://block.arch.ethz.ch/equilibrium