Proceedings of the IASS 2024 Symposium
Redefining the Art of Structural Design
August 26-30, 2024, Zurich Switzerland
Philippe Block, Giulia Boller, Catherine DeWolf,
Jacqueline Pauli, Walter Kaufmann (eds.)

Designing strut-and-tie networks by graph theory and local Airy polyhedra

Jihyun KIM*, Rina KIM[a], Sung-gul HONG[b]

*Seoul National University,
Department of Architecture and Architectural Engineering

Abstract

Strut-tie modeling was first devised as a load-path design strategy for engineers, which aims to represent the internal load path with stress resultants. However, since constructing appropriate load paths is not a straightforward task, strut-tie model generation usually relies on automated, optimization-based techniques. This study proposes an alternative design framework of strut-tie models using rigidity circuits, whose topological structure always admit a state of self-stress. In the design workflow, the initial form diagram must be constructed as a rigidity circuit using additional nodes and edges. The correspondence of a rigidity circuit with a unique polyhedral lifting and graph operations preserving its structure are utilized to establish suitable load path modification operations on the local Airy polyhedron. Simple design examples show the robustness of the resulting force network under changes in direction and its ability to represent various sparse load paths. The proposed framework helps designers develop load paths based on their decisions, reinstating the aspects of design in strut-and-tie modeling.

Keywords : strut-tie model, graphic statics, Airy stress function, load path design, graph theory, rigidity theory, Maxwell-Cremona correspondence

1. Introduction

The strut-tie model is a valid load path expressed with internal force resultants equilibrating external loads applied to a domain, following the lower bound theorem of plasticity [1]. It was first introduced as a consistent tool for calculating the ultimate load and designing reinforcement in concrete structures with complex local stress distributions, but its concept is also widespread in other fields of structural design. Strut-tie models can be used to explore and visualize structural design alternatives in different domains, being interpreted as a physical tensegrity structure with bars and cables or an abstract load path in both discrete and continuous domains [2]. Self-stress states in structures including vaults and gridshells can also be seen as a strut-tie model.

1.1. Related Work

There are two main perspectives in generating strut-tie models. First, there are optimization-based methods which share a common objective to minimize the complementary energy or volume of the structure. Layout optimization is usually employed in the discrete setting, where the ground structure is generated and redundant bars are removed using mixed integer linear programming methods [3]. In the continuous domain, topology optimization is used to find the most efficient distribution of materials under the external loads.

On the other hand, there are recent developments in the automatic generation of strut-tie models which are more focused on the diversity of forms and the interactive nature of design [4]. In these generative models, graphic statics becomes an essential tool which helps designers to quickly and intuitively explore diverse solutions in equilibrium. The process of designing force networks with graphic statics usually exploits the reciprocity of form and force diagrams to directly evaluate and modify the edge

forces by the lengths of the corresponding edges in the force diagram. In order to alter the direction and magnitude of the forces, either the force diagram [5] or the form diagram can be the main object for design.

Konstantatou et al. [6] proposed a geometric design workflow of strut-tie models based on polyhedral graphic statics which uses stress functions that are one dimension higher than the form diagram. The classic observation from Maxwell[7] denotes the correspondence between states of self-stress and polyhedral liftings of planar 3-connected bar and joint frameworks. A polyhedral lifting of the form diagram is equivalent to the discrete Airy stress function of a self-stress state in that forces in the bars are proportional to the difference in the gradients of the faces.

In the proposed workflow, the form diagram is first transformed into a self-stressed truss structure and then lifted vertically to form a closed polyhedron. This process of lifting requires identifying if the given form diagram represents a projection of a polyhedron in plane. However, graphs with a relatively small number of edges must satisfy a strict projective geometric condition such as the Desargues configuration to have a polyhedral lifting[8]. For this reason, recognizing the existence of a polyhedral lifting is greatly affected by instabilities in vertex positions. Also, the boundary forces and the topology of the form diagram are unchanged, limiting the concept of 'design' to parallel redrawing of the form diagram.

1.2. Contributions

This paper extends the geometric Airy-polyhedron based procedure for designing strut-tie models by using results from rigidity theory on graphs to provide more tools for the actual manipulation of load paths. A special category of graphs called *rigidity circuits*[9], which always have a self-stress state from its combinatorial structure, is utilized as the building block of the form diagram. Rigidity circuits are closed under combination and decomposition operations which enable engineers to modify internal load paths. In addition, a planar rigidity circuit can always be lifted into an oriented spherical polyhedron. This fact reduces the difficulty of constructing Airy polyhedra and facilitates interactivity in load path design by visually representing the designed results as a geometric object.

This paper mainly contributes to the following improvements in the design workflow:

1) Additional details on the workflow including construction of the initial two-layered Airy polyhedron and local load path modifications
2) Simple demonstrations of the design procedure to explain the capacity of force networks modeled as rigidity circuits to represent both simple and atypical load paths.

In general, any self-stressed force network may be added or subtracted to the form diagram to modify the edge forces and overall load path. However, using rigidity circuits is a more robust approach to find a spanning load path corresponding to a self-stress state which is not locally self-contained[10]. The main limitation of using rigidity circuits is that the family of rigidity circuits is not strictly closed under combinatorial resultant operations [9]. Also, it will generally be difficult to interpret self-intersecting spherical polyhedra resulting from a sequence of design operations.

2. Rigidity circuits, self-stress, and polyhedral liftings

This section explains the main concepts employed from rigidity theory on graphs used in the design workflow proposed in section 3 including rigidity circuits, polyhedral scenes, and combinatorial resultants.

2.1. Rigidity circuits

The topology of a form diagram can be abstracted as a graph G = (V, E). Such graphs can be realized into a bar-joint framework (G,p) by a configuration map $p: V \rightarrow \mathbb{R}^2$ which maps abstract vertices to points in plane[11]. (G,p) is called generic if the vertex coordinates are algebraically independent, i.e. there is no non-zero polynomial with rational coefficients in the coordinates which evaluate to zero[11]. Self-stresses in frameworks are dependent on both the combinatorial graph structure and its geometric embedding[8]. This section mainly investigates the combinatorial properties of self-stressable frameworks.

2.1.1 Laman graphs and rigidity circuits

In two-dimensional rigidity theory, there are certain counting rules on the combinatorial structure of a graph which makes its realizations generically rigid. The family of generically rigid graphs called *Laman graphs* satisfies the following (2,3) sparsity property : $|E| = 2|V| - 3$ and for every subgraph on a subset of vertices $V' \subset V$, $|E'| \leq 2|V'| - 3$ [12] Sparsity of graphs can be conveniently checked using the pebble-game algorithm[13]. Laman graphs are also equivalent to the bases of the *rigidity matroid* of which the ground set is the edges of the complete graph K_n on n vertices[14]. In structural engineering, structures with Laman graph structures are minimally rigid, in that the removal of any edge makes the structure unstable.

A *rigidity circuit* is a minimally dependent set of edges, which satisfies that removing any edge from the graph results in a Laman graph. Formally, a graph G=(V, E) is a rigidity circuit if $|E| = 2|V| - 2$ and for every proper subgraph on a subset of vertices $V' \subset V$, $|E'| \leq 2|V'| - 3$ [12]. Applying the extended Maxwell's counting rule on the count of self-stresses and mechanisms[15] ,

$$e - 2v + 3 = 1 = s - m \qquad (s \geq 0, m \geq 0)$$

where s is the number of self-stress states and m is the number of mechanisms. The topology of rigidity circuits always guarantees one self-stress state in the corresponding framework.

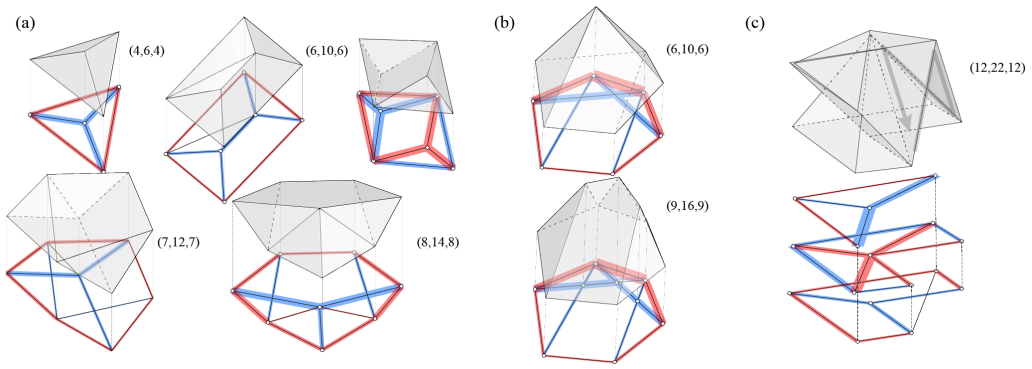

Figure 1 Rigidity circuits and their polyhedral liftings

Typical rigidity circuits with 4 to 8 vertices are shown in Figure 1(a). It should be noted that not all graphs with e=2v-2 are rigidity circuits. Such exceptions always contain a rigidity circuit as a subgraph, which is equivalent to a locally contained self-stress.

In generically embedded rigidity circuits, there should be no edge with zero stress in the induced self-stress. However, an embedding of vertices in $\mathbb{R}^2$satisfying special geometric conditions may violate these properties: when a framework corresponds to a projected picture of a polyhedron, there exists additional unexpected self-stresses and infinitesimal mechanisms[8]. If a subgraph of a rigidity circuit with e<2v-2 satisfies such conditions, the self-stress state will be contained in the subgraph making the stress zero in edges of the complementary graph[16]. Nevertheless, these geometric conditions on the vertex configuration are strict and cannot be satisfied even by small perturbations[17].

2.2. Spherical polyhedra and polyhedral scenes

The correspondence between convex polyhedra and planar 3-connected frameworks with self-stress can be extended to planar 2-connected frameworks, which may be lifted into *spherical* polyhedra including peculiar forms with self-intersections. Any spherical polyhedron can be described with the incidence structure of its vertices and faces. Whiteley[18] shows that a generic framework in plane can be lifted into a unique spherical polyhedron up to lifting equivalence with no non-zero dihedral angle if and only if the underlying graph is a rigidity circuit. This unique lifting fully defines the self-stress state with no redundant edges.

Figure 1 illustrates polyhedral liftings and self-stress states in rigidity circuits. The interior convex and boundary concave edges are defined to be in compression, while the opposites are in tension. Change of

orientation due to self-penetrations and self-intersecting faces should be considered when defining the convexity of edges. In (a), the 2-connected double banana graph with (V,E)=(6,10) always lifts into a polyhedron with two crossing faces. Figure 1(c) shows the lifting of a graph with (V,E)=(12,22) which turns inside-out and has a self-intersecting face. The underlying graph is planar, but embedded with overlapping edges which is separated into three layers for clarity. The resulting stress in the domain can be calculated by adding the stresses of the overlapping edges projected to the same region.

2.3. Combining and decomposing rigidity circuits

Circuits can be combined and decomposed into circuits, just as self-stresses in frameworks can be superposed or separated into distinct self-stresses. Two main graph operations that preserve the properties of rigidity circuits described in Malic and Streinu[9] are summarized here.

1. The *edge-split* operation (*Henneberg-II extension)* enables inductive construction of any 3-connected rigidity circuit. Figure 2(a) explains the sequence of the operation: An edge uv is deleted from the graph, a new vertex a is added, then connected to the existing vertices u, v, w by adding three new edges. The inverse edge-split operation deletes one degree 3 vertex and adds one edge. Both operations preserve the characteristics of rigidity circuits.
2. The *combinatorial resultant* operation on rigidity circuits is the process of combining two rigidity circuits with non-empty intersections and deleting one common edge. In order to preserve the edge count e=2v-2 of the resulting graph, the intersecting subgraph must be Laman[12]. However, rigidity circuits are still not fully closed under combinatorial resultant operations because there are no known conditions under which the (2,3) sparsity property for every proper subgraph is preserved, except for when the common graph is a single edge.

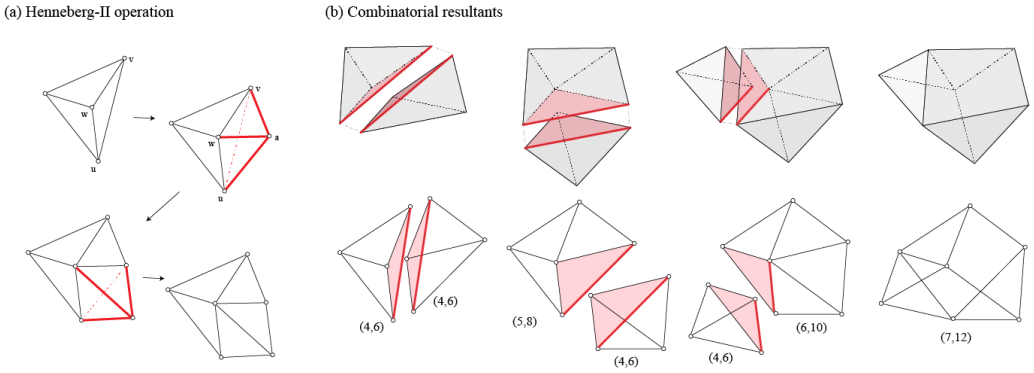

Figure 2 Operations of rigidity circuits and their polyhedral counterparts

According to Malic and Streinu[9], any rigidity circuit can be constructed via combinatorial resultants from the smallest circuit, which is the K4 graph. Also, any rigidity circuit can be decomposed into two circuits whose resultant is the original circuit. This permits the representation of the internal structure of a circuit as a rooted binary tree.

3. Applications to Strut-tie Network design

The proposed workflow is shown in Figure 3. The input data consists of the input domain, boundary conditions, and external load cases. The full input domain is then divided into sub-regions to reduce the complexity of the form diagram and provide more flexibility. Next, the directions and magnitudes of boundary forces are set either fully or partially for each region. Subsequent steps involve constructing the initial form circuit with the Airy stress polyhedron and then applying graph operations mentioned in Section 2, which will be described in detail in the following paragraphs. The step involving saving the operation is necessary to reconstruct the load path when changes in the external forces occur. This step is not detailed in this paper, to be developed in further research.

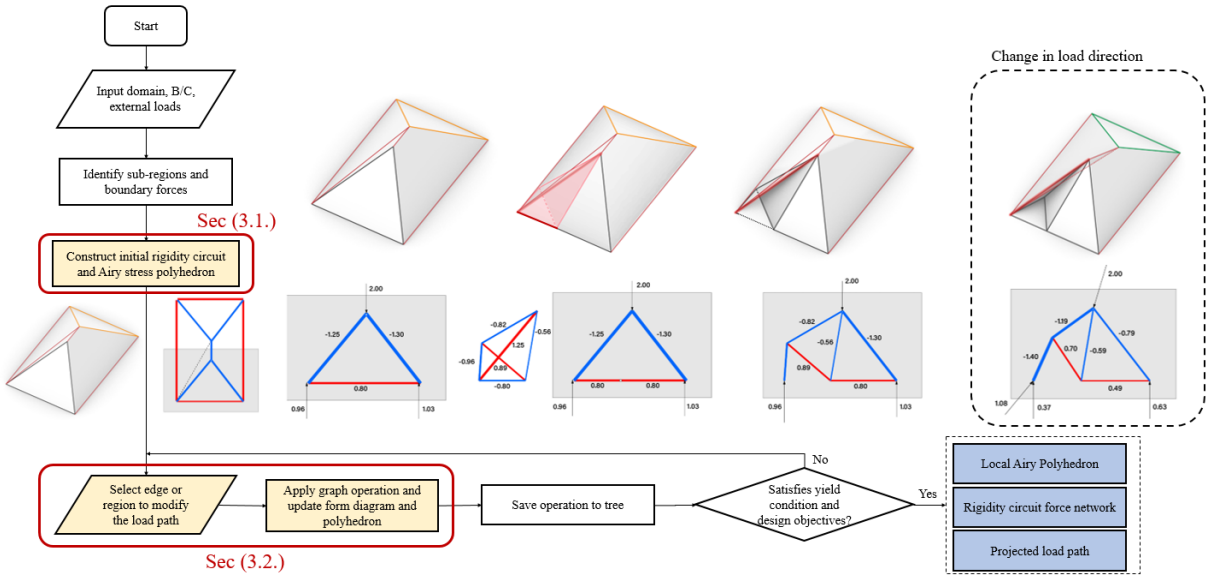

Figure 3 Overview of design workflow using rigidity circuit and local Airy polyhedra

3.1. Construction of initial form diagram

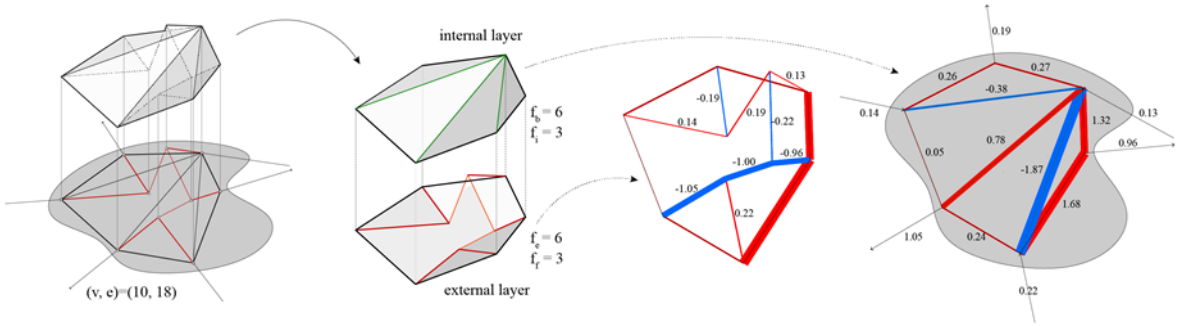

Figure 4 Constructing the twin-layered Airy polyhedron

After designating the boundary forces in a region, nodes and edges must be added to construct a form diagram corresponding to a planar rigidity circuit. To develop a systematic way of adding edges and vertices, the edges are classified into 4 categories: *external force* edges, *funicular* edges, *perimeter edges*, and *internal edges*. The lifted polyhedron is interpreted as a twin-layered Airy stress function as in [19]. The first two categories form the external layer which ensures the equilibrium of boundary forces, while the perimeter and internal edges composing the internal layer correspond to internal load paths. The perimeter edges form the common boundary of the two layers, always forming a closed polygon connecting the points where load is applied. This polygon is typically not coplanar when lifted. Let f_e, f_f, f_p, f_i denote the numbers of the external, funicular, perimeter, and internal edges respectively. From previous steps the number of perimeter and external edges are known. If the direction of all boundary forces are fixed, the two edge counts are equal. The number of internal and funicular edges are determined by counting the number of edges and vertices of the assembled diagram: $e = f_p + f_e + f_f + f_i$ and $v = f_p + f_f + k$. The constant k equals 0 when the funicular polygon is closed and 1 when it is open.

The edge count e=2v-2 and Laman subgraph condition of rigidity circuits imposes constraints on adding funicular and internal edges. One example is when the number of funicular edges is set equal to the external force edges , $f_e = f_f$. In this case, the external layer is in the structure of a polygon surrounded with quads. It is clear that a graph containing this layer cannot be a rigidity circuit, since

removing any edge from this layer would make the graph non-rigid. Equivalently, the graph does not have a corresponding lifting in general, except when the vertices are in special position. To resolve this, an external force edge is added to triangulate one of the quads. Here, a circuit can be constructed upon adding adequate internal edges.

When the directions and magnitudes of all external forces are already computed or predefined to satisfy static equilibrium, the funicular polygon geometry can be easily found from the force diagram. The direction of the edges of the funicular polygon can be obtained by choosing any point in the reciprocal force diagram and coning the force polygon. Although the exact geometry of the previous force layer with quads can be found in this way, the extra force edge is still required to remove instabilities in lifting computations. The chosen point may coincide with the vertices of the force polygon, which reduces the number of funicular edges.

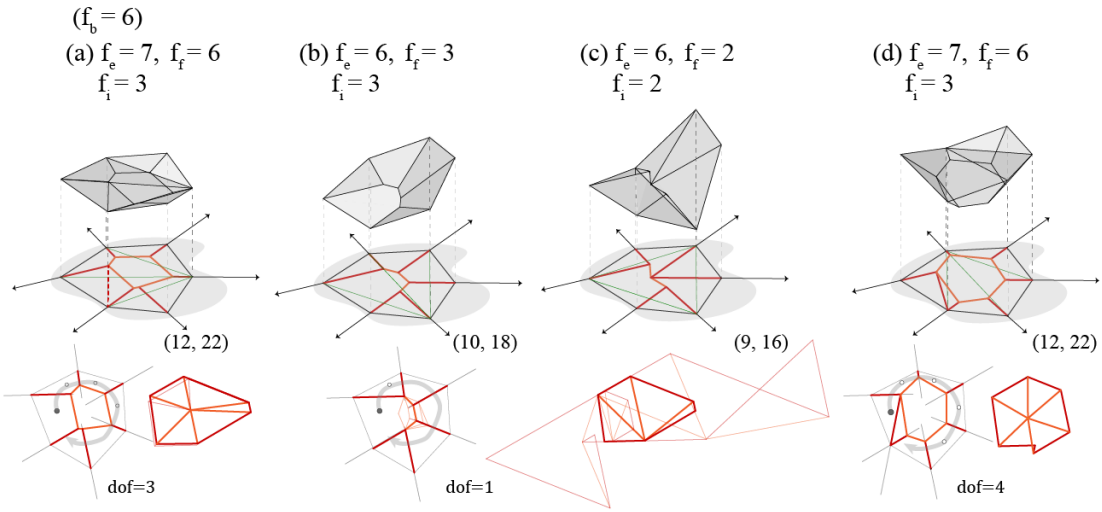

Figure 5 Initial form diagram alternatives of the region in Figure 4

Initial form diagram alternatives and their polyhedral liftings of a sample design domain with 6 boundary points and 6 *known* external forces is illustrated in Figure 5. Once the external force polygon is drawn, the number of funicular forces can be chosen to be 6 as in (a) or 3 as in (b). In (a), an additional external force edge is inserted, whose crease angle in the lifting is approximately 0. These edges would be termed 'inactivated' since the force in the edge is zero in the self-stress of the circuit.

Using rigidity circuits as form diagrams is especially effective when the directions and magnitudes of some external forces are not fixed. The external force edges can be moved around slightly to account for changes in direction. The nodes of the funicular polygon can be moved in the direction of incident external force edges to account for changes in magnitudes only. It is important to note that the chosen topology of funicular forces may limit the 'degree of freedom' of possible equilibrium forces. Consider the form diagrams in Figure 5. Since there are 6 external forces, the degree of static indeterminacy is 3. In 5(b), only two vertices of the open funicular polygon are free to move around in the external force direction. This reduces the force indeterminacy to 2. In contrast, in the form diagram in 5(d) one external force edge is added, with two force edges used to represent the unknown force in one vertex. This underlying topology can model the whole space of admissible equilibrium forces, but increases the number of exact vertex positions to be defined. Careful construction of the funicular edges is essential for simplicity of local polyhedra and robust updating of boundary forces shared by two regions.

3.2. Modification of forces in the internal layer

Unlike the existing policy-based design algorithm in load path design [2], this stage begins from an already connected force network. The perimeter and internal edges are the main objects for load path modification.

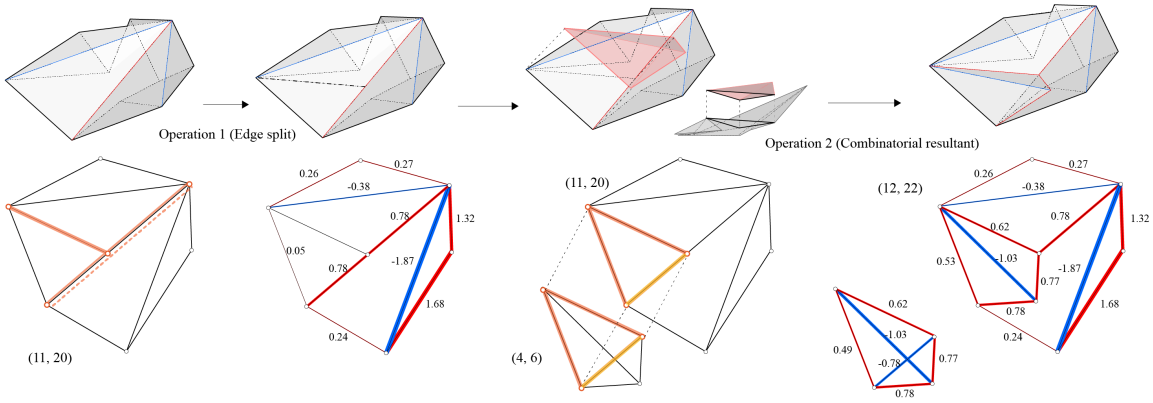

Figure 6 Local modifications of perimeter and internal forces

The graph operations described in section 2 are reformulated to define operations form diagrams embedded in plane. All operations must preserve the circuit properties and planarity of the form graph.

First, the edge-split operation is used to split an edge by a point *on* that edge and add a new edge connecting the point to the original diagram. The new edge must split a face in the internal layer of the Airy polyhedron. These two restrictions are required to avoid complete reconstruction of the polyhedron during the design phase. Namely, if the splitting point is in generic position, this operation might result in non-planar faces and the polyhedron is no longer valid. The newly added edge is inactivated, meaning that the force is zero in the edge in the self-stress state of the circuit. This edge can be activated with following resultant operations. Thus, this operation can be viewed as subdivision of edges to control the lengths of load paths.

Next, the combinatorial resultant operation which combines two circuits is interpreted as replacing a stressed edge with a network of forces equilibrated by the edge. Note that this edge is completely removed from the graph, unlike the inactivated edges mentioned above. In order to maintain planarity of the graph, the common Laman subgraph of the two operands must be planar.

In the polyhedral setting, liftings of circuits can be adjoined to the original polyhedron to remove the dihedral angle corresponding to the deleted edge. This requires that the adjacent faces to the edge in the original and adjoined polyhedra are overlapped. Also, the common Laman subgraph of the original circuit and the circuit to be combined must match the skeleton of connected faces in the Airy polyhedron. Once the common faces are identified, they are deleted from both polyhedra and the remaining faces are glued together on the boundary of the removed faces. The boundaries of the overlapping faces are modified by either addition or subtraction of faces regarding its orientation. The closed spherical polyhedra may be self-intersecting itself or contain self-intersecting faces.

The main challenge is to find a lifting of the added network that contains the same subset of faces. If the common graph is equivalent to three or more faces, then this problem is over-constrained and there might be no solutions. This is because liftings of generic rigidity circuits are uniquely determined when the lifting of one face and one additional information such as the position of a vertex or a dihedral angle of one edge is given [20]. For clarity, we narrow down the cases of common Laman subgraphs to a single edge and a single triangular face. Methods to combine circuits with larger common subgraphs in the polyhedral domain without violating graph planarity are subject to future research.

Figure 6 shows an example of a sequence of form diagram operations applied to the initial polyhedron in Figure 4. The first operation splits the central edge without changing the load path, adding two edges and one vertex to the topology. This edge is then activated by a resultant operation with a K4 graph lifted into tetrahedra.

Some simple strut-tie network alterations using rigidity circuits and Airy polyhedra inspired from typical d-regions in concrete design are illustrated in Figure 7. Initial load paths are modified using both types of operations. In Figure 7(a) (ii), it is notable that the same polyhedron can be obtained from two

different resultant operations, where the common subgraph is respectively set as a triangle and a single edge. Figure 7(b)-(ii) shows overlap in load paths projected to the domain, resulting in the typical strut-tie model of an opening joint.

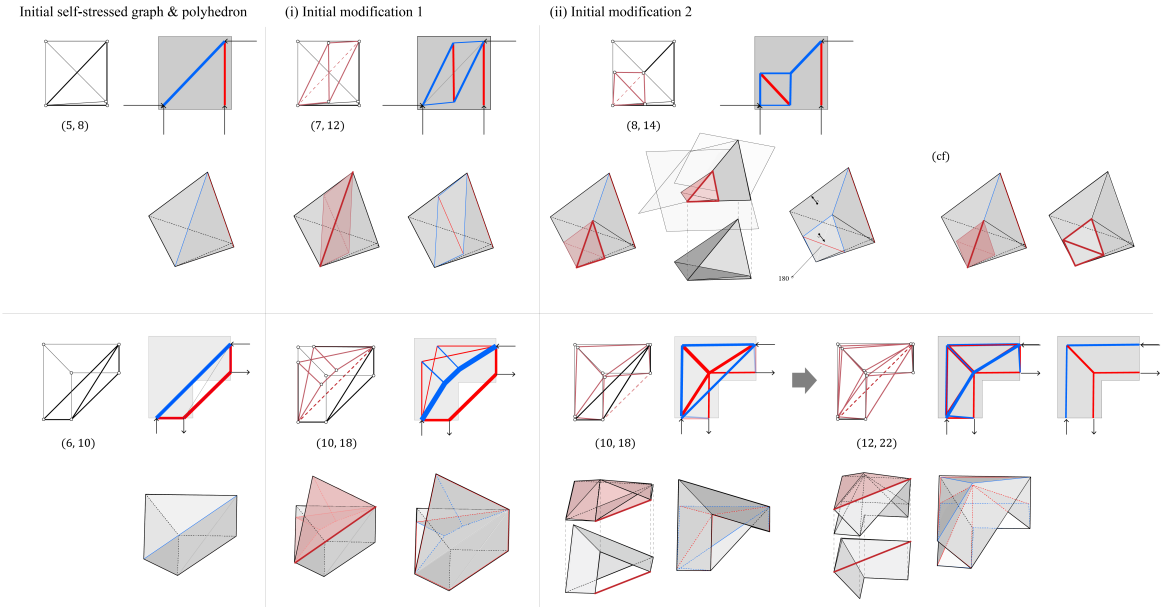

Figure 7 First-step operations on form diagrams of shear panel and joint-type domains

4. Design examples and discussion

Figure 8 shows the application of the proposed framework to generate typical load paths in beams with holes and dapped ends[21, 22]. Initial form diagrams, design alternatives and their corresponding local Airy polyhedra are shown for each domain. Two types of graph operations were performed, preserving the planarity of the graph and the existence of the spherical Airy polyhedron. Once the rigidity circuit force network is determined, designers can move the vertices around to accommodate for changes in the domain boundaries. This provides designers with greater control over the form diagram. While rigidity circuits cannot represent all feasible load paths with higher indeterminacies, simple load paths and stress fields can be determined and updated automatically by directly updating the Airy polyhedron, without iterative calculations. This rigidity circuit-based design framework would encourage a more static-focused conceptual design phase in architectural design practices.

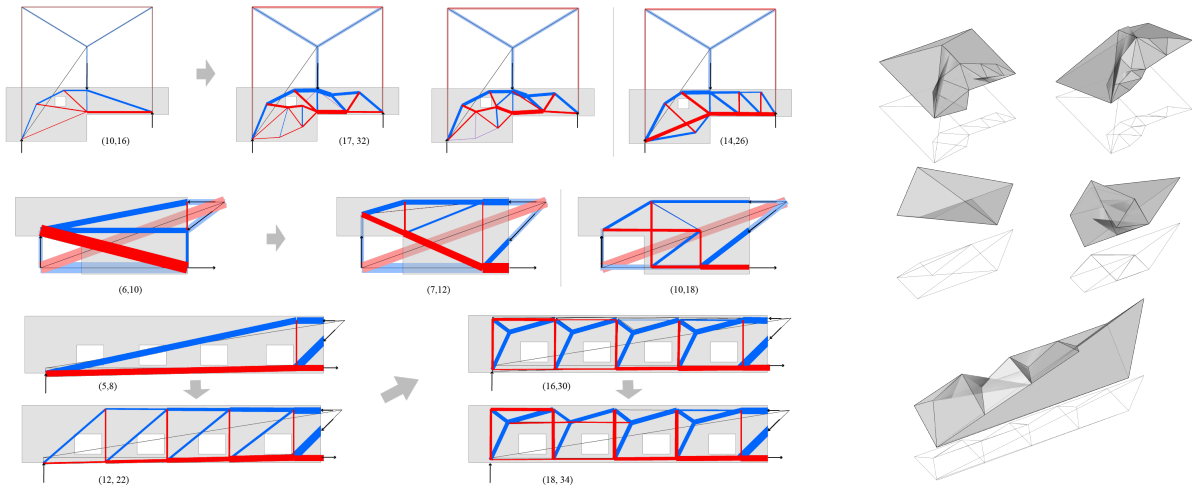

Figure 8 Design example of dapped-ended beams / beams with holes – load paths and Airy polyhedra

Figure 9 shows an example of local load path design in a non-convex, bridge pier-shaped domain. Perimeter forces violating the domain boundaries are removed with three combinatorial resultant operations. The rightmost diagram illustrates the designed load path when the direction of vertical loading is skewed to the right and the direction of one force is reversed, obtained from the same sequence of graph/polyhedral operations. This example demonstrates that force networks modeled as rigidity circuits always admit a state of equilibrium, irrelevant to variations in external load directions.

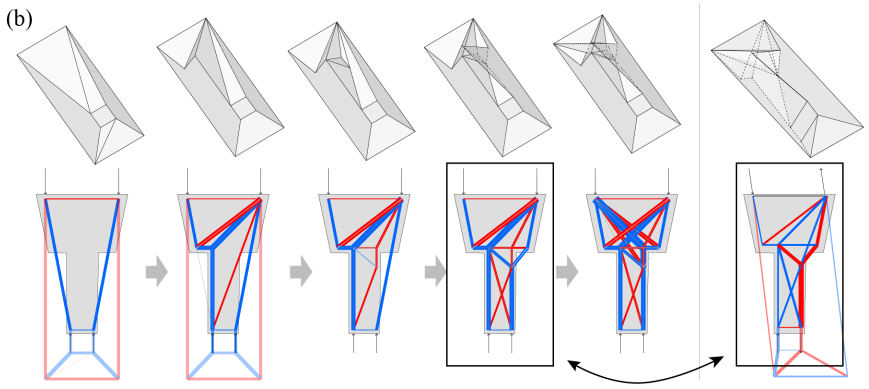

Figure 9 Design example – robustness of rigidity circuit force networks with varying load directions

Designing force networks in multiple sub-regions becomes more complex, as the change in magnitude or direction of external loads in one region would propagate to other regions by shared boundary forces. The number of independent boundary forces shared by two regions determines whether the form diagram and Airy polyhedron must be reconstructed. If there is only one independent force edge connecting two regions, the values can simply be scaled without having to update the network. The exact interaction between external force layers of neighboring sub-regions is planned to be investigated in future studies.

5. Conclusion

This paper extended the existing Airy-stress function-based planar strut-tie design framework using rigidity circuits, spherical polyhedra, and graph operations. The main advantages of new developments include: resilient exploration of load paths with sparse connectivity, direct modification of load paths without iteration, and its robustness under changes in external loads. The next goal is to bring the design framework into the computational domain and provide automated construction and updates of form diagrams, along with geometric manipulations that interactively updates the stored circuit tree data.

There are several theoretical and practical limitations in this framework. Above all, it must be clarified how much the choice of the topology and geometry of the funicular affects the space of equilibrium forces compatible with the form diagram. Regarding internal manipulations, further study in rigidity theory and graph composition techniques would help develop more clear rules concerning graph operations and reduce ambiguity in the topological properties of resulting graphs. Also, projective transformation of Airy polyhedra is worth investigating, which may provide insights into modeling networks with multiple forces in the same lines of action.

The load paths designed by this method may be considered inferior to optimization-based models in terms of structural performance. Nevertheless, the concept of stress decomposition into smaller circuits shows the possibility of integrating the proposed framework to such models to explore close-to-optimal solutions by manipulating local circuits. Overall, the proposed design approach deepens designers' insights into the states of equilibrium in structures, in that they can try combining stressed networks and inspect the impact of the operation through polyhedral representations.

Acknowledgements

This work was supported by the BK21 FOUR (Fostering Outstanding Universities for Research) Project in 2024. (No.4120200113771) The Institute of Engineering Research at Seoul National University provided research facilities for this work.

References

[1] J. Schlaich, K. Schafer, and M. Jennewein, "Toward a Consistent Design of Structural Concrete," *PCI J.*, vol. 32, no. 3, pp. 74–150, May 1987, doi: 10.15554/pcij.05011987.74.150.
[2] C. Fivet, "Constraint-Based Graphic Statics: A geometrical support for computer-aided structural equilibrium design," UCLouvain, 2013. [Online]. Available: http://infoscience.epfl.ch/record/218830
[3] W. S. Dorn, R. E. Gomory, and H. J. Greenberg, "Automatic design of optimal structures," *J. Mec.*, vol. 3, pp. 25–52, 1964.
[4] I. Mirtsopoulos and C. Fivet, "Structural topology exploration through policy-based generation of equilibrium representations," *Comput.-Aided Des.*, vol. 160, p. 103518, Jul. 2023, doi: 10.1016/j.cad.2023.103518.
[5] M. Akbarzadeh, T. V. Mele, and P. Block, "Compression-only Form finding through Finite Subdivision of the Force Polygon," in *Proceedings of IASS Annual Symposia*, 2014.
[6] M. Konstantatou, P. D'Acunto, A. McRobie, and J. Schwartz, "Unified geometrical framework for the plastic design of reinforced concrete structures," *Struct. Concr.*, vol. 21, no. 6, pp. 2320–2338, Dec. 2020, doi: 10.1002/suco.201900440.
[7] J. C. Maxwell, "I.— *On Reciprocal Figures, Frames, and Diagrams of Forces*," *Trans. R. Soc. Edinb.*, vol. 26, no. 1, pp. 1–40, 1870, doi: 10.1017/S0080456800026351.
[8] N. L. White and W. Whiteley, "The Algebraic Geometry of Stresses in Frameworks," *SIAM J. Algebr. Discrete Methods*, vol. 4, no. 4, pp. 481–511, Dec. 1983, doi: 10.1137/0604049.
[9] G. Malić and I. Streinu, "Combinatorial Resultants in the Algebraic Rigidity Matroid," 2021, doi: 10.48550/ARXIV.2103.08432.
[10] B. Schulze, C. Millar, A. Mazurek, and W. Baker, "States of self-stress in symmetric frameworks and applications," *Int. J. Solids Struct.*, vol. 234–235, p. 111238, Jan. 2022, doi: 10.1016/j.ijsolstr.2021.111238.
[11] A. Nixon, B. Schulze, and W. Whiteley, "Rigidity through a Projective Lens," *Appl. Sci.*, vol. 11, no. 24, p. 11946, Dec. 2021, doi: 10.3390/app112411946.
[12] G. Malic and I. Streinu, "Computing Circuit Polynomials in the Algebraic Rigidity Matroid." arXiv, Apr. 24, 2023. Accessed: Mar. 16, 2024. [Online]. Available: http://arxiv.org/abs/2304.12435
[13] I. Streinu and L. Theran, "Sparse hypergraphs and pebble game algorithms," *Eur. J. Comb.*, vol. 30, no. 8, pp. 1944–1964, Nov. 2009, doi: 10.1016/j.ejc.2008.12.018.
[14] W. Whiteley, "Some matroids from discrete applied geometry," in *Contemporary Mathematics*, vol. 197, J. E. Bonin, J. G. Oxley, and B. Servatius, Eds., Providence, Rhode Island: American Mathematical Society, 1996, pp. 171–311. doi: 10.1090/conm/197/02540.
[15] C. R. Calladine, "Buckminster Fuller's 'Tensegrity' structures and Clerk Maxwell's rules for the construction of stiff frames," *Int. J. Solids Struct.*, vol. 14, no. 2, pp. 161–172, 1978, doi: 10.1016/0020-7683(78)90052-5.
[16] M. Konstantatou, "Geometry-based structural analysis and design via discrete stress functions," Sep. 2019, doi: 10.17863/CAM.50698.
[17] K. Sugihara, "Machine Interpretation of Line Drawings," in *MIT Press series in artificial intelligence*, 1986. [Online]. Available: https://api.semanticscholar.org/CorpusID:5946010
[18] W. Whiteley, "Motions and Stresses of Projected Polyhedra," *Struct. Topol.*, no. núm. 7, 1982.
[19] A. McRobie, W. Baker, and T. Mitchell, "Mechanisms and states of self-stress of planar trusses using graphic statics, Part III: Applications and extensions," *Int. J. Space Struct.*, pp. 102–111, 2015, doi: 10.1177/0266351116660791.
[20] W. Whiteley, "How to describe or design a polyhedron," *J. Intell. Robot. Syst.*, vol. 11, no. 1–2, pp. 135–160, Mar. 1994, doi: 10.1007/BF01258299.
[21] A. Muttoni, J. Schwartz and B. Thürlimann, Design of Concrete Structures with Stress Fields. Springer Science & Business Media, 1996.
[22] Y. Xia, M. Langelaar, and M. A. N. Hendriks, 'Automated optimization-based generation and quantitative evaluation of Strut-and-Tie models', Computers & Structures, vol. 238, p. 106297, Oct. 2020, doi: 10.1016/j.compstruc.2020.106297.