A uniform characterization of augmented shapes

Rudi Stouffs[1*] and Ramesh Krishnamurti[2]

[1] National University of Singapore, School of Design and Environment, Department of Architecture, 4 Architecture Drive, Singapore 117566, Singapore

[2] Carnegie Mellon University, School of Architecture, 5000 Forbes Avenue, College of Fine Arts 201, Pittsburgh, PA 15213

Shapes are considered as finite arrangements of spatial elements from among points, line and plane segments, circles and ellipses, (circular) arcs, quadratic Bezier curves, of limited but non-zero measure, in 2D and 3D. Augmented shapes are defined as shapes augmented with attributes, e.g., labels, weights, colors, enumerative values, and (parametric) descriptions. Different attribute types specify different behaviors under operations of sum, product and difference and a part relationship. We review different shape attribute propositions from the shape grammar literature and characterize them uniformly. This uniform characterization of augmented shapes is intended to assist in formalizing new shape attribute propositions that may have been visually conceived.

Highlights:

- A uniform characterization of the behavior of different attribute types is presented under operations of sum product and difference, and a part relationship.
- A uniform description of the behavior of augmented shapes, i.e., shapes with attributes, is presented under the same operations and relationship.

---

[*] Corresponding author

1 Introduction

We consider shapes that are imbued with (visual) attributes, for instance, color, thickness, texture, labels, etc., and examine how such augmented shapes can be uniformly characterized so as to lend themselves amenable to computation. Consider for example a collection of three shape rules to create a spiral of inscribed squares (Figure 1) [1]: a first rule creates a square from an initial marker; a second rule creates a rotated square inscribed within the original square, and within squares produced thereof; and a third rule removes the marker.
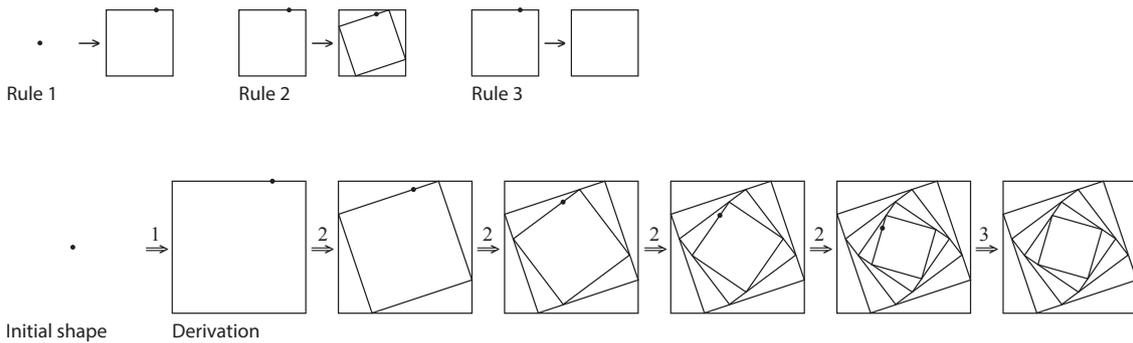

Figure 1: A shape grammar comprised of three rules, generating recursively inscribed squares (redrawn from [1]).

Now, consider a variation on this computation that produces the same spiral of inscribed squares, however, with an alternating infill of the squares in black and white (Figure 2) [2]. The collection of rules to produce such computation can be derived from the collection of rules in Figure 1. However, the specific representation adopted for the infill values must be taken into account in the process and may influence the outcome of this process. We illustrate this with two alternative representations for the alternating infill, the first enumerates the values "black" and "white" and considers a discrete behavior, the second interprets the surface tones as numeric weights and considers an ordinal behavior [3].
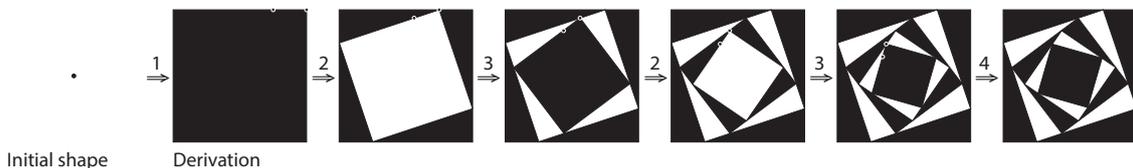

Figure 2: An alternative computation, generating recursively inscribed squares with alternating black and white infill.

In the case of the attribute algebra enumerating the values "black" and "white", the rule inscribing a rotated square within the original square will have two versions, one identifying a

"black" square and inscribing a "white" square, and one identifying a "white" square and inscribing a "black" square (Figure 3a). Under the discrete behavior for enumerative values, matching either rule onto a given shape requires the enumerative value specified in the left-hand-side of the rule to be identical to the enumerative value for the square with the marker in the given shape. Applying the matched rule involves replacing the matched left-hand-side of the rule with the right-hand-side of the rule under the same transformation.

In the case of using numeric weights under an ordinal behavior, matching requires the numeric value specified in the left-hand-side of the rule to be less than or equal to the numeric value for the square with the marker in the given shape. Assuming higher numeric values for darker surface tones, the numeric value for white will be less than or equal to the numeric values of both white and black, thus, a white square in the left-hand-side of the rule will match either a white square or a black square in the given shape. Conversely, if two rules are only distinguished by the numeric value assigned to the square, corresponding to either black or white, both rules will match a given shape having a black square with marker, and an alternating infill will no longer be guaranteed. This problem may be resolved by also considering a tone weight for the marker, more specifically, the opposite tone weight of the respective square (Figure 2b).

(a) shape grammar using enumerative colors as attributes
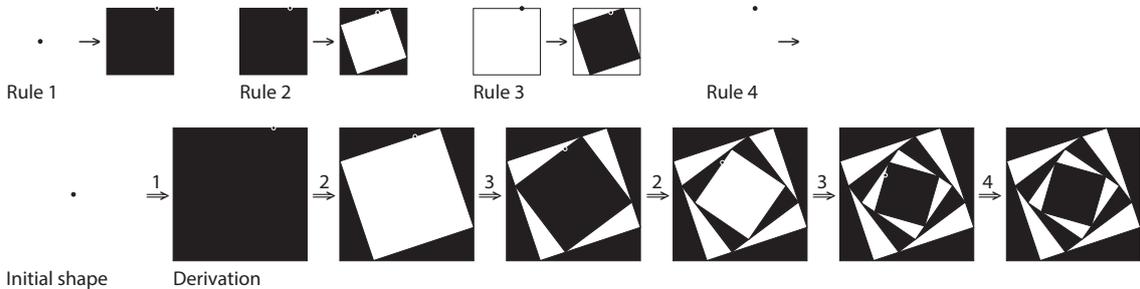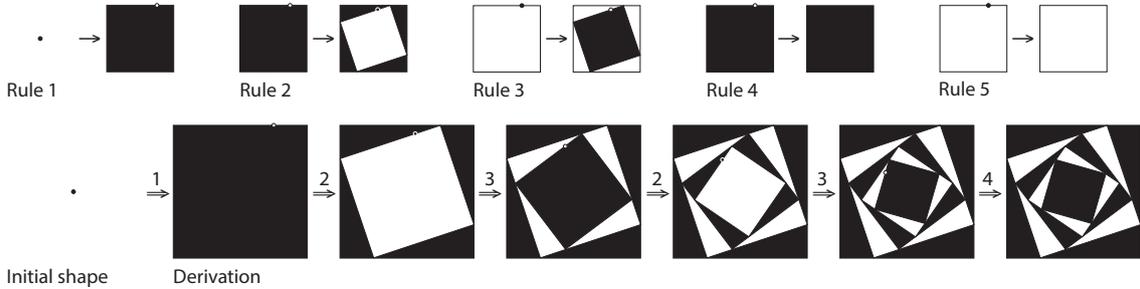

(b) shape grammar using weights as attributes

Figure 2: Two shape grammars generating recursively inscribed squares with alternating infill: (a) using enumerative colors as shape attributes (b) using numeric weights as shape attributes.

In both cases, the shape attributes are visualized as black and white infills. A white segment, or point, is indicated by a lightly drawn outline in order to distinguish it from the background.

The example above may be considered as an albeit simple example of generative design. While many authors have attempted to define generative design (see [4] for a synoptical review), from a historical point of view, analytical form finding techniques broadly led to form-finding algorithms and algorithmic design approaches (e.g., [5]). Even parametric or associative modelling can be considered as a form of algorithmic design, combining both visual programming and dataflow modelling [6]. Bohnacker et al. [7] equate algorithms to rules or rule sets, in the context of generative design in the Processing environment. Conceptually this equation makes sense, rules are a convenient way of expressing design moves or actions and their underlying principles in a structured form [8], especially in the context of design derivation [9]. "Designers preferentially use past successful moves, their own or others, in future projects. One of the long promises of grammars is the ability to explicitly encode such moves" [9].

Grammar formalisms for design come in a large variety (e.g., [10]-[14]), requiring different representations of the entities being generated, and different interpretative mechanisms. Shape grammars also come in a variety of forms, even if less broadly. Most examples of shape grammars rely on labeled shapes, a combination of (2D) line segments and labeled points [11]. However, even in the original conception of shape grammars [15], an iconic shape (made up of curved lines) serves the role of non-terminal marker rather than labeled points, and a colored infill of the resulting shapes is considered part of the generative specification, though not of the shape grammar.

Next to labels, other non-geometric attributes have been considered for shapes. Stiny [3] proposes numeric weights as attributes to denote line thicknesses or surface tones. Knight [16]-[17] considers an extension to the shape grammar formalism that allows for a variety of qualitative aspects of design, such as color, to be integrated in the rules of a shape grammar. Though not specific to colors, the resulting grammar is called a color grammar and notions of transparency, opacity and ranking are introduced to regulate the behavior of interacting quality-defined areas or volumes.

Sortal grammars [2] [18] take this one step further. *Sortal* grammars are a class of formalisms for design grammars, utilizing *sortal* structures as representational structures, where these structures are defined as formal compositions of other, primitive, *sortal* structures, termed *sorts*. *Sortal* grammars benefit from the fact that every component *sort* specifies a partial order

relationship on its individuals and forms, defining both the matching operation and the arithmetic operations for rule application. In addition, whereas in all other formalisms the augmented shapes have been derived from shapes of spatial elements by associating symbols, labels or other qualitative aspects to the elements, under a shape-attribute relationship, in *sortal* grammars, shapes may be either the object or the attribute in the relationship, or both (or neither, though such examples do not constitute spatial grammars as such).

A *sortal* grammar interpreter, denoted SortalGI, has been developed in the form of a library and API in the Python programming language (www.sortal.org). Using *sortal* structures as the representational building blocks allows for a modular implementation of a generalized shape grammar interpreter for different grammar forms. The SortalGI library supports both parametric and non-parametric shape grammars, including points, line and plane segments, circles and ellipses, (circular) arcs, quadratic Bezier curves, labels, weights, colors, enumerative values, and (parametric) descriptions, in 2D and 3D. Emergence is naturally supported. Note that the SortalGI library adopts a graph-based representation for parametric shapes, however, different from other graph-based implementations [19]-[21], it does not use any sub-graph matching algorithm but instead relies on a combinatorial enumeration of potential matches. In general, graph-based, parametric subshape recognition is non-polynomial, even with a hypothetical, linear time subshape detection algorithm [22]. In comparison, a combinatorial enumeration, searching for k maximal elements within a set of n (distinguishable) maximal elements, yields a tight bound of $O(nk)$. Depending on the size of k, this bound is exponential in the worst case, while one can use labels to limit the combinatorial explosion.

While SortalGI offers a modular implementation of a shape grammar interpreter and can be extended with new *sorts*, this requires a uniform characterization of *sorts*. Previous research has dealt with extending the maximal element representation for shapes to (rectilinear) plane and volume segments [23] and (quadratic Bezier) curves [24]-[25].

Instead, Stouffs [26] demonstrates how an algebra with carrier $\wp(A \times \wp(B))$ and signature including sum, product, difference, and reduction can be defined in terms of the (attribute) algebra with carrier $\wp(B)$ and signature including sum, product, difference, and reduction. Here, the carrier of the algebra denotes the set of elements of the algebra. A and B denote any two vocabularies, for instance, the set of all points P and the set of all labels L. In the case of P and L, $\wp(P)$ and $\wp(L)$ denote the powerset of points and labels, respectively, and $\wp(P \times \wp(L))$ the powerset of points where each point may have a set of labels as attribute. The signature of the algebra denotes the operations of the algebra and includes a reduction operator, which reduces any set to a set of maximal elements (see Section 3). The part relation ('$\leq$') is not

considered an operation of the algebra but can be expressed in terms of the operation of product ('·'): $a \leq b \Leftrightarrow a \cdot b = a$. Finally, the similarity transformations can be considered external to the algebra, as part of the algebra's signature or as part of the algebra's carrier [27]-[28]. This definition is argued to be independent of the specifics of the (attribute) algebra with carrier $\wp(L)$, while it is dependent on the *behavior* of the algebra with carrier $\wp(P \times \wp(L))$, which is similar to the behavior of the algebra with carrier $\wp(P)$. An algebra of points has a discrete behavior, that is, the operations of sum, product, and difference correspond to the normal set operations of union, intersection, and difference. Under a discrete behavior, any set is maximal because any duplicates are automatically removed (see Section 3).

Augmentation of shapes is not limited to labels. Stiny [3] proposes numeric weights as attributes to denote line thicknesses or surface tones (Figure 2b). Knight [16]-[17] considers a variety of qualitative aspects of design, such as color, as shape attributes (e.g., Figure 2a). Stiny [11] also proposes to augment a shape grammar with a description function in order to enable the construction of verbal descriptions of designs. Although most authors do not consider descriptions as attributes to shapes, in his thesis, Beirão [1] specifically considers descriptions as attributes to spatial objects. Other kinds of attributes, or even variations in the specification of a kind of attribute, can also be considered. For example, colors can be specified in different ways, as in a three-dimensional RGB or HSV (Hue, Saturation, Value) space, or in an enumerative way as Knight [16]-[17] considers. Labels may adhere to a strict textual format that allows for a different part relationship to be distinguished, for example, dates that can be chronologically ordered, or time intervals that may contain one another. Obviously, dates (and time intervals) may also be considered as numeric attributes, while visualized textually. For these reasons, a uniform characterization of augmented shapes becomes important, such that new or different augmentations can be considered as part of augmented shape algebras. In particular, the simple illustrative examples in Figure 2 suggest a need for a uniform characterization for both enumerative values and weights allowing their respective behaviors to be easily compared and the rule sets adapted accordingly.

While we identify this issue from the viewpoint of shape grammars, the argument can easily be extended to design, and computer aided design, in general, especially for architecture. Eastman [30] recognizes that any human-machine system to aid the designer must recognize the designer's reliance on multiple representations. Design problems commonly require a multiplicity of views each distinguished by particular interests and emphases, where each view—derived from an understanding of current problem solution techniques in the respective domain—requires a different representation of the same (abstract) entity. Research in cognitive science and design cognition has shown that expertise in both problem solving and design often

relates to having access to more and better representations [31]-[32]. This is especially true in architecture, Akin refers to architecture in this respect as a "representation saturated problem domain" [33]. Importantly, the outcome of the design process relates to the representation that is used. Furthermore, architectural design differs from other forms of problem solving in that the problems in architecture are generally ill-structured; defining the problem space is an intricate part of the design activity [32]. As the problem shifts during the design process, so should the representation adapt. Design representations may be as much an outcome of as a means to the design process [1]. While there has been continued and concerted effort in developing integrated product models that span multiple disciplines, multiple methodologies and support different views (e.g., [35]), these do not support the idiosyncrasies of individual designers or projects. On the other hand, modeling schemes for defining product models and ontologies exist (e.g., [36]-[37]) that allow for the development of representations, but these are not supported by current computer aided design tools. A uniform characterization of shape attributes may allow for a more flexible adaptation of computer aided design tools to specific needs of individual designers or projects with respect to shape attributes.

Below, we review different shape attribute propositions and characterize them uniformly. We start with an algebraic introduction of shapes and attributes (Section 2), followed by an exploration of the partial order relation underlying the algebraic representation of shapes and a questioning of its applicability to shape attributes (Section 3). We then express the maximal element representation for shapes as a behavioral specification under the operations of sum, product and difference and the subshape relationship, depending on the dimensionality of the shape's spatial element, and, similarly, suggest behavioral specifications for augmented shapes considering different attribute types (Section 4.1). From this, we extract a uniform characterization of attribute behaviors that enables a unique behavioral expression for augmented shapes in terms of this uniform characterization and the behavioral specification for shapes earlier defined (Section 4.2). Finally, we explore the limitations of this uniform characterization from an implementation point of view and from the similarity with an algebraic abstraction of augmented shapes [26] (Section 5).

2 Shapes and attributes

For our purpose, a shape is defined as a finite arrangement of spatial elements from among points, line segments, rectilinear plane and volume segments, circles, ellipses, (circular) arcs and quadratic Bezier curves, of limited but non-zero measure. This definition can be extended to include other kinds of curves, surfaces and solids.

A shape is considered an element of an *algebra U* that is (partially) ordered by a part relation and closed under the operations of sum, product, and difference, and the similarity transformations. Limiting shapes to compositions of rectilinear hyperplane segments [3], U_{ij} denotes the set of all finite arrangements of i-dimensional hyperplane segments of limited but non-zero measure in a j-dimensional space, $j \geq i$. If j is unambiguously understood, it may be dropped and U_{ij} can be referred to as U_i. Thus, U_0 refers to an algebra of shapes made up of points, U_1 an algebra of shapes made up of line segments, U_2 an algebra of shapes made up of plane segments, U_3 an algebra of shapes made up of volume segments, and so on. Extending on the concept of U_{ij}, or U_i, we may consider U_a (or U_{aj}) an algebra of shapes made up of (circular) arcs, U_b an algebra of shapes made up of quadratic Bezier curves, U_c an algebra of shapes made up of circles, and U_e an algebra of shapes made up of ellipses. Note that while circles can be considered a particular subset of arcs, or ellipses, distinguishing U_c may still be useful in order to limit the elements in the grammar, or part thereof, to circles. A shape may consist of more than one type of spatial element, in which case it belongs to the algebra given by the Cartesian product of the algebras of its spatial element types [1]. For example, shapes made up of points and line segments belong to the algebra $U_0 \times U_1$.

A shape can be augmented by distinguishing certain parts of the shape, which introduce additional spatial relations. The most common definition of a shape grammar [10] uses labeled points as non-terminals. In this approach, points are distinguished by labels and these are used to constrain rule application. Stiny [3] introduces an algebra V_{ij} denoting the set of all finite arrangements of i-dimensional hyperplane segments of limited but non-zero measure in a j-dimensional space, with associated labels. The algebra V_{ij} has (about) the same property as U_{ij}; it is ordered by a part relation and closed under the operations of sum, product, and difference, and the similarity transformations. Omitting the dimension of the space, V_0 refers to an algebra of shapes made up of labeled points, V_1 an algebra of shapes made up of labeled line segments, and so on; again, we can extend on this with V_a an algebra of shapes made up of labeled arcs, and so on. We could be tempted to consider the Cartesian product of algebras to also apply when conceiving the algebra of augmented shapes, e.g., given a set L of labels, $V_{ij} = U_{ij} \times \wp(L)$. However, the relationship between spatial elements of different algebras, e.g., in $U_0 \times U_1$, on the one hand, and between spatial elements and associated (non-spatial) elements, e.g., in V_0, on the other hand, is quite different. The former is a relationship of coordination and disjunction; any spatial element in a shape of points and line segments is either a point or a line segment. The latter, instead, is a relationship of subordination and (partial) conjunction; a labeled point consists of a point and an associated set of labels. The set of labels may be empty, but the point cannot.

3   A partial order relation

Algebraically, shapes have the structure of a Boolean ring [1]. For a shape algebra Σ, $\Sigma \equiv (\Sigma, 0, \leq, (+, \cdot, -, \otimes))$, the least element 0 is the empty shape, $\leq$ is the part relation and $+$, $\cdot$, $-$ and $\otimes$ are the following operations: for any shapes x and y, sum $x + y$ is their least upper bound, product $x \cdot y$ is their greatest lower bound, difference $x - y$ is the least shape z that solves the equation $x - z = x \cdot y$, and symmetric difference $x \otimes y$ is the shape given by $(x - y) + (y - x) = x + y - x \cdot y$. It follows that $x - y \leq x \leq x + y$, $y - x \leq y \leq x + y$, $x \cdot y \leq x$ and $x \cdot y \leq y$, additionally, $x - y \leq x \otimes y \leq x + y$ and $y - x \leq x \otimes y \leq x + y$, and the part relation is a partial order relation, that is, it is reflexive, anti-symmetric and transitive.

It is tempting to consider attribute algebras to necessarily adhere to the same structure. However, most authors conceive attributes' behavior visually, rather than structurally, and such visual reconstruction may or may not fit the exact structure of a Boolean ring. For example, Stiny [3] conceives a behavior for weights (e.g., line thicknesses or surface tones) as is apparent from drawings on paper—a single line drawn multiple times, each time with a different thickness, appears as if it were drawn once with the largest thickness, even though it assumes the same line with other thicknesses.

It obviously follows that for two weights u and v, $u + v = \max(u, v)$ and $u \cdot v = \min(u, v)$. It is less obvious to define $u - v$. If we consider the equation above, $u - (u - v) = u \cdot v$, we must distinguish the cases $u \leq v$ and $u > v$. For $u \leq v$, $u \cdot v = u$ and, thus, the least shape $u - v$ for which the equation holds is 0 (Figure 3 left). For $u > v$, $u \cdot v = v < u$ and, thus, the least shape $u - v$ for which the equation holds is the arithmetic difference of u and v [3] [39] (Figure 3 middle). On the other hand, Stouffs [1] defines the difference operator in opposition to the sum operator—where adding a smaller value leaves the existing value unchanged, subtracting a smaller value similarly leaves the existing value unchanged—, $u -_+ v = u$ if $u > v$, and 0 otherwise (Figure 3 right). However, this can be proven to contradict the structure of a Boolean ring.
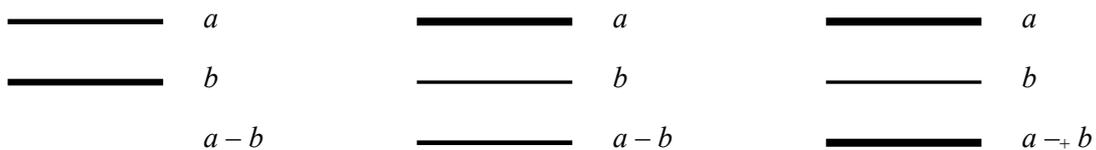
Figure 3: Subtracting two line segments with different thicknesses: (left) subtracting an equal or larger thickness results in a line with 0 thickness, thus, no line; (middle) subtracting a smaller

thickness adhering to arithmetic difference; (right) subtracting a smaller thickness leaves the existing thickness unchanged.
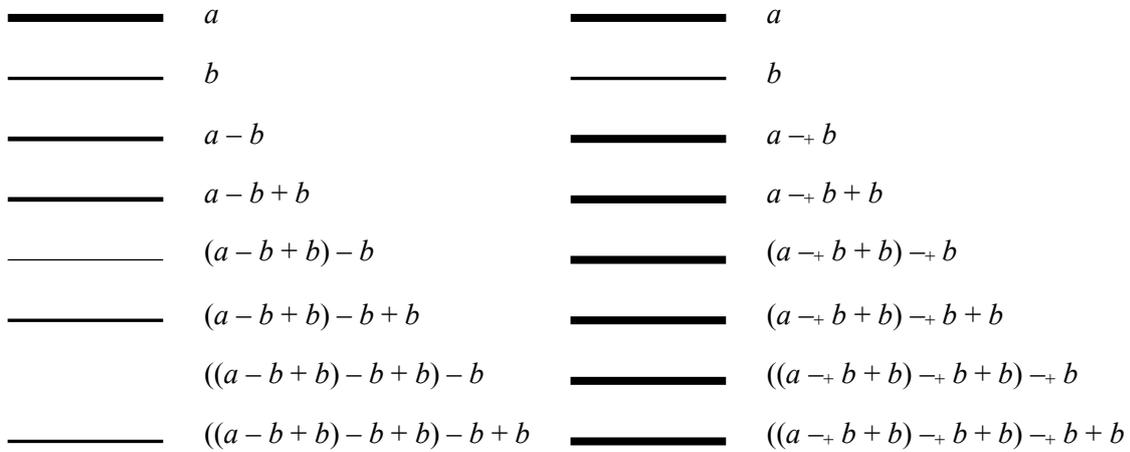
——	a	——	a
——	b	——	b
——	$a - b$	——	$a -_+ b$
——	$a - b + b$	——	$a -_+ b + b$
——	$(a - b + b) - b$	——	$(a -_+ b + b) -_+ b$
——	$(a - b + b) - b + b$	——	$(a -_+ b + b) -_+ b + b$
	$((a - b + b) - b + b) - b$	——	$((a -_+ b + b) -_+ b + b) -_+ b$
——	$((a - b + b) - b + b) - b + b$	——	$((a -_+ b + b) -_+ b + b) -_+ b + b$

Figure 4: Repeatedly subtracting and adding a line segment with thickness b from another line segment with original thickness a, where $a > b$: (left) using arithmetic difference; (right) subtracting a smaller thickness leaves the existing thickness unchanged.

We can interpret the difference between the two approaches visually. Given a line segment with thickness a, let us subtract the same line segment with thickness b, with $b < a$ (Figure 3). Next, we add the same line segment with thickness b back to the result (Figure 4). Under arithmetic difference, subtracting thickness b from thickness a always yields a smaller value, $a - b$, while adding thickness b back to the result will produce the maximum value of $a - b$ and b. If $a - b > b$, we can repeat the same operations resulting in the maximum value of $a - b - b$ and b. Eventually, the result will become equal to b, subtracting thickness b will return 0 and adding thickness b will give us b again (Figure 4 left). Instead, considering Stouffs' [1] definition, subtracting thickness b from thickness a has no impact, neither has adding thickness b back to the result (Figure 4 right). The line segment retains the thickness a throughout these operations, whether repeated or not. Which is more attractive, visually, may depend on the designer's expectations and intent. On the one hand, considering that $(a - b) + b = b$ if $a \leq b$—this holds for both approaches—then it may be quite logical to assume $(a - b) + b$ will eventually yield b, even if $a > b$. On the other hand, considering that $(a - b) + b = a$ if $b = 0$, it may seem odd that an infinitesimally small b, though different from 0, will at first yield a result infinitesimally close to a; yet, after repeated applications of the two operations will eventually result in a value far different from a, specifically the infinitesimally small value b. Either approach might be equally valid, in differing situations. Considering that weights can be interpreted as line

thicknesses, but also surface tones, we might even consider one solution for line thicknesses, and another one for surface tones, or other application of weights.

This need for flexibility becomes even more apparent when we consider the behaviors Knight [16]-[17] conceives for colors and materials. Knight applies colors to plane segments (also denoted regions [17] or fields [16]) and overlapping colored plane segments are handled formally with rankings [17]; for example, an "opaque" ranking implies that any colored plane segment that is added in a rule application covers any part of a colored plane segment already in the design (Figure 5). That is, under an opaque ranking, the operation of sum is not considered commutative, $x + y$ may be different from $y + x$. More importantly, an opaque ranking no longer supports a partial order relation. Under a partial order relation, if, for two colors c and d, we write $c + d = d$, then it necessarily follows that $c \leq d$. However, under an opaque ranking, we also write $d + c = c$, as the color d covers the color c where it is applied. From this, we should conclude that $c \geq d$, or $c = d$. Obviously, that is not always the case. Note that if we consider the fact that any part of a plane segment can have only one color assigned, the partial order relation would become a total order relation, however, this doesn't alter the findings.
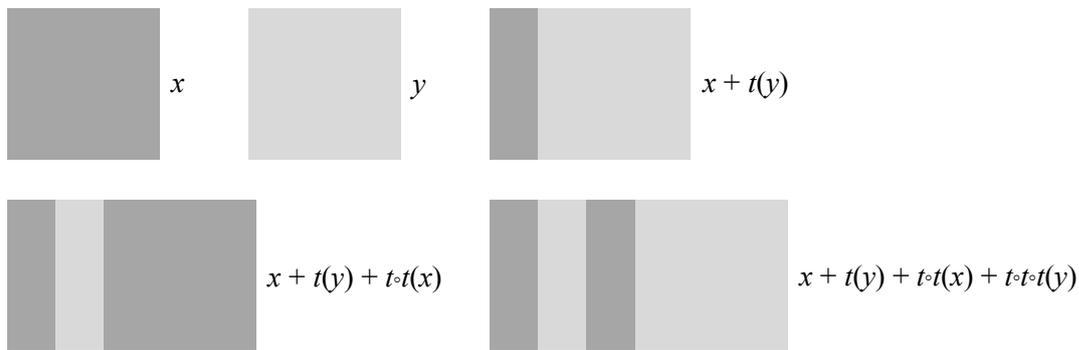
Figure 5: Repeatedly adding a translated square with alternating color under an "opaque" ranking. t is the translation, $\circ$ the symbol for a composition of transformations.

This visual interpretation yields an interesting difference on how we may deal with different kinds of attributes. In the case of line thicknesses, we have assumed that a line with 0 thickness is equivalent to no line. Both are equally invisible. However, if, instead, we consider labels as attributes, an empty set of labels may not necessarily require the shape to be considered empty as well. Visually, a point without labels is still visibly a point. That is, if we consider the labels as visual tags. Instead, if we consider labels as expressing layers in analogy to Photoshop layering, then, an empty set of labels would require the shape to be considered empty as well, as a shape may not exist outside of any layer. While this may seem inconsistent computationally, we must exactly embrace such incongruities in order to increase the applicability of shapes to different design contexts and scenarios. With respect to augmented shape algebras, this is

exactly the objective of this paper, exploring a uniform characterization of augmented shapes in defiance of variety or idiosyncrasy.

For the interested reader, the difference between the algebras U_{ij} and V_{ij} is the fact that under the former, the operations of product and difference define a partitioning. Given two shapes x and y of U_{ij}, the product $x \cdot y$ and the differences $x - y$ and $y - x$ form a partitioning of the sum $x + y$, that is $(x \cdot y) + (x - y) + (y - x) = x + y$, while $(x \cdot y) \cdot (x - y) = 0$ (or empty), $(x \cdot y) \cdot (y - x) = 0$ and $(x - y) \cdot (y - x) = 0$ (Table 1). We can simplify this using the symmetric difference operation, $x \oplus y = (x - y) + (y - x)$, such that $(x \cdot y) + (x \oplus y) = x + y$, while $(x \cdot y) \cdot (x \oplus y) = 0$. This not necessarily the case for V_{ij}. Consider the case of V_{02}, labeled points in two dimensions (Table 1). If the labeled shapes x and y contain the same point with a different label, we may consider the product of x and y to contain that point without any label(s). After all, the point is common, even if there isn't a common label. However, if we do so, the shapes $x \cdot y$, $x - y$ and $y - x$ no longer partition the shape $x + y$.

Table 1: The operations of sum, product and difference for points (U_{02}) and labeled points (V_{02}), and for line segments (U_{11}) and weighted line segments (W_{11}).

Algebra	x	y	$x + y$	$x \cdot y$	$x - y$	$y - x$
U_{02}						
V_{02}						
U_{11}						
W_{11}						

Note that this is not simply a result of the fact that we consider the product of labeled points to behave differently from the behavior of weighted line segments. Consider the algebra W_{ij} to denote the set of all finite arrangements of i-dimensional hyperplane segments of limited but non-zero measure in a j-dimensional space, with associated weights [3]. Given two shapes x and y of W_{ij}, the product $x \cdot y$ and the differences $x - y$ and $y - x$ do not always form a partitioning of the sum $x + y$. Given two shapes of weighted line segments in one dimension, with each shape consisting of a single line segment and both line segments overlapping, while both line segments have different weights (line thicknesses). The product of x and y contains the line

segment that is common to both shapes with the minimum of both weights. This same line segment also belongs to the difference of both line segments, when subtracting the line segment with the smaller weight from the one with the larger weight (Table 1).

4   A maximal element representation

Stiny [10] introduced the concept of maximal lines or line segments. Krishnamurti [40] defined the maximal representation of a shape as a set of disjoint spatial elements, each represented by its *co-descriptor* and boundary. The co-descriptor of a spatial element is its carrier—not to be mistaken with the carrier of an algebra— for example, the carrier of a point is the same point, the carrier of a line segment is the infinite line on which the line segment lies, and likewise for plane and volume segments. Not every carrier is necessarily an infinite spatial element, the carrier of a circular arc is the circle on which the arc lies. The co-descriptor of a spatial element is an important characteristic of the spatial element (or shape) under the shape operations of sum, product and difference: the co-descriptor is invariant under these operations, and two spatial elements interact under sum, product or difference only if they share the same co-descriptor, that is they are *co-equal*. More specifically, two points interact only if they are co-incident, two line segments if they are co-linear and two plane segments if they are co-planar [40].

Thus, a maximal shape is represented as a set of spatial elements such that any two elements in the set are either not co-equal or, otherwise, disjoint, that is, they neither overlap, nor share boundary. For example, in the case of a shape of line segments, any two line segments l and l' that are part of the shape combine into a single line segment if l and l' are co-equal and, moreover, l and l' overlap or share boundary. If no two line segments in the representation of the shape can combine in this way, then the shape is maximal. Note that points are co-equal only if they are identical and, otherwise, are disjoint. It follows that any set of points necessarily represents a maximal shape of points. We express the behavior of shapes of points as follows, where $x : X$ specifies the set X as a representation of the shape x:

$$(x : X), (y : Y) \in U_0 \implies$$
$$x + y : X \cup Y$$
$$x - y : X \setminus Y$$
$$x \cdot y : X \cap Y$$
$$x \leq y \Leftrightarrow X \subseteq Y \tag{1}$$

The behavior of shapes of other spatial elements can be expressed in terms of their co-descriptors and boundaries. In order to simplify the expression, we limit the behavioral expression to co-equal shapes, thereby expressing shapes solely by their boundaries. We refer to Stouffs [26] for an algebraic formulation that distinguishes all three abstraction levels, general shapes, co-equal shapes and spatial elements. Let $B[x]$ denote the boundary (or boundaries) of a (co-equal) shape x. The boundary of a shape of i-dimensional hyperplane segments is necessarily composed of $(i-1)$-dimensional hyperplane segments; the boundary of a shape of line segments is composed of points, the boundary of a shape of plane segments is composed of line segments, etc. Thus, the boundary of a shape in the algebra U_{ij} can be considered as a shape in the algebra $U_{(i-1)j}$. Given two (co-equal) shapes x and y, let I_x denote the boundary subshape of x that lies within y, O_x the boundary subshape of x that lies outside of y, M the boundary subshape of both x and y where the insides of x and y lie on the same side of the boundary, and N the boundary subshape of both x and y where the insides of x and y lie on opposite sides of the boundary (Figure 6 and Table 2) [41]. Note that the shapes I_x, I_y, O_x, O_y, M and N are necessarily disjoint. The behavioral expression then becomes:

$$(x : B[x]), (y : B[y]) \in U_{i \neq 0} \implies$$
$$x + y : B[x + y] = O_x + O_y + M$$
$$x - y : B[x - y] = O_x + I_y + N$$
$$x \cdot y : B[x \cdot y] = I_x + I_y + M$$
$$x \leq y \Leftrightarrow I_x = 0 \wedge O_y = 0 \wedge N = 0 \tag{2}$$
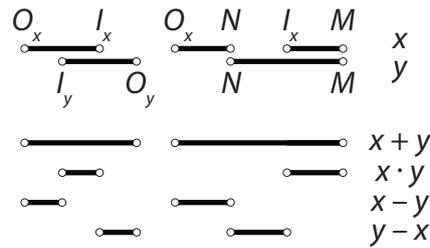
Figure 6: Specification of the boundary subshapes I_x, O_x, I_y, O_y, M and N, given two (co-equal) shapes of line segments x, and their contribution to $x + y$, $x \cdot y$, $x - y$ and $y - y$.

Table 2: Specification of the boundary subshapes I_x, O_x, I_y, O_y, M and N, given two (co-equal) shapes of plane segments (U_2) or volume segments (U_3), and the boundary subshapes' respective contribution to the sum, product and difference of both shapes.
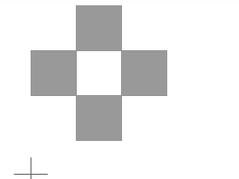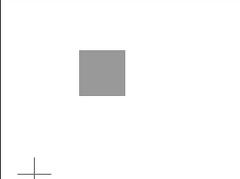
U_{22}		U_{33}	
$+$ x	$+$ y	$+$ x	$+$ y
$+$ O_x	$+$ O_y	$+$ O_x	$+$ O_y
$+$ I_x	$+$ I_y	$+$ I_x	$+$ I_y
$+$ M	$+$ N	$+$ M	$+$ N
$+$ $x + y$	$+$ $x \cdot y$	$+$ $x + y$	$+$ $x \cdot y$
$+$ $x - y$	$+$ $y - x$	$+$ $x - y$	$+$ $y - x$

4.1 Behaviors for augmented shapes

Next, we consider the maximal behavior for augmented shapes. Starting with the algebra V_{ij} of labeled shapes (Table 1), let us represent a (maximal) shape s with label set L by the pair (s, L), which we denote a *shape-attribute pair*. This representation assumes that every spatial element in s shares the same set of labels L, that is, every label in L is an attribute to every spatial element in s. A general labeled shape can then be represented as a set of shape-attribute pairs $\{(s_1, L_1), \ldots (s_n, L_n)\}$ with the shapes $s_1, \ldots, s_n$ not overlapping, though they may share boundary. For the labeled shape to be maximal, additionally, the label sets $L_1, \ldots, L_n$ must be distinct, otherwise, two pairs (s_i, L_i) and (s_j, L_j) with $L_i = L_j$ can easily be replaced with a single pair $(s_i + s_j, L_i)$. We can then express the behavior of labeled shapes in terms of the behavior of shapes. We observe that while any labeled shape may have an empty set of labels, no set of labels can be associated with the empty shape. Therefore, we consider a function 'e$_l$' to reduce any shape-attribute pair with zero shape (an 'empty' shape) to zero as follows:

$$e_l(s, L) = 0 \qquad\qquad \text{if } s = 0$$
$$ (s, L) \qquad\qquad \text{otherwise} \tag{3}$$

In addition, we consider a function 'm' to add a single shape-attribute pair (s, L) to a maximal, labeled shape sL, where s is known not to overlap with the labeled shape, though L may not be distinct from the label sets in the labeled shape. As any contribution to sL, and 'm', may be 0, e.g., following the application of the function 'e$_l$', we take the opportunity to remove any occurrence of 0 in sL.

$$m((s, L), sL) = sL \setminus \{0, (s', L)\} \cup \{(s + s', L)\} \qquad \text{if } \exists\,(s', L) \in sL$$
$$ sL \setminus \{0\} \cup \{(s, L)\} \qquad\qquad\quad \text{otherwise}$$
$$m(0, sL) = sL \setminus \{0\} \tag{4}$$

In order to simplify the behavioral expression for labeled shapes, we will assume that each of the input shapes can be expressed as a single shape-attribute pair:

$$(x : (s, L)), (y : (s', L')) \in V_{ij} \implies$$
$$x + y : m(e_l(s - s', L), m(e_l(s \cdot s', L \cup L'), \{e_l(s' - s, L')\}))$$
$$x - y : m(e_l(s - s', L), \{e_l(s \cdot s', L \setminus L')\})$$
$$x \cdot y : e_l(s \cdot s', L \cap L')$$
$$x \le y \Leftrightarrow s \le s' \wedge L \subseteq L' \tag{5}$$

Note that while we assume that each of the labeled input shapes can be expressed as a single shape-attribute pair, there is no such assumption about the result of the operations of sum, product and difference on two single-pair labeled shapes. In fact, the operation of sum may result in up to three shape-attribute pairs, that is, if neither L is a subset of L', nor L' is a subset of L, then the shapes $s - s'$, $s \cdot s'$ and $s' - s$ will each have a different associated label set and each of these three shapes may be non-empty. In the case of the operation of difference (of $(s, L) - (s', L')$), this reduces to up to two shape-attribute pairs as the shape $s' - s$ does not play part in the result. In the case of the operation of product, only the shape $s \cdot s'$ contributes to the result. Finally, a shape-attribute pair is part of another shape-attribute pair, in the context of labeled shapes, only if the subshape relationship holds for the respective shapes and the subset relationship for the respective label sets. We note that this behavioral expression can be extended to general labeled shapes, composed of multiple shape-attribute pairs, by comparing respective shape-attribute pairs and accumulating the results dependent on the operation under consideration. We leave this exercise to the reader.

When adopting labels in the case of layering shapes, e.g., in analogy to Photoshop layering, or filtering shapes in the case of say, medical tomography, we must consider a different function 'e_w' to reduce a shape-attribute pair with either zero shape or zero attribute to 0. We denote the attribute w in anticipation of weights as attributes but note that this function can equally be written using L instead of w.

$$e_w(s, w) = 0 \qquad \text{if } s = 0 \text{ or } w = 0$$
$$\qquad\qquad (s, w) \qquad \text{otherwise} \tag{6}$$

We consider layered shapes as members of the algebra $U_{ij} \wedge L$, with L an algebra for layer labels and '$\wedge$' denoting an operation of attribution between algebras [26].

$$(x : (s, L)), (y : (s', L')) \in (U_{ij} \wedge L) \Rightarrow$$
$$\qquad x + y : \mathrm{m}(e_w(s - s', L), \mathrm{m}(e_w(s \cdot s', L \cup L'), \{e_w(s' - s, L')\}))$$
$$\qquad x - y : \mathrm{m}(e_w(s - s', L), \{e_w(s \cdot s', L \setminus L')\})$$
$$\qquad x \cdot y : e_w(s \cdot s', L \cap L')$$
$$\qquad x \leq y \Leftrightarrow s \leq s' \wedge L \subseteq L' \tag{7}$$

Next, consider the algebra W_{ij} of weighted shapes (Table 1), similarly representing a shape s with weight w by the pair (s, w). Note that weights assigned to the same shape always interact, any two weights always combine into a single weight that is the maximum of both weights. As a

consequence, a weight attribute is always represented as a single (numeric) value. Also remember that, in contrast to labeled shapes, weighted shapes may neither have a zero weight, nor can any weight (other than zero) be associated with the empty shape. Therefore, we adopt the function 'e_w' to reduce a shape-attribute pair with either zero shape or zero weight to 0, but retain the function 'm' to adding a single shape-attribute pair (s, w) to a maximal weighted shape sw (replacing L with w).

$(x : (s, w)), (y : (s', w')) \in W_{ij} \implies$

$$x + y : m(e_w(s - s', w), m(e_w(s \cdot s', \max(w, w')), \{e_w(s' - s, w')\}))$$

$$x - y : m(e_w(s - s', w), \{e_w(s \cdot s', \max(w - w', 0))\})$$

$$x \cdot y : e_w(s \cdot s', \min(w, w'))$$

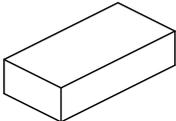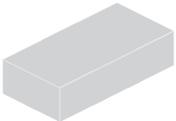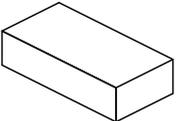$$x \leq y \Leftrightarrow s \leq s' \wedge w \leq w' \tag{8}$$

If, instead, we consider Stouffs' [1] definition for subtracting weighted shapes, then, we adopt a ternary conditional operator ('?:') to specify the interaction of two weights in the context of the operator of difference. Recall that, in this case, for two weights u and v, $u -_+ v = u$ if $u > v$, and 0 otherwise.

$(x : (s, w)), (y : (s', w')) \in W_{ij} \implies$

$$x + y : m(e_w(s - s', w), m(e_w(s \cdot s', \max(w, w')), \{e_w(s' - s, w')\}))$$

$$x - y : m(e_w(s - s', w), \{e_w(s \cdot s', (w > w' ? w : 0))\})$$

$$x \cdot y : e_w(s \cdot s', \min(w, w'))$$

$$x \leq y \Leftrightarrow s \leq s' \wedge w \leq w' \tag{9}$$

Having expressed the maximal behaviors for labels and weights, let us explore additional behaviors drawing from, e.g., Knight's [17] color grammars. Note that with respect to the maximal behavior here identified, that is, the operations of sum, product and difference, and the subshape relationship, Stiny's [11] descriptions behave exactly the way as labels when used as shape attributes. Therefore, descriptions will not be explored any further. Returning to Knight's [17] color attributes, let us first consider an "opaque" behavior for colors, where any colored shape that is being added "covers" any existing colored shape it overlaps, such that the new color replaces the existing color for this overlapping subshape (Figure 5). Knight [16]-[17] only specifies what happens in the case of adding a colored shape, she doesn't explicate any behavior under product or difference, though she does present examples of rules and derivations from which it can be concluded that an exact color match is required for subshape detection.

Similarly, for the operations of product and difference, a colored shape shares a part with another colored shape only if both shapes share the same color for this part (Table 3).

Table 3: The operations of sum, product and difference for plane segments (U_{22}) and colored plane segments ($U_{22} \wedge C$), and for volume segments (U_{33}) and colored volume segments ($U_{33} \wedge C$), with different colors under an "opaque" behavior.

	U_{22}	$U_{22} \wedge C$	U_{33}	$U_{33} \wedge C$
X				
Y				
$x + y$				
$x \cdot y$				
$x - y$				
$y - x$				

We consider a shape-attribute pair (s, c) for colored shapes but do not specify any specific representation for the color other than that is always a single color value. This 'single' value could be a number expressing a gray scale, a triple of numbers specifying a color in an RGB or

HSV color space, or a label specifying a color name. We consider colored shapes as members of the algebra $U_{ij} \wedge C$, with C an algebra for colors, and adopt the ternary conditional operator ('?:') to specify the interaction of two colors in the context of the operators of product and difference. The "opaque" behavior for colored shapes can be expressed as follows (Table 3):

$$
\begin{aligned}
(x : (s, c)), (y : (s', c')) &\in (U_{ij} \wedge C) \Rightarrow \\
x + y &: m(e_w(s - s', c), \{e_w(s', c')\}) \\
x - y &: m(e_w(s - s', c), \{e_w(s \cdot s', (c == c' ? 0 : c))\}) \\
x \cdot y &: e_w(s \cdot s', (c == c' ? c : 0)) \\
x \leq y &\Leftrightarrow s \leq s' \wedge c = c'
\end{aligned}
\tag{10}
$$

Knight [17] generalizes the opaque behavior described above through rankings. When two overlapping shapes, each with a different color, combine, the colors interact in accordance to their ranking. Either one color is ranked higher and covers the other color, or both colors rank equally and these may be replaced by any other color. Note that the ranking of two colors can be specified to be dependent on the ordering of the colors. This is the case for an "opaque" behavior where any color ranks higher than any other color, in that order if the first color is being added to the second color. If two overlapping shapes share the same color, this color can be replaced by another one, by virtue of the fact that both colors—being the same—rank equally. Knight [17] refers to the replacement of two colors by another one as the "blending" of the colors and offers an example with colors interpreted as materials, e.g., aluminum and wood. In line with this naming of colors—or materials—, here, we consider colors as an enumeration accompanied by a matrix that specifies for each pair of enumerated values their ranking, or rather the value resulting from a combining of both values. For example, Table 4 illustrates the ranking matrix for the opaque ranking considered above for two enumerated colors, "black" and "white" (see also, [2]); Table 5 illustrates the ranking matrix for one of the material ranking examples presented by Knight [17]. Note that Knight [17] only identifies the ranking in the case of equal materials, as other cases do not arise in her derivations. The matrix in Table 4 has been completed for unequal materials.

Table 4: Ranking matrix for the opaque ranking of "black" and "white".

$x \setminus y$	black	white
black	*black*	*white*
white	*black*	*white*

Table 5: Ranking matrix example for material ranking [17].

$x \setminus y$	aluminum	wood
aluminum	*wood*	*wood*
wood	*aluminum*	*aluminum*

Ranking materials is not much different from the way BIM software (e.g., ArchiCAD) resolves intersecting building elements. Element layers with different material properties are prioritized according to the material. For instance, assume a wall is composed of three layers, a load-bearing layer, an insulation layer and a finish layer. When two walls intersect, the load-bearing layer should prioritize over the insulation layer, which in turn should prioritize over the finish layer (Figure 7). The resulting ranking matrix is shown in Table 6.
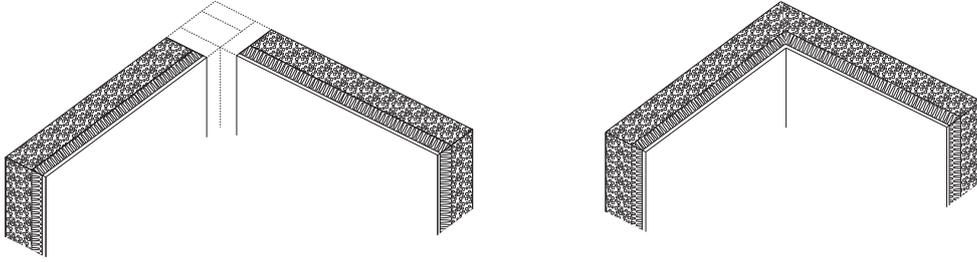
Figure 7: (left) Two walls to intersect, each wall is composed of three layers (load-bearing concrete, insulation, finish); (right) The result following the material ranking in Table 6.

Table 6: Ranking matrix example for wall material ranking.

$x \setminus y$	load-bearing	insulation	finish
load-bearing	*load-bearing*	*load-bearing*	*load-bearing*
insulation	*load-bearing*	*insulation*	*insulation*
finish	*load-bearing*	*insulation*	*finish*

Denoting the ranking matrix as 'xy[]', we can specify a generalized behavior for colors specified as enumerated values with ranking behavior as follows, retaining C to denote the algebra of colors:

$$(x : (s, c)), (y : (s', c')) \in (U_{ij} \, {}^\wedge C) \implies$$

$$x + y : \mathrm{m}(\mathrm{e_w}(s - s', w), \mathrm{m}(\mathrm{e_w}(s \cdot s', \mathrm{xy}[w, w']), \{\mathrm{e_w}(s' - s, w')\}))$$

$$x - y : \mathrm{m}(\mathrm{e_w}(s - s', c), \{\mathrm{e_w}(s \cdot s', (c == c' \, ? \, 0 : c))\})$$

$$x \cdot y : \mathrm{e_w}(s \cdot s', (c == c' \, ? \, c : 0))$$

$$x \leq y \Leftrightarrow s \leq s' \wedge c = c' \tag{11}$$

4.2 A uniform characterization

The similarities between the different behavioral specifications for augmented shapes, corresponding to different attribute types, are obvious, yet, so far, every new attribute type has required us to specify a behavioral expression for augmented shapes, or shape-attribute pairs, rather than only for the attributes in the context of augmented shapes. Here, we explore a uniform characterization of attribute behaviors within a fixed expression for augmented shapes. Let us first address the uniform behavioral expression for augmented shapes, independent of the behavior of the attribute type.

We consider general shape-attribute pairs (s, a) composed of a shape s and an attribute *form a*, with the attribute form a member of the attribute algebra A. We consider the function 'm' to add a single shape-attribute pair (s, a) to a maximal, augmented shape S, where s is known not to overlap with the augmented shape, though a may not be distinct from the attribute forms in the augmented shape. This is not any different from the function 'm' as defined before (4):

$$m((s, a), S) = S \setminus \{(s', a), 0\} \cup \{(s + s', a)\} \qquad \text{if } \exists \, (s', a) \in S$$
$$S \setminus \{0\} \cup \{(s, a)\} \qquad \text{otherwise}$$
$$m(0, S) = S \setminus \{0\} \qquad\qquad\qquad\qquad\qquad\qquad\qquad (12)$$

The same does not apply to the function 'e' that reduces any shape-attribute pair with zero shape and/or zero attribute form to zero, as there were two versions of 'e' defined. A first 'e_l' that applied to labels (and descriptions) and allowed a zero attribute form, though not a zero shape (3), and a second 'e_w' that applied to weights and colors and allowed neither a zero shape, nor a zero attribute form (6). In order to combine both versions into one function 'e', we assume an additional value 'nil', that is similar to zero, i.e., no or an empty value, but distinguishes itself from zero in that a 'nil' value is never allowed to persist as an attribute value, while a zero value may.

$$e(s, a) = 0 \qquad\qquad \text{if } s = 0 \text{ or } a = \text{nil}$$
$$(s, a) \qquad\qquad \text{otherwise} \qquad\qquad\qquad\qquad\qquad (13)$$

Then, a uniform behavioral expression for augmented shapes can be expressed as follows, considering operations of sum, product and difference, and a part relationship on attribute forms that will be differentiated according to attribute type:

$$(x : (s, a)), (y : (s', a')) \in (U_{ij} \wedge A) \implies$$

$$x + y : \mathrm{m}(\mathrm{e}(s - s', a), \mathrm{m}(\mathrm{e}(s \cdot s', a + a')), \{\mathrm{e}(s' - s, a')\}))$$

$$x - y : \mathrm{m}(\mathrm{e}(s - s', a), \{\mathrm{e}(s \cdot s', a - a')\})$$

$$x \cdot y : \mathrm{e}(s \cdot s', a \cdot a'))$$

$$x \leq y \Leftrightarrow s \leq s' \wedge a \leq a' \tag{14}$$

The attribute form is maximal if the individual attribute elements are disjoint, where disjoint is defined in terms of the behavior of the attribute type. For example, labels are disjoint of they are distinct. Similar to points, a set of labels necessarily represents a maximal form of labels. As a result, a behavioral expression for forms of labels is similar to (1), where x and y denote forms of labels, and X and Y the sets representing them, that is, $(x : X), (y : Y) \in \wp(L)$. The empty set equates to the zero form.

$$(x : X), (y : Y) \in \wp(L) \Rightarrow$$

$$x + y : X \cup Y$$

$$x - y : X \setminus Y$$

$$x \cdot y : X \cap Y$$

$$x \leq y \Leftrightarrow X \subseteq Y \tag{15}$$

On the other hand, weights are never disjoint, any two weights always combine into a single weight that is the maximum of both weights. As a consequence, a maximal form of weights is always represented as a singleton set. The behavioral expression for forms of weights can be written as follows. Note the use of 'nil' instead of zero resulting from the subtraction of a bigger weight from a smaller weight, in order to ensure that the resulting shape-attribute pair will be reduced to zero by the application of the function 'e'.

$$(x : \{w\}), (y : \{w'\}) \in \wp_1(\mathfrak{R}^+) \Rightarrow$$

$$x + y : \{\max(w, w')\}$$

$$x - y : (w > w' ? \{w - w'\} : \mathrm{nil})$$

$$x \cdot y : \{\min(w, w')\}$$

$$x \leq y \Leftrightarrow w \leq w' \tag{16}$$

As explained above, colors expressed via rankings take on a special behavior. With respect to the operation of sum, colors are never disjoint and any two colors always combine into a single color, as specified by the ranking matrix. Therefore, a maximal form of colors is also always represented as a singleton set. However, with respect to the operations of product and

difference, and the part relationship, colors are always disjoint unless they are equal. The behavioral expression for colors with ranking matrix can be written as follows:

$$(x : \{c\}), (y : \{c'\}) \in \wp_1(C) \Rightarrow$$

$$x + y : \{xy[c, c']\}$$
$$x - y : (c == c' \, ? \, \text{nil} : \{c\})\}$$
$$x \cdot y : (c == c' \, ? \, \{c\} : \text{nil}))$$
$$x \leq y \Leftrightarrow c = c' \tag{17}$$

5 Discussion

We have expressed a uniform characterization of augmented shapes in terms of shape-attribute pairs (s, a), where s is a co-equal shape and a an attribute form, and every spatial element in s shares the same attribute form a. Then, an augmented shape is generally represented by a set of shape-attribute pairs $\{(s_1, a_1), \dots (s_n, a_n)\}$ with the following conditions. Firstly, no two shapes from among $s_1, \dots, s_n$ may overlap, though they may share boundary. Obviously, there is no requirement for them to be co-equal and note that only co-equal shapes can possibly overlap. Secondly, the attribute forms $a_1, \dots, a_n$ must be disjoint with respect to the operation of sum on these attribute forms. Note that sets of labels are disjoint if these do not share any label, weights are never disjoint and any attribute form of weights specifies only a single weight, and colors under ranking are never disjoint under the operation of sum, though they may be disjoint with respect to the operations of product and difference, and the part relationship.

We have left to the reader the exercise of expressing a behavior of augmented shapes in terms of the behavior of shape-attribute pairs. Actually, from an implementation point of view, we may opt instead to represent augmented shapes in terms of spatial element-attribute pairs, rather than collecting spatial elements that share the same attribute form together. An augmented shape can then be said to be maximal if no spatial elements overlap, if spatial elements that share boundary have distinct attribute forms, and all attribute forms are maximal. Expressing the uniform characterization in terms of spatial element-attribute pairs would be more complicated as the product or difference of two spatial elements is not necessarily a single spatial element. From an implementation point of view, this may not matter though as the complexity of dealing with collections of spatial elements resulting from operations of product and difference on spatial elements will likely outweigh the complexity of considering an additional level of aggregation of subshapes of spatial elements that share the same attribute form.

Representing shapes as collections of spatial elements allows the imposition of an ordering relationship on the spatial elements that can assist in identifying elements that may interact. For example, any total ordering on points can simplify the process of determining coincident points in two shapes of points that serve as arguments to any of the operations of sum, product or difference, or the subshape relationship. The same holds for line segments that can be ordered by their co-descriptor and their position along the infinite line that is expressed by this co-descriptor. Since line segments only interact if they are co-equal, that is, share the same co-descriptor, and overlap or share boundary, they must be either adjacent in an ordered collection of line segments, or be separated by one or more line segments that would also overlap (see, [42]). A similar ordering cannot be found for plane segments or volume segments where, instead, a classification and construction approach along the line of (2) is more appropriate (see, [41]).

Finally, while we have provided a function 'm' to add a single shape-attribute pair (s, a) to a maximal, augmented shape S, where s is known not to overlap with any part of S, we have omitted a general function or operator that takes any collection of spatial elements, any collection of attribute elements, or any collection of shape-attribute pairs, and turns it into a maximal shape, attribute form, or augmented shape, respectively. Note that the operation of sum can be considered a simplified version of such general function or operator as it takes as arguments two elements that may not be maximal with respect to each other, though each would have to be maximal on its own, and returns a maximal shape or form combining both arguments. Therefore, a maximal operator can be conceived, at the minimum, by iteratively applying the operation of sum on partial shapes or forms. We leave the exercise to the reader.

5.1 An algebraic abstraction

Stouffs [26] demonstrates how an algebra with carrier $\wp(A \times \wp(B))$ and signature including sum, product, difference, and reduction can be defined in terms of a two-sorted partial algebra with carrier $\{A, \wp(A)\}$ and an attribute algebra with carrier $\wp(B)$. The partial algebra has a signature including operations of combine, common and complement; these correspond to the shape operations of sum, product and difference but, instead, defined on two spatial elements. The attribute algebra simply has a signature including sum, product, difference, and reduction. As such, it provides an algebraic characterization of spatial elements, on the one hand, and attribute forms, on the other hand, and derives an algebraic characterization of augmented shapes from these. We will not revisit this algebraic formulation here, instead, we will draw from this algebraic abstraction in order to explore the impact of our uniform characterization of augmented shapes.

Stouffs [26] notes that the algebraic derivation of augmented shape algebras distinguishes between shape algebras, expressing the object of the attribute relationship, and attribute algebras, expressing the non-spatial attribute in the relationship, as we have done here. However, it is also noted that shape and attribute algebras may share the same behavior and no conditions have been imposed on the behavior of either the shape algebra component or the attribute algebra component of the augmented shape algebra. As such, the algebraic derivation equally applies to augmented (shape) algebras where the base component is a two-sorted partial algebra for, e.g., labels, and the attribute algebra is a shape algebra. This, generally, allows for combinations of shapes and non-geometric attributes, non-geometric elements and shape attributes, shapes and shape attributes, and non-geometric elements and non-geometric attributes [26]. Stouffs et al. [43] provides an example from sensor planning for buildings, where the primary information target of a planner of embedded sensors is the building slab for the sensor to be embedded, e.g., represented as a label identifying the slab, while the shape of the slab is also required, but only as an attribute to the label.

Upon inspection of our own characterization, we can conclude that the same conclusion holds. Not only have we used a similar formulation for the behavioral expressions for shapes and attribute forms of different kinds, we have even identified the same expression for points (1) and labels (15). Except for the explication of U_{ij} in the uniform behavioral expression for augmented shapes, this expression applies equally to other combinations of base element types and attribute element types. The same applies to the function 'm'; only the function 'e' would need to be altered slightly, because when considering non-spatial element types for the first entity in the argument pair, the value may potentially be 'nil' and this case should be caught.

5.2 An implementation

In this paper we have omitted any algorithmic elaboration of the uniform characterization we have presented. We have already argued that from an implementation point of view, representing augmented shapes directly in terms of spatial element-attribute pairs is preferable over collecting spatial elements that share the same attribute form together in an in-between representation. This is further corroborated by the algebraic derivation of combinations of basic shape algebras with attribute algebras [26]. In fact, this algebraic abstraction at the same time serves as a procedural abstraction, offering insights into the modular implementation of a general shape grammar interpreter for different grammar forms. The SortalGI shape grammar interpreter exactly adopts this modular implementation by distinguishing different classes of (characteristic) individuals and (behavioral) forms. Each characteristic individual class specifies at a minimum the result of operations of combine, common and complement on a pair of instances of the class. Each behavioral form class specifies the result of operations of sum,

product and difference (and maximize) on a pair of forms of individuals, thereby relying on the implementation of the characteristic individual class when dealing with individuals from both forms. The representation of any data type (or sort) then relies on the combination of a characteristic individual class and behavioral form class. As the latter can be easily reused, e.g., a discrete behavior applies not only to points but also to labels, adding a new data type mostly only requires a new characteristic individual class implementing the operations of combine, common and complement (among others).While these have been developed for many geometrical data types, including circular arcs and quadratic Bezier curves, fewer attribute types are pre-defined and it is exactly with respect to attribute data types that idiosyncratic types can be more common. While some coding is still necessary, we hope that the uniform characterization of the behavior of different attribute types under operations of sum product and difference, and a part relationship, can serve designers and researchers to formalize their potentially idiosyncratic attribute data types and extend the SortalGI shape grammar interpreter to support their grammatical approach to design.

6 Conclusion

We have presented a uniform characterization of the behavior of different attribute types under operations of sum product and difference, and a part relationship. Together with a uniform characterization of the behavior of spatial element types under the same operations and relationship, and a unique behavioral expression for augmented shapes in terms of these characterizations, we have been able to uniformly describe the behavior of augmented shapes. In fact, the characterizations of attribute types and spatial element types are almost uniform, providing for the ability to combine spatial and non-spatial element types in other combinations of augmented forms.

7 Acknowledgements

8 References

[1]   Stiny, G. Computing with form and meaning in architecture, J Archit Educ 1985; 39(1):7–19. https://doi.org/10.1080/10464883.1985.10758382

[2]   Stouffs R. On shape grammars, color grammars and sortal grammars. In: Achten H, Pavlicek J, Hulin J, Matejovska D, editors. Digital Physicality, volume 1. Brussels: eCAADe; 2012, p. 479–87.

[3]   Stiny G. Weights. Environ Plann B Plann Des 1992;19:413–30. https://doi.org/10.1068/b190413

[4]   Agkathidis A. Generative Design Methods - Implementing Computational Techniques in Undergraduate Architectural Education. In: Martens B, Wurzer G, Grasl T, Lorenz WE, Schaffranek R, editors. Real Time, volume 2. Brussels: eCAADe; 2015, p. 47–55.

[5]   Terzidis K. Algorithmic Architecture. Architectural; 2006.

[6]   Janssen P, Stouffs R. Types of Parametric Modelling. In: Ikeda Y, Herr CM, Holzer D, Kajima S, Kim MJ, Schnabel MA, editors. Emerging Experienced in the Past, Present and Future of Digital Architecture. Hong Kong: CAADRIA; 2015, p. 157–66.

[7]   Bohnacker H, Grossa B, Laub J. Generative Design: Visualize, Program, and Create with Processing. Princeton Architectural; 2012.

[8]   Flemming U. The role of shape grammars in the analysis and creation of designs. In: Kalay YE, editor. Computability of Design. Wiley-Interscience; 1987, p. 245–72.

[9]   Woodbury R, Burrow A. Whither design space? AI-EDAM 2006;20:63–82. https://doi.org/10.1017/S0890060406060057

[10] Stiny G. Introduction to shape and shape grammars. Environ Plann B Plann Des 1980;7:343–51. https://doi.org/10.1068/b070343

[11] Stiny G. A note on the description of designs. Environ Plann B Plann Des 1981;8:257–67. https://doi.org/10.1068/b080257

[12] Carlson C, McKelvey R, Woodbury RF. An introduction to structure and structure grammars. Environ Plann B Plann Des 1991;18(4):417–26. https://doi.org/10.1068/b180417

[13] Heisserman J, Woodbury R. Geometric design with boundary solid grammars, In: Formal Design Methods for CAD. Amsterdam: North-Holland; 1994, p. 85–105.

[14] Duarte JP, Correia R. Implementing a description grammar: generating housing programs online. Constr Innov 2006;6(4):203–16. https://doi.org/10.1108/14714170610713890

[15] Stiny G, Gips J. Shape grammars and the generative specification of painting and sculpture. In: Information Processing71. Amsterdam: North-Holland; 1972, p. 1460–1465.

[16] Knight TW. Color grammars: designing with lines and colors. Environ Plann B Plann Des 1989;16:417–49. https://doi.org/10.1068/b160417

[17] Knight TW. Color grammars: the representation of form and color in design. Leonardo 1993;26:117–24.

[18] Stouffs R, Krishnamurti R. Sortal grammars as a framework for exploring grammar formalisms, In: Burry M, Datta S, Dawson A, Rollo J, editors. Mathematics and Design 2001.

Geelong, Australia: The School of Architecture & Building, Deakin University; 2001, p. 261–269.

[19] Grasl T, Economou A. From topologies to shapes: parametric shape grammars implemented by graphs. Environ Plann B Plann Des 2013;40(5):905–22. https://doi.org/10.1068/b38156

[20] Wortmann T. Representing shapes as graphs. Master's Thesis. Cambridge, MA: MIT; 2013.

[21] Strobbe T, Pauwels P, Verstraeten R, De Meyer R, Van Campenhout J. Toward a visual approach in the exploration of shape grammars. AI-EDAM 2015;29(4):503–21. https://doi.org/10.1017/S0890060415000475

[22] Wortmann T, Stouffs R. Algorithmic complexity of shape grammar implementation. AI-EDAM 2018;32(2):138–46. https://doi.org/10.1017/S0890060417000440

[23] Stouffs R, Krishnamurti R. Algorithms for classifying and constructing the boundary of a shape. J Des Res 2006;5(1):54–95. http://dx.doi.org/10.1504/JDR.2006.010796

[24] Jowers Y, Earl C. The construction of curved shapes. Environ Plann B Plann Des 2010;37:42–58. https://doi.org/10.1068/b35093

[25] Jowers Y, Earl C. Implementation of curved shape grammars. Environ Plann B Plann Des 2011;38:616–35. https://doi.org/10.1068/b36162

[26] Stouffs R. Implementation issues of parallel shape grammars. AI-EDAM 2018;32:162–76. https://doi.org/10.1017/S0890060417000270

[27] Krstic D. Constructing algebras of design. Environ Plann B Plann Des 1999;26:45–57. https://doi.org/10.1068/b260045

[28] Krstic D. Algebras of shapes revisited. In: Gero JS, editors. Design Computing and Cognition '12, Dordrecht: Springer; 2012, p. 361–376.

[29] Beirao JN. CItyMaker: Designing Grammars for Urban Design. PhD Thesis. Delft, the Netherlands: TU Delft; 2012.

[30] Eastman CM. On the analysis of intuitive design processes. In: Moore GT, editor. Emerging Methods in Environmental Design and Planning, Cambridge, MA: MIT; 1970, p. 21–37.

[31] Latour B. Drawing things together. In: Lynch M, Woolgar S, editors. Representation in Scientific Practice, Cambridge, MA: MIT; 1990, p. 19–68.

[32] Eastman C. New directions in design cognition: studies of representation and recall. In: Eastman C, McCracken M, Newstetter W, editors. Design Knowing and Learning: Cognition in Design Education, Amsterdam: Elsevier; 2001, p. 147–98.

[33] Akin Ö. "Simon says": design is representation. Arredamento 2001;July. http://www.andrew.cmu.edu/user/oa04/Papers/AradSimon.pdf.

[34] Stouffs R. Constructing design representations using a sortal approach. Adv Eng Inform 2008;22:71–89. https://doi.org/10.1016/j.aei.2007.08.007

[35] Kiviniemi A. Ten years of IFC development - why are we not yet there? Keynote lecture at the Joint International Conference on Computing and Decision Making in Civil and Building Engineering, Montreal, 14-16 June 2006.

[36] AIA Model Support Group. IFC2x Edition 3. International Alliance for Interoperability; 2006. http://www.iai-international.org/Model/R2x3_final/index.htm.

[37] Manola F, Miller E, editors. RDF Primer. W3C World Wide Web Consortium; 2004. http://www.w3.org/TR/rdf-primer/.

[38] Arnold BH. Logic and Boolean Algebra. Englewood Cliffs, NJ: Prentice-Hall; 1962.

[39] Stiny G. Shape: Talking about Seeing and Doing. Cambridge, MA: MIT; 2006.

[40] Krishnamurti R. The maximal representation of a shape. Environ Plann B Plann Des 1992;19:267–88. https://doi.org/10.1068/b190267

[41] Krishnamurti R, Stouffs R. The boundary of a shape and its classification. J Des Res 2004;4(1):75–101. http://dx.doi.org/10.1504/JDR.2004.009843

[42] Krishnamurti R. The arithmetic of shapes. Environ Plann B Plann Des 1980;7:463–84. https://doi.org/10.1068/b070463

[43] Stouffs R, Krishnamurti R, Park, K. Sortal structures: supporting representational flexibility for building domain processes. Comput Aided Civ Infrastruct Eng 2007;22:98–116. https://doi.org/10.1111/j.1467-8667.2006.00473.x