
Nikos A. Salingaros
University of Texas at San Antonio
Supplement

• More on universal scaling and the scaling factor: the assembly problem
• New material that was not originally presented as part of the lecture series
1.4. Scaling from division

- A larger form can be virtually divided into smaller parts
- Division generates the smaller scales
- Grouping of smaller elements occurs within larger scale
- Similarity establishes scaling coherence between distinct scales
Analogy with embryonic development

• The embryo starts out as a single cell
• It subdivides into an increasing number of cells, clustering into groups
• All the subsequent cell divisions work together to form the growing embryo
• Insights of Christopher Alexander
Division in one dimension

- To illustrate scale formation through division, consider only lengths
- One-dimensional architectural model makes computations easier
- Divide a length into 2, 3, or more parts of comparable size
How many divisions?

- Smaller scales are created by subdividing the larger scale
- Simplest division is into two parts
- Too many identical parts, however, produce combinatorial complexity
Divide into two identical parts
Divide length into 3 parts
Recombination

- The parts created by division must be appropriate for reconstructing the original larger scale
- Division is a process that reinforces, and does not destroy, the whole
- Grouping and recombination relates daughter and parent scales
Linking scales

• Scaling hierarchy grows out of a relationship among three scales:

• A particular scale
• — generates an immediately smaller scale through division
• — and is related to its immediately larger scale through grouping
The Golden Mean doesn’t apply

• It is impossible to divide a form into fewer than 2 comparable parts!
• Therefore, we cannot use a scaling factor of $\Phi = 1.618$ to divide a form
• This elementary error was made by Le Corbusier in proposing his Modulor scheme for design
Relationship of 1.62 : 1
Adding unrelated structure

• Suppose we try using 1.62 as a scaling factor generating smaller parts

• A group of objects on this new smaller scale cannot fit into the original scale

• The resulting smaller scale is NOT a division, but an entirely unrelated scale
Conclusion: the scaling factor

- Scaling from division defines the lowest value for the scaling factor.
- The scaling factor must be larger than or equal to 2.
- But not so large that we face the problem of combinatorial complexity.
1.5. Combinatorial complexity

• Suppose we have a large number of identical smaller parts
• Triggers comparison, a combinatorial process that generates fatigue
• Monotonous repetition is thus not only boring, it can actually be stressful
Unexpected complexity

• NOT Kolmogorov complexity, which considers monotonous repetition as simple instead of complex

• — measures complexity as the length of the algorithm required to produce it

• We are instead interested in a very different combinatorial complexity
Neural system

• Evolved to cope with the natural world
• Expends energy to arrange data from senses into coherent patterns
• Tries to group similar pieces into larger wholes (Gestalt)
• Keeps working to find some grouping
Conjecture on perception

• The brain works combinatorially
• Tries out all possible geometric combinations, deciding which is more effective for understanding
• In the absence of explicit groupings, this process leads to stress and fatigue
Cognitive stress

• We don’t really know the cognitive mechanism that evaluates a configuration
• A monotonous sequence with too many similar pieces is cognitively exhausting
• Our perception works continuously to evaluate all NONEXISTENT groupings in possible combinations
Support from “Enterprise Architecture”

• The structure and processes of a business, and how information systems and technology help those processes
• An economic pillar of 20C society
• “Architecture” is used here in the sense of designing software and business systems — analogy to buildings
Sessions’ Law of Software Complexity

• “The complexity of a software system is a function of the number of states in which that system can find itself”

• Combinatorial complexity increases with the number of identical parts

• Solution is to iteratively partition sets of parts into coherent groups
Possible permutations

• In what permutation can we assemble $n$ parts to create a higher scale?
• The number of possible permutations of $n$ distinct parts is equal to $n!$
• $2! = 1 \times 2 = 2$, $3! = 1 \times 2 \times 3 = 6$, $4! = 24$
• A well-known result from combinatorics
Six ways of assembling three similar parts
Total number of states

- A “binary” cellular automaton has cells that can be either black or white.
- A row of $n$ cells can assume $2^n$ states.
- For $n$ parts, the number of comparisons equals the possible states $= 2^n$.
- The choices grow exponentially as $n$ increases.
Cellular automaton with 3 cells
Comparisons of combinations

• Suppose that the mind compares pairs to find similarities, symmetries, groupings, and patterns
• For $n$ parts, the number of pair-wise comparisons equals $n(n – 1)/2$
• The choices grow as $n^2/2$ as $n$ increases
Pair-wise matching of 4 cells
Comparisons of different counts

- Does cognition evaluate all possible permutations of the repeating parts?
- Or does it count repetitions as states of a one-dimensional cellular automaton?
- Or does it work according to pair-wise comparisons among identical parts?
Complexity we cannot handle

- Suppose we have 10 repeating parts:
  - there are $10! = 3,628,800$ possible permutations
  - there are $2^{10} = 1,024$ states of a binary Cellular Automaton
  - there are 45 pair-wise comparisons
The magic number 7

• The human mind can simultaneously handle around 7 pieces of information, and preferably nearer 5
• We can grasp 5 to 7 combinations
• Applying this concept, we should prefer a set of repeating units that leads to no more than about 7 comparisons
Complexity that we CAN handle

- Compare three different mechanisms:
  - Permutations: 3 parts = 6 states, 4 parts = 24 states (too many)
  - Binary cellular automata: 3 parts = 8 states, 4 parts = 16 states (too many)
  - Pair-wise comparisons: 4 parts = 6, 5 parts = 10, 6 parts = 15 (too many)
Sessions’ Third Law of Partitions

• “The number of autonomous units that make up a given partition should be in the range of 3 through 8”

• Support from Enterprise Architecture for the number of elements in a group

• This is precisely the optimal size of a group so as to avoid cognitive fatigue
Monotonous repetition is tiring

- According to our conjecture, repetition of identical parts is cognitively tiring
- Need to break the monotony:
  - (A) either make each similar part slightly different using variety
  - (B) or group the parts into clusters
First solution: symmetry with variety

- Different capitals or surface design on Medieval columns
- Variations in a row of repeating windows, but still in strict alignment
- Repeating units are distinguished by variety on a lower scale
Columns with variety
Windows gain variety
Second solution: grouping parts

- Create intermediate clusters into which several parts assemble into groups
- Grouping generates intermediate scales
- The process of grouping according to scales recursively generates the universal scaling hierarchy
Grouping into clusters of three
Grouping into clusters of four
Creation of scales

• Solving the combinatorial complexity problem generates the scaling hierarchy
• VARIETY acts on a smaller scale, thus differentiation creates several smaller scales
• GROUPING PARTS creates a larger scale
• But monotonous repetition prevents the formation of the scaling hierarchy
Conclusion: repetition

• Scaling hierarchy aids cognition
• Monotonous repetition is not only boring, it is stressful for our perception
• But it forms part of 20C design
• We strongly condemn the stressful effects of the modernist design canon
• Variation and groupings are necessary