Citation for published version:
Gopsill, J 2018, 'Modelling the design spaces of construction kits' HPC Symposium 2018, Bath, UK United Kingdom, 6/06/18 - 6/06/18, .

Publication date:
2018

Modelling the design spaces of construction kits

Dr. J. A. Gopsill
https://jamesgopsill.github.io/

Principles of Construction Kits

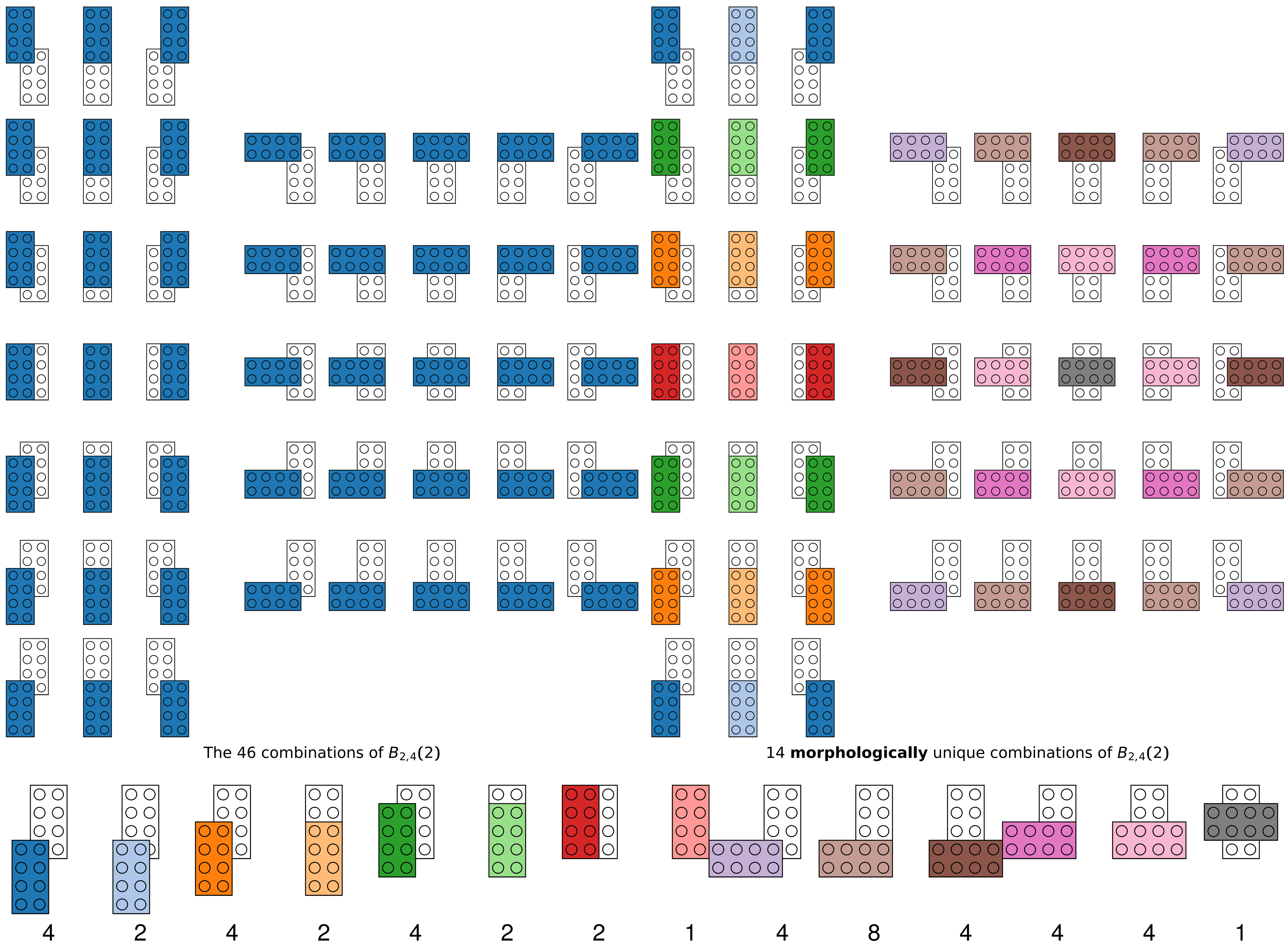

Fig. 1: Digital & physical construction kits

Introduction

Construction Kits (CKs) are a fundamental tool for the generation and communication of ideas. First appearing as a means to support children's development through the building of models, they have since been used to support the learning of languages and physical environments. The application of CKs to support New Product Development (NPD) is also well established where they are used extensively to support the creation and sharing of ideas. They have been particularly successful in supporting configuration design tasks, such as city, town, office and manufacturing facilities planning, where stakeholders from multiple disciplines come together to design.

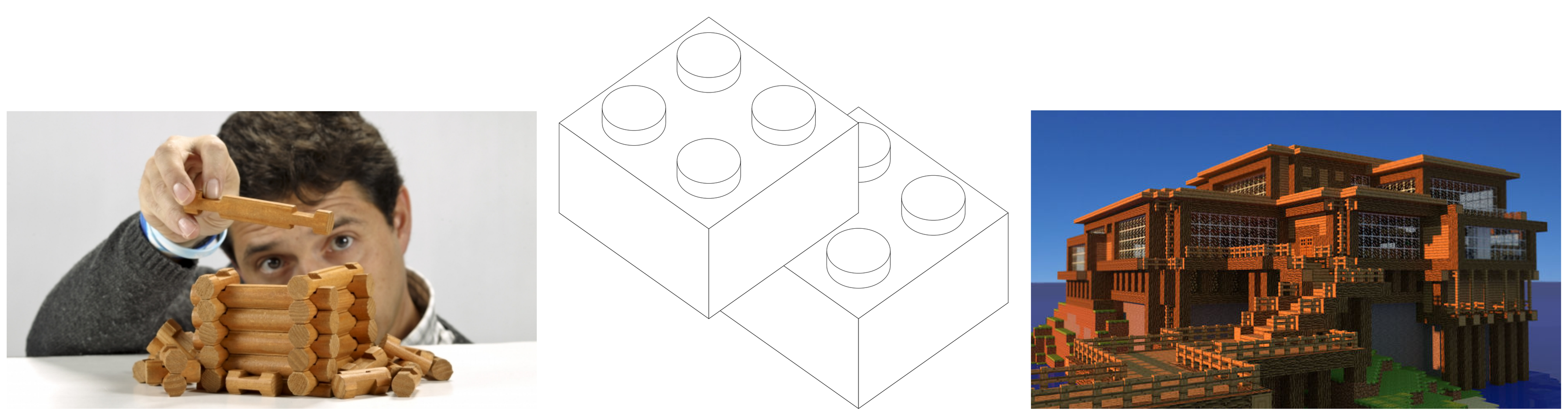

Fig. 2: Digital & physical construction kits

Whilst their ubiquity across society is apparent, it is surprising to see that there is little research on how a CK represents and constrains the design space. The manner in which a CK constrains the design space may have a profound effect on the solutions generated by individuals as well as supporting the development of their problem solving skills. This poster presents the development of code that utilises High-Performance Computing (HPC) to compute the design spaces of CKs so that a better understanding of this phenomena can be developed.

Using High-Performance Computing to Solve the Combinatorial Problem

To solve the combinatorial problem of construction kits, a novel breadth first search algorithm has been developed and deployed on the University of Bath's Balena High-Performance Computer. The algorithm simulates an individual constructing with the kit and provides the ability to place additional rules and constraints that an individual might need to adhere to. For example, limiting the orientation that a brick could be placed in as well as the type of brick that is being placed.

The code identifies the number of morphologically unique combinations for a set of n bricks and determines the number of pathways to each of these combinations. The first question that has been asked is whether there exists any bias within a CK and if so, to what extent?

The code has been developed in Python and interaction between nodes is handled by the mpi4py library. The implementation equally distributes the computing of the combinations across the number of node requested.

Findings

Initial findings reveal that significant bias does exist within CKs and the addition of rules significantly reduces the design space but has less of an effect on the bias. This begs the question:

Are our designs of our own making or are we simply actuating the intended model the CK biases?

$B_{w,h}(n)$	P	M	$\frac{P}{M}$	$P_{\min}$	$P_{\max}$	$A_{non-biased}$	A_{biased}	B	G
NAC									
$B_{2,4}(2)$	92	14	6.6	2	16	620	137	472	0.31
$B_{2,4}(3)$	12.3×10^3	429	28.7	4	104	2.6×10^6	98.0×10^3	1.6×10^6	0.39
$B_{2,4}(4)$	2.2×10^6	33.3×10^3	66.6	4	832	36.9×10^9	557.2×10^6	21.4×10^9	0.42
$B_{2,4}(5)$	497.8×10^6	3.0×10^6	166.2	4	14.9×10^3	744.8×10^{12}	4.5×10^{12}	370.8×10^{12}	0.51
NRC									
$B_{2,4}(2)$	42	8	5.3	2	8	162.0	49.0	131.0	0.27
$B_{2,4}(3)$	2.5×10^3	139	17.9	4	32	172.5×10^3	10.8×10^3	126.0×10^3	0.29
$B_{2,4}(4)$	198.5×10^3	4.6×10^3	43.5	4	448	449.7×10^6	10.5×10^6	300.6×10^6	0.34
$B_{2,4}(5)$	19.7×10^6	193.4×10^3	101.6	4	2.0×10^3	1.9×10^{12}	18.7×10^9	1.1×10^{12}	0.42

Fig. 3: Number of pathways (P) and morphologically equivalent combinations (M) of $B_{(2,4)}(n)$

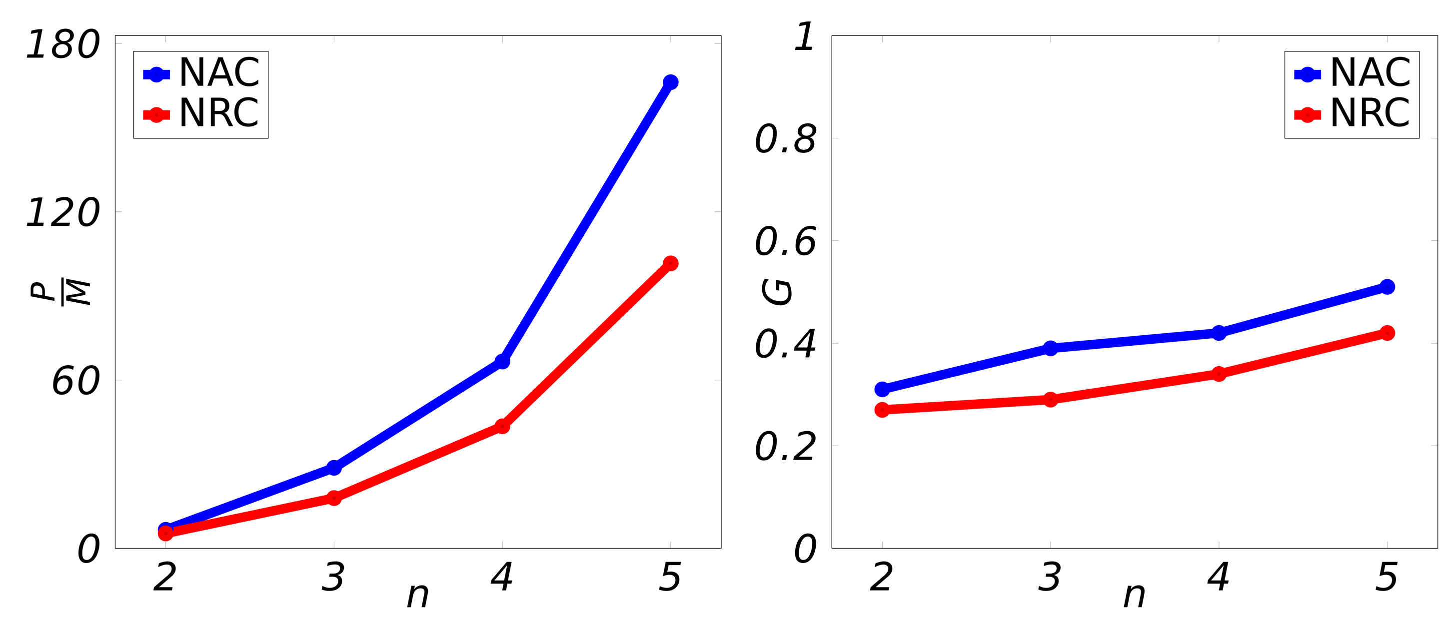

Fig. 4: Comparing a No Addition Constraint (NAC) and No Rotation Constraint (NRC) case

HPC Challenges & Future Work

Due to the scaling of the number of combinations, the current issue is one of storage and checking morphologically equivalence across all the combinations of n bricks. Thus, work is being performed to reduce the memory intensity of the code by using index representations of all the potential positions of the bricks rather than having each combination holding an object that contains all the details of each brick in their combination.

In addition, the nature of the operation lends itself to further parallelisation of the code and to explore the use of GPU architectures as a means to compute the combinations.

LaTeX TikZposter