Advanced Engineering Informatics 42 (2019) 100928

Contents lists available at ScienceDirect

Advanced Engineering Informatics

journal homepage: www.elsevier.com/locate/aei

A lattice-based approach for navigating design configuration spaces

Alison McKay[a,*], Hau Hing Chau[a], Christopher F. Earl[b], Amar Kumar Behera[c,1], Alan de Pennington[a], David C. Hogg[a]

[a] University of Leeds, Leeds LS2 9JT, UK
[b] Open University, Walton Hall, Milton Keynes MK7 6AA, UK
[c] Queen's University Belfast, University Road, Belfast BT7 1NN, UK

ARTICLE INFO

Keywords:
Design description
Design representation
Design structure
Design language
Bill of materials
Hypercube lattice

ABSTRACT

Design configurations, such as Bills of Materials (BoMs), are indispensable parts of any product development process and integral to the design descriptions stored in proprietary Computer Aided Design and Product Lifecycle Management systems. Engineers use BoMs and other design configurations as lenses to repurpose design descriptions for specific purposes. For this reason, multiple BoMs typically occur in any given product development process. For example, an engineering BoM may be used to define a configuration that best supports a design activity whereas a manufacturing BoM may be used to define the configuration of parts that best supports a manufacturing process. Current practice for the definition of BoMs involves the use of indented parts lists and dendograms that are prone to error because it is easy to create discrepancies across BoMs that, in essence, are defined through collections of part identifiers such as names and part numbers. Such errors have a significant detrimental effect on the performance of product development processes by creating the need for rework, adding costs and increasing time to market.

This paper introduces a design description capability that ensures consistency across BoMs for a given design. A boolean hypercube lattice is used to define a design configuration space that includes all possible configurations for a given design description. Valid operations within the space are governed by the mathematics of hypercube lattices. The design description capability is demonstrated through an early engineering design configuration software tool that offers significant benefits by ensuring consistency across the BoMs for a given design. The software uses and generates design descriptions that are exported from and imported to commercially available design systems through a standard (ISO 10303-214) interface format. In this way, potential for early impact on industry practice is high.

1. Introduction

The success of today's global supply networks depends on the efficient and effective communication of design descriptions (including design intent and shape definitions) that suit the requirements and capabilities of the wide range of engineering functions, processes and suppliers involved in the delivery of products to markets. Technical product data packages are used to provide these design descriptions. At a recent industry summit [1], a representative of Boeing noted that some 40% of the technical data needed to create a product resides outside the shape definitions in the technical product data package. This non-shape data includes design requirements, functional descriptions of the product, design configurations, manufacturing information and process-related information such as change histories and design approvals. The focus of this paper is on the design configurations, defined using Bills of Materials (BoMs), that are integral parts of both shape definitions and the 40% of non-shape related product data. A BoM is a hierarchical structure that defines a configuration of parts for a given product. They are fundamental to engineering design and development processes because they act as integrators: adapting detailed design descriptions to suit the needs of particular engineering processes. A given design can have multiple BoMs, each of which provides a different configuration of parts to suit a different purpose, such as, assembly, manufacturing, and service. The ability to reconfigure BoMs while maintaining internal consistency of the technical data package (where all BoM configurations are complete and compatible with each

* Corresponding author.
E-mail addresses: a.mckay@leeds.ac.uk (A. McKay), h.h.chau@leeds.ac.uk (H.H. Chau), c.f.earl@open.ac.uk (C.F. Earl), a.behera@qub.ac.uk (A.K. Behera), a.depennington@leeds.ac.uk (A. de Pennington), d.c.hogg@leeds.ac.uk (D.C. Hogg).
[1] Work completed at University of Leeds.

https://doi.org/10.1016/j.aei.2019.100928
Received 1 October 2018; Received in revised form 17 April 2019; Accepted 19 May 2019
Available online 07 June 2019

Nomenclature

BoM	bill of materials
CAD	computer aided design
CSG	constructive solid geometry
EBoM	engineering bill of materials
ERP	enterprise resource planning
IT	information technology
MRP	material requirements planning
PLM	product lifecycle management
SBoM	shipping bill of materials
SME	small to medium sized enterprise
TDP	technical data package

other and underlying design descriptions) is a major challenge [2,3]. If solved, the ability to reconfigure BoMs will lead to new ways of managing and designing product development systems.

It is widely acknowledged that improving the management of engineering knowledge and information is critical to delivering better quality products to markets at less cost and more quickly. Literature describing approaches to the management of design descriptions is reviewed in Section 2 and the potential applicability of embedding (currently used in shape computation applications) to describe alternative BoMs is introduced. Embedding is a general mathematical construct that allows one mathematical structure to be a consistent substructure of another [‘holding’ structure]; the substructures have the same mathematical properties as their holding structure. In this way, embedding has the potential to ensure the internal consistency of multiple structures, such as multiple BoMs in a technical data package. However, general purpose methods for the implementation of embedding are not yet available. In Section 3 we introduce lattice theory as such a mechanism and in Section 4 we outline the prototype-based approach that we used to generate the results presented in Section 6. The paper reports a feasibility study which used a simplified robot case study (in Section 5) to illustrate an exploration of whether embedding can be used to allow multiple BoMs to be superimposed on each other. If successful this has the potential to reduce data duplication in design descriptions, provide improvement opportunities for the management of change and allow new BoMs to be defined as and when needed through the entire product lifecycle. In this way, the research is paving the way for a new generation of design tools that support the configuration of BoMs. These success criteria are used in Section 7 to evaluate the results presented in Section 6, followed by a summary of key findings and areas for future work in Section 8.

2. Current approaches to the management of design descriptions

This paper proposes a novel approach to ensuring the referential integrity of BoMs in technical data packages. Technical data packages are core aspects of what is widely referred to as “model based engineering” [4,5]. Frechette et al. [6] describe a technical data package in the following way.

> *“A technical data package (TDP) contains a technical description of an item adequate for supporting an acquisition strategy, production, engineering, and logistics support. The description defines the required design configuration or performance requirements, and procedures required to ensure adequacy of item performance. It consists of applicable technical data such as models, drawings, associated lists, specifications, standards, and performance requirements. Lastly, … a Technical Data Package (TDP) should provide not only sufficient data to procure “up front” but re-procure later in the product lifecycle.”*

Current approaches to the integration of engineering information seek a common underlying meta-model to support, ultimately, model-based solutions. Many meta-models and design ontologies, and associated data models, for specific application domains are available in the literature. McMahon [7] acknowledges the role of ontologies in design informatics and example research cases do exist. However, the industrial uptake of research ontologies is limited, primarily because the cost of change would be prohibitive and the ontologies are validated only in limited application areas. In NIST’s 2017 “Model-Based Enterprise Summit” [1], and in line with other presentations, Kassel [8] challenges the notion of a single source (the “model”, which would be underpinned by an ontology). Kassel describes a digital thread that includes design structures and a need for effective configuration management, amongst other things, to string together heterogeneous collections of design descriptions, each of which might be underpinned by its own domain specific ontology. The use of a digital thread removes the need for the integrated meta-model, and its underlying ontology, that are prerequisites for the creation of a single design model.

The details of specific meta-models and associated design ontologies created to deliver such improvements are beyond the scope of this paper. A common weakness in such solutions lies in their high emphasis on technological aspects of the problem and limited or no emphasis on organisational dimensions that are critical to delivering improvements in product development process performance [9]. However, there is a more general literature on approaches to the integration of engineering information with a view to implementing model-based solutions. The goal for the research presented in this paper was to enable the definition of design structures that are consistent with the design technical data package, as and when needed through the life of the product. Although company design processes can define the types of design structures and descriptions that will be required for passage through the process in a stage gated manner, there are also circumstances where design structures need to be created without prior knowledge of what the structure would be because it is not always possible to predict reliably future needs. Liu et al. and Yin & Ma [10,11] propose feature-based approaches but these depend on a knowledge of the downstream design structures and are, therefore, only applicable to specific cases. The annotation of lightweight shape models overcomes this problem [12] but the annotations are text-based so limited when new structures are required and models become cluttered as the volume of annotations increases through the life of the product [13].

More recently, there has been a renewed focus on design configurations. A number of authors, such as Kashkoush and El Maraghy [14], identify BoMs as being critical elements of technical communication and highlight limitations in current IT support for the configuration of BoMs. Zhou et al. [3] propose a method for transforming an as-built BoM into a service BoM which includes general purpose operations that could be applicable to the transformation of other types of BoM. However, as a transformation process, the information content of the resulting BoM is inevitably limited by that of the source BoM. BoMs typically exist in multiple systems and formats, such as PLM systems for design data, ERP systems for manufacturing and asset management systems when products are in use. Although each system has bespoke functionalities, the maintenance of consistency across multiple systems and data formats is critical in reducing rework and improving product development process performance. The research reported in this paper addresses this issue by ensuring the referential integrity of technical data packages. This is achieved by exploiting the capabilities of boolean hypercube lattices to provide the grammatical rules for design configuration (these are common across all data packages) and coupling them with a vocabulary for design configuration that is generated automatically from a source BoM and specific to a given data package. In this way we have created a tool for design configuration where new configurations can be defined within a conceptual space, the design

configuration space, which includes only valid configurations of the design. Because the design configuration space includes all possible valid configurations, specific configurations can be defined as needed, throughout the product lifecycle. This paper contributes to the literature on design descriptions by providing a means of computing design ontologies, in the form of hypercube lattices, from a single BoM that is an integral part of a given design description. The lattice representing each design ontology forms a space for the configuration of structurally[2] valid BoMs for the design. For this reason, in contrast to Zhou et al., rather than transforming BoMs into other BoMs, we create a design configuration space that includes all possible BoMs and from which appropriate BoMs can be selected. In this way, the limitations of BoM transformation processes are avoided.

The approach introduced in this paper was inspired by work on computational design synthesis where the provision of computational support requires the manipulation of emergent shapes in design descriptions. An example of such a shape is the middle square formed from the two larger squares in Fig. 1 which sees this shape as composed of two large squares. The smaller square is not explicitly defined in the shape description but the requirement is that such shapes can be manipulated in the same way as explicitly defined parts of the description. In essence, we regard unarticulated BoMs as equivalent to the emergent smaller square in Fig. 1. The process used in shape computation to make the smaller square available in the design process is called “embedding” [15,16]. An important prerequisite to the implementation of embedding is a common representation scheme for all (explicitly defined and emergent) shapes in the design description. Similarly, the implementation of this functionality for design configuration requires a suitable representation scheme for all (explicitly defined and unarticulated) design configurations. Here, lattices are used as this representation scheme.

The realisation of shape grammar-based design synthesis requires a means of implementing the embedding of shapes into each other; Krstic and Chau et al. outline hypercube lattice-based solutions to this problem [17,18]. March [19] used lattices to describe geometric shapes, Stiny [16] used a lattice of parts to describe continuity in a sequence of shape rule applications and Krstic [17] used lattices in shape decompositions. In this example, the edges of the small square in Fig. 1(a) and (b) can be described as parts of edges of the bigger squares or as edges of the smaller, emergent, square. Using the latter description results in a description of the whole shape that includes the smaller square, which can then be manipulated to form the shape in Fig. 1(c). The definition of the whole shape is a network structure because each edge of the smaller square has two parents: the smaller square and one of the larger squares. The same idea is exploited in this paper for the reconfiguration of BoMs. The examples used in this paper are simpler than the shape example because the BoM definitions are tree structures where each element (parts of the product in this case) has only one parent. I.e., in a given BoM, each part can be a part of only one sub-assembly. However, in general, this is not the case and the use of hypercube lattices provides scope for future developments to cover BoMs that are network structures and to explore other kinds of design structure, such as function structures, and their relationships with BoMs.

3. Lattice theory

Lattices are widely used to represent concept classification structures in knowledge management applications [20]. An example of such a structure is shown in Fig. 2(a) which can be read as the concept of *mammals* includes (i.e., is an aggregation of) *dogs* and *cats* which themselves are an aggregation of *wild cats* and *pet cats*. Its representation as a lattice visualised using a Hasse diagram is shown in Fig. 2(b)

[2] Some configurations may not be feasible for other reasons but the structural integrity of BoMs in a given technical data package is assured.

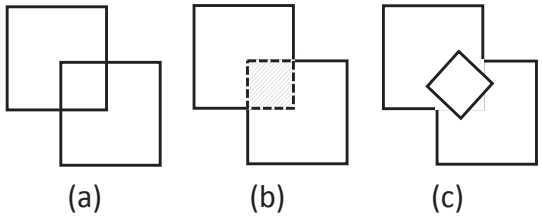


Fig. 1. Emergent shapes in a shape defined by two overlapping squares.

where relationships across tiers in the structure support classifications that span more than one tier, e.g., *wild cats* as *mammals*. A knowledge management application could use this lattice to support the classification of data. In such an application, users could be offered a choice to classify a *mammal* as either a *cat* or *dog* and, if *cat* was chosen, a further choice of *wild cat* or *pet cat*. In addition, *wild* and *pet cats* could be classified directly as *mammals*. An important point to note is that relationships in the lattice represent aggregation relationships; the semantics of classification are added in the knowledge management application.

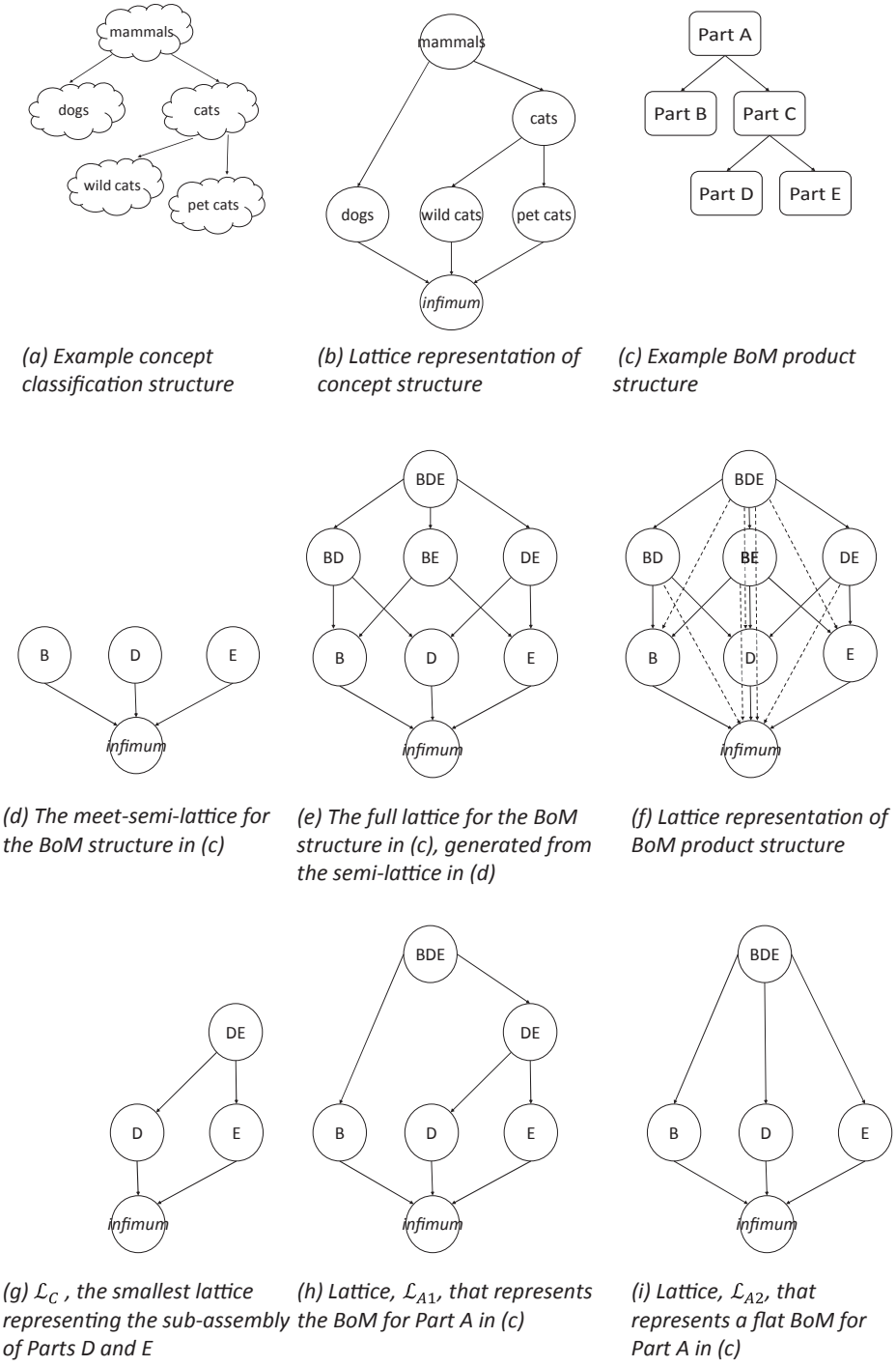
The lattice structure [21] is a mathematical formalism and provides precision in the representation of structure. We exploit these benefits and associate a different meaning with the aggregation relationships in the lattice, namely how parts aggregate into composite parts rather than how concepts aggregate. An example BoM, that is structurally the same as the concept structure in Fig. 2(a), is shown in Fig. 2(c). This can be read as, “*Part A* is an aggregation of *Parts B* and *C* which itself is an aggregation of *Parts D* and *E*.” The representation of this structure as a hypercube lattice is shown in Fig. 2(f) where, for clarity, the dashed lines show cross tier relationships that exist as pathways through the lattice (e.g., the relationship *AB* can be realised through the paths *BDE-BD-B* or *BDE-BE-B*) but are not an explicit part of the lattice. Like the concept structure, this can be read from the top down or the bottom up. That is, as the BoM structure itself or in the following way, “All parts in the lattice [including the null part represented by the infimum] are parts of *Part A*.” Similarly, “The infimum and *Parts D* and *E* are parts of *Part C*.” In essence, the hypercube lattice of all subsets of the parts of *Part A* captures all possible BoMs (i.e., parts and aggregation relationships) that could be used as configurations of *Part A*. The approach introduced in this paper shows how properties of lattice structures can be exploited to ensure the referential integrity of all BoMs in a given technical data package. We do this by using the lattice to provide the valid operations for design reconfiguration (the operation to transform one BoM to another), and the part names in the source BoM to provide the vocabulary for use in the reconfiguration of a specific design. The reconfiguration capability ensures that each part is only used once in a given configuration. The mathematical formalism that sits behind hypercube lattices facilitates this by enabling the calculation of lattice complements where each part is only used once.

A lattice [21,22] $\mathscr{L}$ is a partially ordered set L (poset) with relationship denoted as $\leq$ such that each pair of elements x, y has a unique least upper bound $x \vee y$ (join) and a unique greatest lower bound $x \wedge y$ (meet). Further the whole lattice has a global upper bound or supremum. A global lower bound or infimum is also defined. The lattice is denoted

$$\mathscr{L} = \langle L; \leq \rangle$$

Thus for the structure in Fig. 2(c), the lattice set is $L = \{A, B, C, D, E\}$ with leaf nodes B, D, E (which are the BoM nodes without subparts).

Let us confine attention to lattices whose elements are subsets of a set which is partially ordered by set inclusion. The underlying set for the part structure in 2(c) is $\{B, D, E\}$ with subsets $\{D, E\}$, $\{B\}$, $\{D\}$, $\{E\}$, and $\{B, D, E\}$. The whole set $\{B, D, E\}$ is the supremum of the corresponding lattice and the empty set $\{\}$ is the infimum.



(a) Example concept classification structure

(b) Lattice representation of concept structure

(c) Example BoM product structure

(d) The meet-semi-lattice for the BoM structure in (c)

(e) The full lattice for the BoM structure in (c), generated from the semi-lattice in (d)

(f) Lattice representation of BoM product structure

(g) $\mathcal{L}_C$, the smallest lattice representing the sub-assembly of Parts D and E

(h) Lattice, $\mathcal{L}_{A1}$, that represents the BoM for Part A in (c)

(i) Lattice, $\mathcal{L}_{A2}$, that represents a flat BoM for Part A in (c)

Fig. 2. Example structures and lattices. NOTE: Nodes BDE and DE in the lattices correspond to Parts A and C respectively in the BoM.

We demonstrate how to generate this lattice from its parts embedded in the lattice of all possible subsets. First the lattice of all subsets is generated from the component parts in the set $\{B, D, E\}$. We create an intermediate structure, a meet-semi-lattice [21, p8]. A meet-semi-lattice has a greatest lower bound for each pair of elements but not necessarily a least upper bound. A particular meet-semi-lattice is used to generate the whole lattice. This meet-semi-lattice is

$$S\mathscr{L} = \{B, D, E, \{\}\}$$

created from the component parts of the BoM (i.e., all nodes in the BoM that are not decomposed into other parts, i.e., the leaf nodes of the BoM) which results in the structure shown in Fig. 2(d). In this semi-lattice, each part is connected to the same infimum but there is no supremum.

To build a full lattice of all subsets, we use a generator algorithm [21,p p29-31] which, from the part nodes in $S\mathscr{L}$, generates the full hypercube lattice shown in Fig. 2(e) on the set $\{B, D, E\}$ with all its eight subsets $\{\{BDE\}, \{B, D\}, \{B, E\}, \{D, E\}, B, D, E, \{\}\}$ in Fig. 2(e). In essence, the full lattice creates a temporary design configuration space that contains all possible BoMs for a given collection of parts. As a result, for any pair of parts (represented as nodes) in the lattice, there exists a unique least upper bound and a unique greatest lower bound. This property is exploited in the detail of the implementation when calculating, for a given part, its parents, children and siblings. To create

a BoM, users navigate this space and select parts (both sub-assemblies and base parts) for the new BoM. For example, to define the lattice associated with the BoM structure shown in Fig. 2(c), a user might select nodes *D* and *E* from the full lattice in Fig. 2(e) which would then be used in a join operator to determine the smallest lattice representing the sub-assembly of the two parts, $\mathscr{L}_C$ (Fig. 2(g)). This could then be combined with node *B* to give a lattice, $\mathscr{L}_{A1}$, (Fig. 2(h)) that represents *Part A* in the BoM.

$\mathscr{L}_C = \{D, E\}, B, D, \{\}\}$

$\mathscr{L}_{A1} = \{\{B, D, E\}, \{D, E\}, B, D, E, \{\}\}$

Similarly, a second, flat BoM for *Part A*, with associated lattice, $\mathscr{L}_{A2}$, (Fig. 2(i)) could also be defined with respect to the same underlying lattice.

$\mathscr{L}_{A2} = \{\{B, D, E\}, B, D, E, \{\}\}$

In the context of the lattice structure, the number of possible remaining sub-lattices reduces as parts are added to the sub-lattice because each part can only be used once in a given BoM. As a result, the remaining available configuration options can be calculated using the lattice complement operation on sub-lattices in a universe that is the whole underlying lattice.

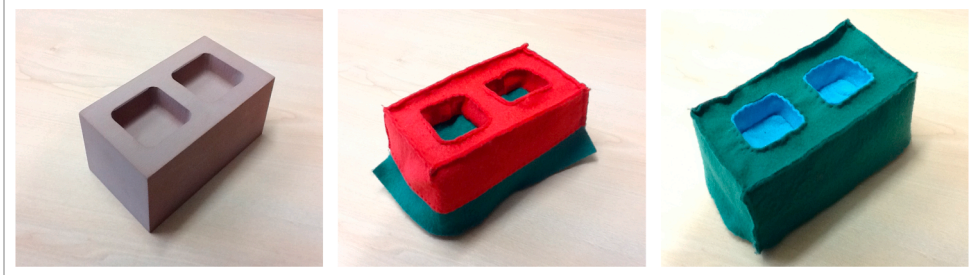
The hypercube lattices representing all subsets of a set (or equivalently all possible aggregations of parts in a product) are complemented distributive lattices. When a Hasse diagram is used to represent a partially ordered set (poset), the ordering relation is represented by a line adjoining two nodes that have different vertical positions, where the lower node is a part of the upper one. Their horizontal positions are immaterial. Other calculations can use the lattices that represent the different BoMs. For example, $\mathscr{L}_{A1}$ and $\mathscr{L}_{A2}$ could be compared with each other to identify their set of common parts or sub-assemblies.

The example used here is simple for reasons of clarity and assumes that each part in the BoM is represented once in the BoM. We do not yet support shared parts (where, for a given collection of shared parts, each part would be replicated in the lattice) or part-whole relationships where the parent contains multiple parts. Again, these would be implemented by replicating each occurrence of the part in the lattice. In both of these cases, if the parts needed to be grouped then this could be achieved using a join operation. A benefit of this, however, is that our part-whole relationships do not have any attributes so can be represented as parent–child relationships in the lattice.

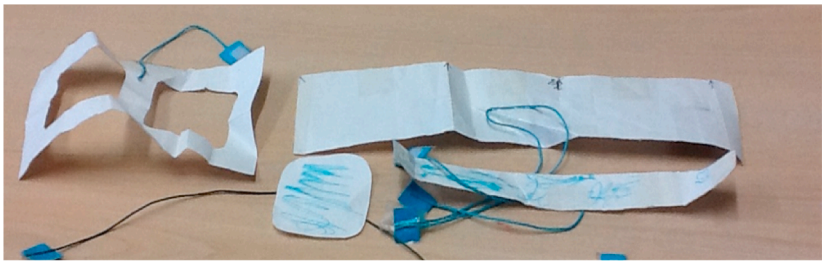
4. Research approach

A design prototyping approach was used to produce a series of five prototypes (one physical model and four software prototypes) that enabled the reconfiguration of design descriptions. The first prototype was a physical model that was used to illustrate and build insights on the problem of design reconfiguration using a 3D shape with different coats (see Fig. 3). The shape, a block with two pockets, is one where ribs are used in design and pockets in manufacture. In addition, when constructing the prototype, we found a third useful design description: one that related the pattern pieces of the coat to the design of the block. The coats were made from felt purchased in squares. At design time the availability of felt was unknown and in the future, if more coats were required, it is plausible that the felt would be purchased from a roll which would allow the outside piece to be made from one strip of felt. This is a simple example of where the design structure needed for manufacturing is not known during design, because it depends of the availability of materials, and so cannot be predicted at design time: hence the need to be able to create design structures as and when needed through the entire product lifecycle. In parallel, we illustrated embedding of these relationships in 2D using acetate film overlaid onto design descriptions for an industry case study [23]. Learning from the physical model was important in both improving detailed understanding of the problem and articulating possible benefits of the research to prospective end users.

A series of four software prototypes were then built. The first two demonstrated the technical feasibility of exporting BoMs from a CAD system (Solidworks) using a neutral format (ISO10303-214) to generate hypercube lattices (visualised as Hasse diagrams using the LatDraw software) and reading lattices back into the neutral format and the CAD



(a) block with pockets; (b) & (c) coats with design (ribs) and manufacturing (pockets) features



(d) pattern pieces for the coats

Fig. 3. The block, its coats and the pattern pieces for the coats & shape grammar example.

system. The second prototype also demonstrated the feasibility of embedding alternative BoMs into a given lattice but with a very slow speed of computation as the lattice size increased (e.g., 30 min to generate a lattice for a 15 part BoM). A hundred-fold improvement was achieved in the third prototype by optimising the implementation of the code but a size limitation in LatDraw led to the development of an alternative interface for viewing the lattice. Finally, the fourth prototype, StrEmbed-4[3], incorporates a purpose built lattice visualisation routine and a command line interface for reconfiguring BoMs. The remainder of this paper relates to the fourth software prototype.

5. Case study: A robotic arm

A robotic arm case study was used to demonstrate the engineering application of StrEmbed-4. Two alternative configurations are shown in Fig. 4. The configuration shown in Fig. 4(a) is used when a model that can simulate the functionality of the product is needed; its corresponding BoM, the engineering BoM (E-BoM) for the robotic arm, is shown in Fig. 4(c). A second configuration, the one used for shipping the robotic arm, is shown in Fig. 4(b) with its corresponding shipping BoM (S-BoM) in Fig. 4(d).

The data was produced by a researcher who created two SolidWorks models, one for each configuration, to represent the wider problem of design configuration. In practice, these models would be constructed from shared models of the component parts. However, this becomes less feasible when configurations, and associated models, are created across longer timescales and by different people. In addition, the feasibility of creating all product configurations in CAD or an associated PLM system becomes even less feasible when the people creating new configurations are located in different organisations and at different stages the product lifecycle, often without access to the design definition. For this reason, research on issues in the wider implementation of design configuration tools would need richer case study data. However, this data is sufficient here because the research is considering the theoretical foundations of future design tools to support the configuration of BoMs rather than their wider application.

6. A Software tool for design configuration

Either of the case study BoMs, or any other BoM that includes all of the component parts of the robot, can be used to generate the hypercube lattice shown using grey fine lines in Fig. 5. This lattice, generated using the approach described in Section 3, provides a design configuration space that includes all possible BoMs for the design from which it was generated. The two case study BoMs, along with any other valid BoM, can then be embedded in the lattice. In Fig. 5, the case study SBoM is shown in heavy grey solid lines and the EBoM is shown in grey dashed lines. In this section, we demonstrate how StrEmbed-4 can be used to navigate the design configuration space and define any valid BoM as a sub-lattice within this space.

The lattice generation process begins with a shape model which is translated into an ISO10303-214 file for input to StrEmbed. For the case study, the EBoM was used to generate this lattice. A screenshot of the EBoM embedded in the underlying lattice is shown in Fig. 6(a). The window includes two panels: the assembly tree on the left-hand side and the Hasse diagram on the right-hand side. The Hasse diagram allows users to visualise the BoM they are working with in the context of the underlying lattice and parts (nodes in the lattice) can be selected using mouse clicks and edited using the editor window shown in Fig. 6(b). The editing options in the editor window are based on lattice algebra and allow sub-lattices, each representing a specific BoM, to be defined and manipulated. This capability provides editing operations that are sufficient to transform between any pair of valid BoMs.

[3] StrEmbed-4 is available from https://doi.org/10.5518/227

The following operations on the sub-lattice that represents the BoM being edited are supported.

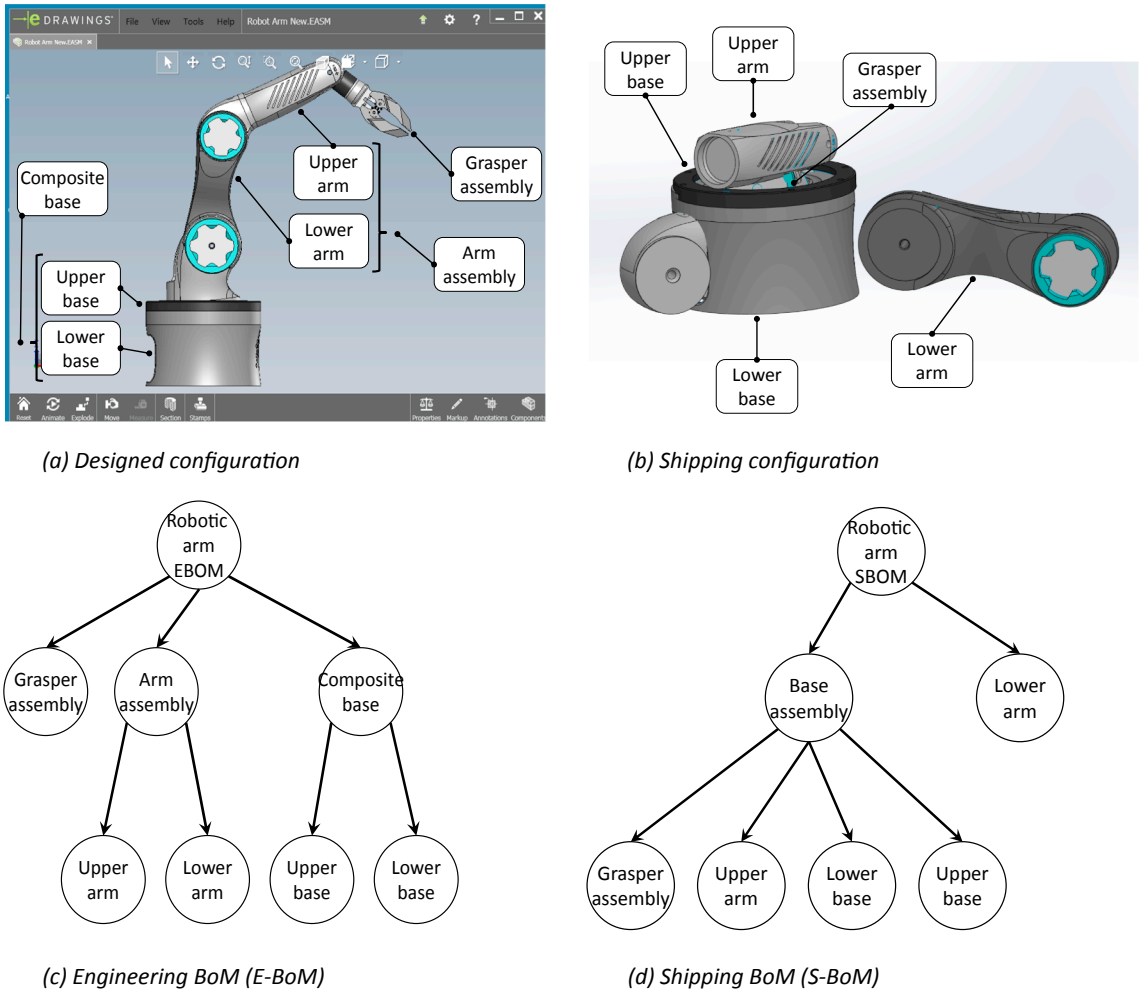
- **Insert before & Insert after**: reorders siblings under a given node. This does not change the structure of the sub-lattice but allows users to adjust how they visualise it.
- **Adopt**: moves a selected node to a [new] parent node in the sub-lattice.
- **Assemble**: creates a new sub-assembly in the sub-lattice by selecting the node that is parent to the two selected nodes in the underlying lattice.
- **Collapse**: removes a sub-assembly node from the sub-lattice. The parts of this node are re-parented to their original grandparents.

In the case study, the reconfiguration of the SBoM from the EBoM requires five steps: **collapse** the arm assembly in the EBoM, **assemble** the upper arm and grasper assembly to form ASSY_1, **assemble** ASSY_1 with the composite base, **collapse** ASSY_1, and **collapse** the composite base. The results of these steps are shown in Fig. 7. In this way, we have demonstrated the feasibility of using embedded lattice structures for the reconfiguration of BoMs within a design configuration space. There are two key limitations in the software. Firstly, typical BoMs often have in excess of 100 parts but, beyond seven component parts which results in 2^7 lattices, it is not possible to see the lattice using StrEmbed-4 because the underlying lattice is too large. Secondly, regardless of the speed of computation, the lattice generation process is not scalable because the size of the lattices grows exponentially with the number of parts in the BoM. To address these issues, in future developments, we propose calculating the relevant parts of the underlying lattice on the fly, as needed, which will avoid the need to generate or visualise entire underlying lattices.

7. Discussion

60% of a typical technical data package is related to shape. An important benefit of today's geometry-based shape design systems is that any shape definition created by a user is guaranteed to be a valid solid. This is because theories such as the constructive solid geometry formalism that underpins many such systems [24] provides a theoretical foundation for the description of design shapes that combine primitive shape elements (half-spaces) and relationships that are allowable operations based in Boolean algebras between shapes. As a result, the behaviour of such shape descriptions is predictable and, although it is possible to describe shapes that are not the ones intended by the user, all defined shapes are created through valid operations on valid solids. The aspiration of the research reported in this paper is to create a comparable capability for the definition of design configurations such as BoMs. To this end, we have shown that a boolean hypercube lattice containing all possible BoMs for a given design can be calculated from a single description that includes any valid aggregation of the design's parts. This lattice, in turn, can be used as a mathematically defined space within which new BoMs can be configured. Table 1 highlights key functionality of today's constructive solid geometry-based 3D CAD systems and the design configuration editors we envisage. Rudimentary implementations of each feature have been demonstrated in the software prototype presented earlier.

The nearest academic work on design configuration in this area is that of Zhou et al. [3] who describe operations needed to transform an as-built BoM into one to support through-life support operations. The work reported in this paper relates to product design and development, where parts are identified by design-related identifiers such as part and/or drawing number. On the other hand, the transformation process in [3] begins with an as-built BoM (where parts are identified by design- and production- related identifiers) and translates it into one where individual physical parts are identified, for example, by serial number and physical location. In addition, engineering change is within



(a) Designed configuration

(b) Shipping configuration

(c) Engineering BoM (E-BoM)

(d) Shipping BoM (S-BoM)

Fig. 4. Two configurations of the robotic arm case study.

the scope of Zhou et al. but not here. For these reasons, Zhou et al.'s 'replace node' operation does not have an equivalent in the approach presented here. However, their other three node operations (node addition and deletion, and the creation of intermediate nodes) could be supported using lattices. Further work would be needed to deal with the different part (and so node) identifiers but one approach could be to generate (from a design description) an as-built lattice (with parts identified by serial numbers) into which different service BoMs could be embedded. For both approaches, however, although it is possible to define different configurations for a given product there is no way, over and above the use of human expertise, to determine whether a given BoM, e.g., a shipping BoM, is the best BoM to support the shipping process. Our current thinking is that creating ways to select the optimal BoM for a given process would require the integration of alternative BoMs with simulations of the target process. Early work on this is reported in [25].

With respect to the success criteria introduced earlier (reduce data duplication in design descriptions, provide improvement opportunities for the management of change and allow new BoMs to be defined as and when needed through the entire product lifecycle), we have demonstrated steps towards satisfying each criterion. In contrast to dendograms, where a user re-enters part identifiers (such as part names or numbers), users select parts from the underlying lattice. In this way, part identifiers are not duplicated and scope for the introduction of errors is reduced. The lattices and BoMs that users can work with are currently small in size but this research is paving the way to a new generation of design tools that support users in navigating design configuration spaces and selecting BoM structures rather than creating new ones. With respect to the management of change, this removes duplication of data. In this way, if parts were added to or removed from an underlying lattice then discrepancies between sub-lattices embedded in the original host lattice would be straight forward to detect by using lattice algebra to compare the original sub-lattices with the new host. We have not yet carried out experiments in this area, and new software prototypes would be needed to enable them, but lattice theory provides the mathematical formalism that would be needed to realise such prototypes. Finally, we have demonstrated that the approach allows new BoMs to be defined as and when needed against an underlying lattice into which other BoMs have already been embedded. There are no restrictions on the number of sub-lattices that can be embedded into a given host lattice.

However, significant further work is needed to deliver design configuration editors, possibly as core parts of future PLM systems [26], before the internal consistency of technical data packages can be assured. The future functionality of such an editor, using the reconfiguration example given in Fig. 8, is illustrated in Table 2 where key steps in the reconfiguration process are shown along with current capabilities in Str-Embed-4 and research challenges that would need to be addressed to deliver the full functionality. Early, paper-based experiments on how lattice algebras can be used to calculate valid options for a reconfigured BoM indicate that being able to apply the complement operation to lattices will be critical to achieving this. Further development of the software also needs consideration of engineering practice. For example, configuration engineers (as opposed to CAD

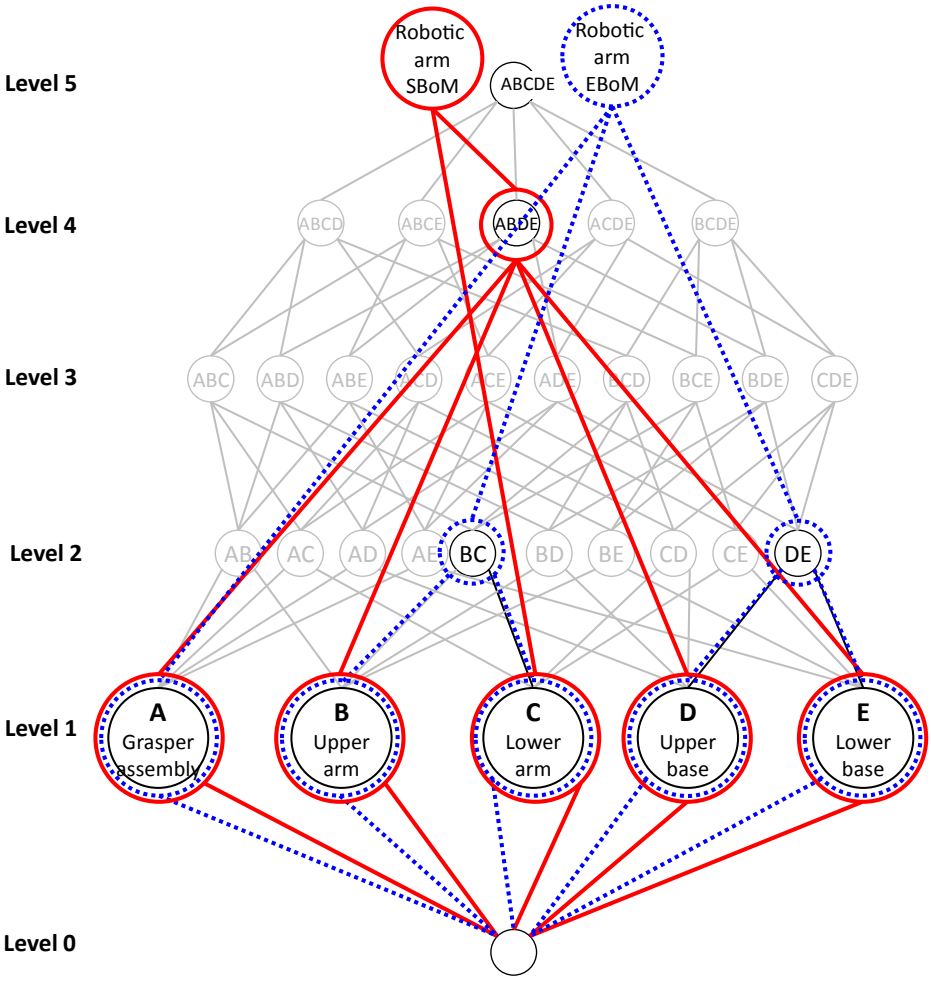
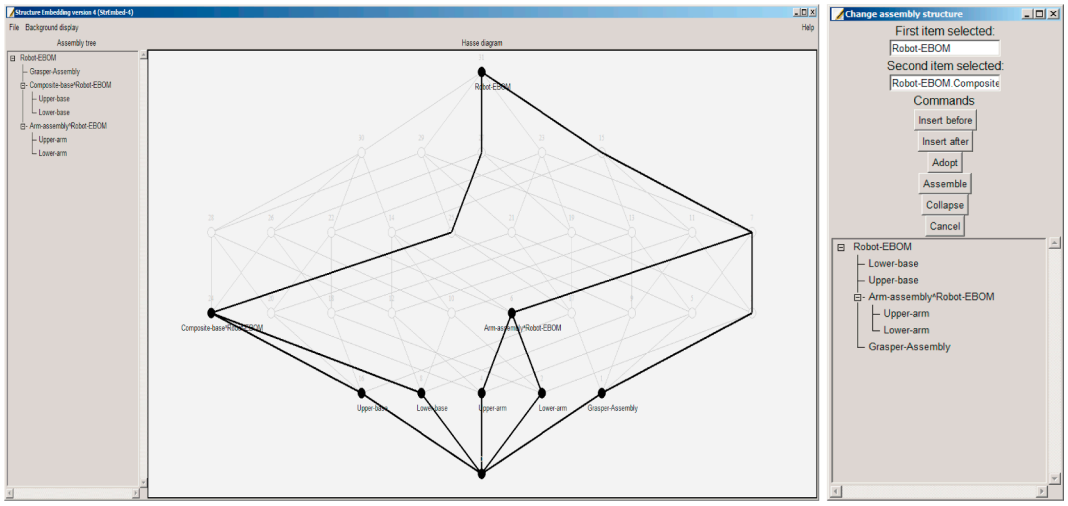


Fig. 5. Robotic arm case study showing two different BoMs embedded into a common underlying lattice.

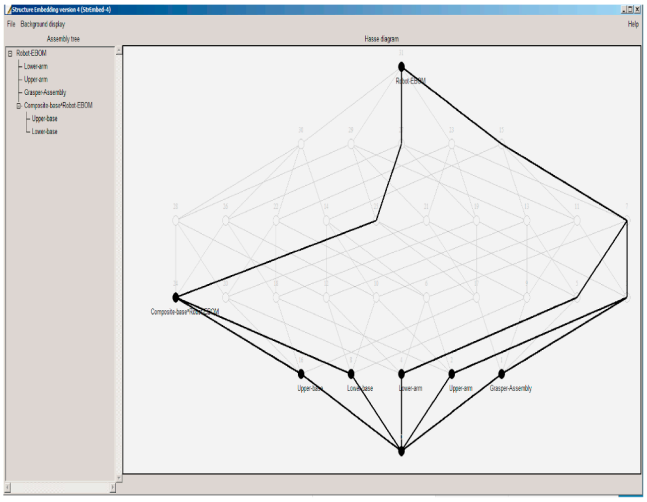


(a) Lattice and sub-lattice representing the EBoM (b) Editor window

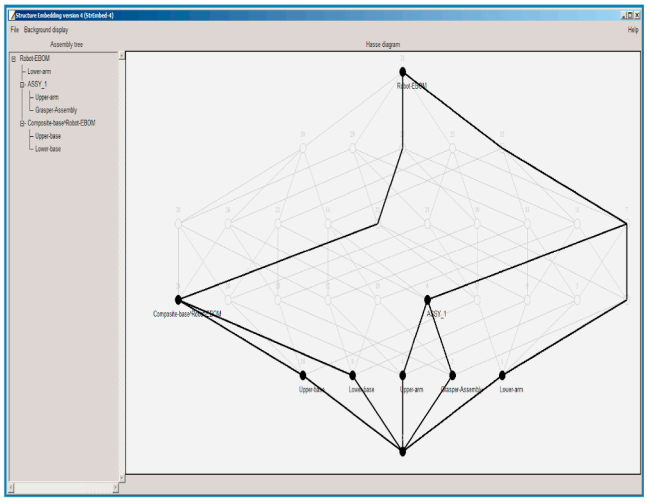
Fig. 6. Screenshots of the EBoM and underlying lattice generated using StrEmbed-4.

users) tend not to work with BoMs of entire products which are too big. They typically work across two or three layers in the system architecture, e.g., components in a subsystem, and this could be exploited in developing industry strength tools that do not require the entire underlying lattice.

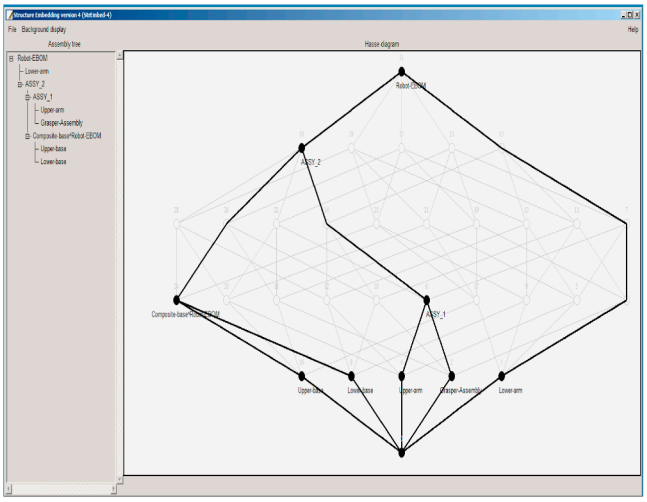
There are also broader research challenges to be addressed in both software development and support for configuration engineers. The software prototype introduced in this paper operates on lattices that have been generated from shape models but the connections to these design descriptions are not maintained. How such relationships might



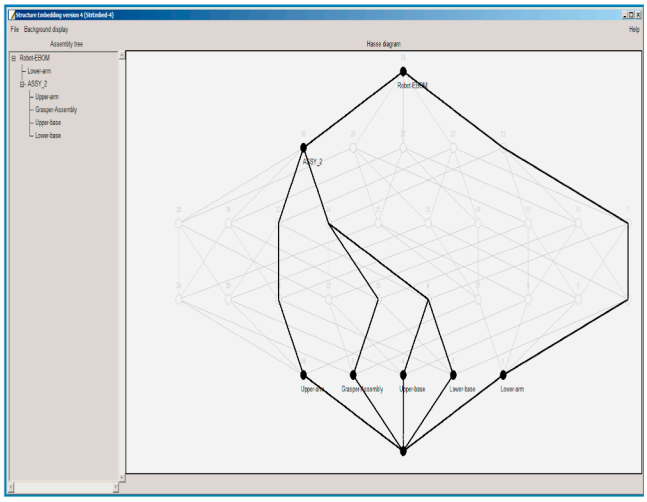
(a) Sub-lattice after collapsing the arm assembly in the EBoM



(b) Sub-lattice after assembling the upper arm and grasper assembly to form ASSY_1



(c) Sub-lattice after assembling ASSY_1 with the composite base to form ASSY_2



(d) Sub-lattice after collapsing ASSY_1 and the composite base to form the SBoM

Fig. 7. Operations to reconfigure the EBoM to the SBoM.

Table 1
Comparison of current 3D CAD system architectures and envisaged BoM editors.

Feature	3D [CSG-based] CAD	BoM editor
Mathematically defined workspace	The 3D space within which shapes are defined	A lattice space: composed of nodes and arcs from a lattice generated from a CAD model that includes a part hierarchy
Mathematically defined objects to work with (nouns in a design language)	Half-spaces or geometric primitives defined using them, e.g., block, cylinder, cone	Parts and part-whole relationships in the lattice. Names of parts for a given design will come from the source CAD model.
Permissible operations (verbs in a design language)	Boolean operations from Constructive Solid Geometry	Operations drawn from lattice algebra

be maintained or re-established has not been considered but this would be needed in an industry strength solution. Relationships with shape models could also be important in supporting the visualisation capabilities that engineers might prefer to use when interacting with BoMs and other design configurations. The current software prototype has enabled initial explorations of how configuration engineers might interact with lattices. Our conclusion, however, is that they should not be interacting with the lattice. Instead, the lattice should remain hidden and be used to guide the engineer who would see the BoM and available configuration options but not the entire lattice which is too large to see, let alone work with.

In supporting configuration engineers, this paper provides insights on the kinds of design configuration tools that are becoming feasible. However, further work is needed to better understand the requirements for such tools. For example, a fuller version of the case study used in this paper is likely to include additional items, such as packaging materials, in the underlying lattice for the shipping BoM. In sectors where specialist equipment (such as ground support equipment and tooling in

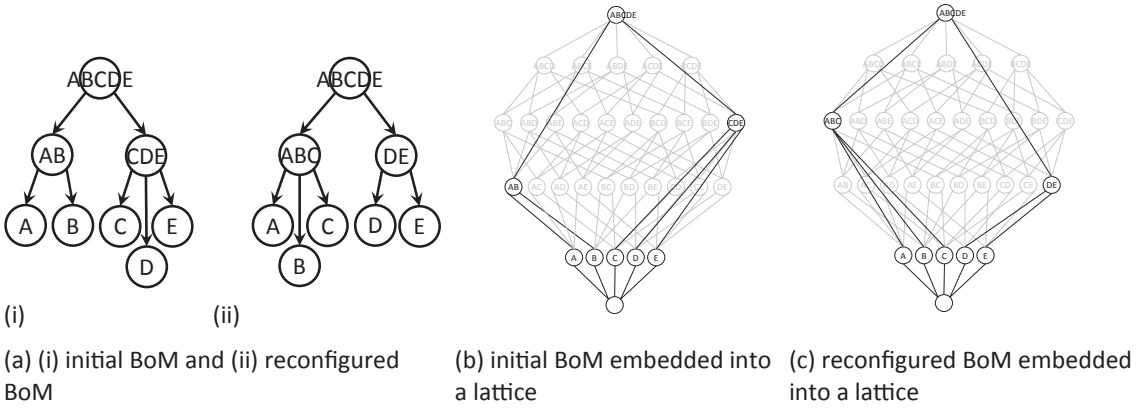


Fig. 8. Example of a BoM reconfiguration problem (changing from (AB, CDE), to (ABC, DE)). Both initial and reconfigured BoMs are embedded into a common lattice. The lattice forms a design configuration space of all possible valid BoMs and provides a structure within which such BoMs can be defined.

the aerospace industry) is needed for manufacturing, this could be a requirement through the life of the product. In addition, further work is needed to understand the consequences and opportunities of this research in the context of engineering change [27] and the configuration of other kinds of design structure including bills of processes and function structures, and the range of design structures needed to support the design of product families and their variants [28]. This paper relates to the different configurations needed by a single product (comparable to an individual member of a product family) to support different engineering activities through its life [29]. Different issues occur in the configuration of BoMs and other architectures when designing and using product families [30]. We are confident that the lattice-based approach introduced in this paper has the potential to contribute to the design of product families but further work is needed to verify this. For example, it appears feasible to generate a lattice from a generic BoM and associated configuration options into which BoMs for specific variants could be embedded.

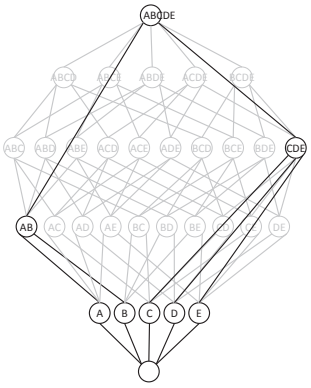
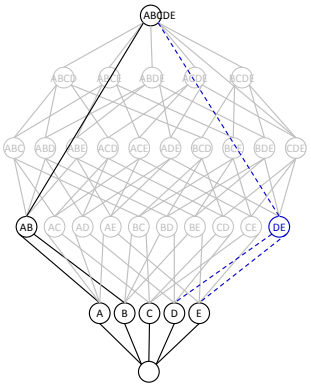
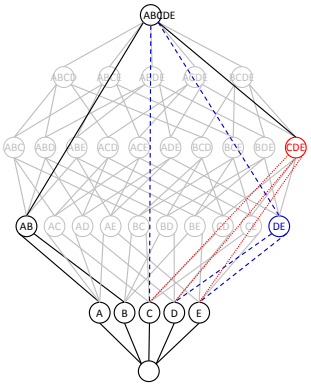
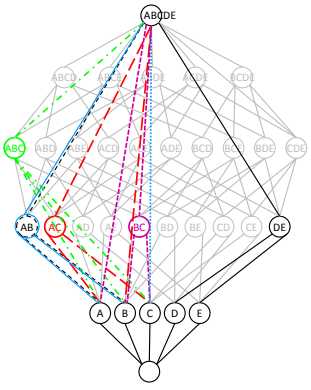
8. Conclusions & future work

There is a latent industry need to be able to associate multiple BoMs with one of more descriptions of a given design. This need has remained hidden because current design technologies tend to subsume BoMs within proprietary data representations. However, engineers use BoMs and other design structures as lenses to repurpose design descriptions for specific purposes. For this reason, new design technologies are needed that make BoMs and other design structures available for engineers to work with directly but without compromising the functionalities of existing design tools and technologies. Decisions on this are likely to need to take into account factors including the perspectives and preferences of users, organisational and regulatory drivers, and technical feasibility. These issues are likely to become more difficult when such design configuration tools are used in supply networks and other contexts where a full design definition is unlikely to be available. For example, will customers provide suppliers with reconfigured BoMs and, if so, how will they share an embedded BoM without sharing the underlying design description? And, from a business perspective, how might such a capability change engineering practice and improve product development system performance?

In this paper we have demonstrated how embedding, implemented using lattice theory, might be used to underpin such a technology using a simplified robotic arm assembly as a case study. The simplification process reduced the number of parts by aggregating assemblies into components and removed details of the joint assemblies. This was appropriate because the purpose of the case study was to demonstrate an outline architecture for a potential new design technology, whose maturity would position it at the lowest Technology Readiness Levels (1 or 2). The case study included a CAD model that allowed the design shape to be visualised and two BoMs: one, an engineering BoM, and a second, a shipping BoM, for use in transportation of the final product. Typical current practice is that one of these BoMs, probably the engineering BoM, would be an integral part of the CAD model and other BoMs would be developed separately as and when needed. A problem with this approach is that the secondary BoM is not connected to the original one. Discussions with practitioners indicated that this leads to errors in secondary BoMs and makes the management of change challenging leading to errors and rework that have a detrimental impact on the performance of the design process. The approach introduced in this paper overcomes these problems. By providing a language, in the form of a design configuration space, where the grammar is derived from lattice theory and the vocabulary from the source design description, scope for errors in secondary BoMs is dramatically reduced because it is not possible to incorrectly name parts or create configurations of parts that are inconsistent with the source. In the future, if lattices can be related to underlying design descriptions then opportunities for new ways of managing engineering change are likely to emerge. The design structures considered in this paper, BoMs, are simple tree structures of discrete parts. Support for more complex design structures is also needed if product development processes are to be supported and improved. Examples of such structures include function structures (where the elements are functions rather than parts), more network-like structures (such as the physical manifestation of a given function which may be part of a number of discrete parts) and process structures (which in this research have been referred to by industry partners as “Bills of Processes”). A key benefit of using lattices is that they provide a simple representation scheme that can be used to support machine learning and other applications where large data sets with simple meta-structures are required. We are currently exploring the use of artificial intelligence technologies to reason with design structures represented as lattices which, if successful, could create opportunities for new forms of design automation.

A new generation of computer aided design tools that allow designers to navigate and explore design spaces is being realised. For example, Chen et al. [31] report work that uses an embedding operation to support the exploration of spaces populated with design shapes; a key challenge identified by Chen et al. lies in ensuring that the population of shapes captures the full range of possible geometric variability. Here we have introduced a theoretical foundation that enables the exploration of a different kind of design space: one that is populated with possible configurations of a given design. The lattice-based approach ensures that all possible configurations are members of this population. Both Chen et al. and this paper refer to an embedding process. Chen et al.’s process involves mapping between higher and lower dimensionality design [shape] spaces whereas our embedding

Table 2
Potential future functionality of a lattice-based design configuration editor.

Target capability	Lattice illustration	Current software	Research challenges
The initial BoM, is imported from CAD (using a neutral format such as ISO 10303) and made visible on an underlying lattice which has been generated from the initial BoM. To reconfigure the initial BoM, *ABCDE (AB, CDE)*, to *ABCDE (ABC, DE)*, the user starts by disconnecting C from CDE.		StrEmbed-4 can generate a lattice for 15 parts in < 1 s but the time taken still grows exponentially with the number of parts, affecting scalability and so is not workable for an interactive editor. The lattice and embedded BoM can be displayed, but not in a way that a user could easily interact with.	For large products, such as an aero engine with over 30,000 parts, how can engineers be supported to (a) bound the depth & breadth of the BoM from which the lattice is generated? (b) express BoMs for analytical/ reporting purposes, e.g., generate textual formats from the editor?
The editor highlights alternative configurations given this break in the structure. The user selects Part DE, including its substructure and arc to the supremum (dashed lines in the illustration). This leaves Part C as a floating part with no link to the supremum (see centre bottom of the illustration) which means that the current BoM is not a valid BoM for the design.		StrEmbed-4 has a rudimentary interface for reconfiguring BoMs. Alternative structures are not visible to users. Invalid BoMs cannot be defined because they would be invalid w.r.t. lattice which is not linked to geometry.	(a) What support will let users see allowable BoMs and shape models of individual parts for a given design? (b) Is there potential for learning from other sectors where support for the visualisation of large structures is provided [24,26]? (c) How might users wish to adjust the underlying lattice? E.g., add new parts, aggregate parts (to simplify the BoM or lattice), disaggregate a part so that its parts can be manipulated (d) How will such lattice changes affect geometric definitions of the product?
The editor provides options to resolve the discrepancy with C. Options, based on the underlying lattice, could include - omit C from the new BoM - highlight and let the user select other options that include C and DE, possibly with other configurations of ABC. These options (shown by dashed and dotted lines in the illustration) can be calculated using the underlying lattice and the defined part of the embedded lattice.		We can calculate all possible alternative BoMs, because the lattice space makes the problem bounded and all options exist in the underlying lattice. However, an implementation has not yet been built. This space would be dynamic in that each design would have its own space, generated from a CAD hierarchy view, and changeable, e.g., if parts were aggregated or disaggregated.	Is it possible to build a sufficiently efficient implementation to compute all options from a given node and excluding identified parts? If this is feasible, what opportunities does this create for engineering practice?
The editor allows the user to select a valid option and a valid BoM is formed. The options are illustrated here using different line styles.		This functionality is not yet implemented. However, given the presence of the underlying lattice and a partially defined BoM represented as a sub-lattice, it is feasible to calculate all remaining options using lattice operations such as the complement operator.	For realistically sized BoMs, a significant challenge would lie in building interfaces that engineers can interact with effectively. In practice, BoMs are often linked to shape descriptions of the product and reconfiguring a product can substantially alter the geometric layout of the product. For example, the case study used in this paper had two geometric layouts for the same parts. Linkages to geometry were beyond the scope of this research but would need to considered in the deployment of lattice-based solutions for design configuration. Arcs in the lattice represent the inclusion of one part in a design configuration. Further work is needed to consider how groups of parts might be represented using lattice structures.

process is a general mathematical operation that allows one mathematical structure [a lattice] to be a consistent substructure of another. Further work is needed to understand how these embedding operations relate to each other.

The theoretical contribution of this paper is an application of boolean hypercube lattices that provides a representation scheme for part-whole structures [32] and the grammatical rules for design configuration where part-whole relationships are manipulated. When reconfiguring a design using this approach, new configurations are defined in terms of a common underlying lattice which is generated from a design BoM but can be used through the life of a product without changing, or needing access to, an original design description. This has the potential to provide a well-founded underpinning for BoM configuration management tools[4] that could sit at the heart of future generations of design system with the potential to transform the performance of product development systems. However, there are also shorter term opportunities. The software tool introduced in this paper uses a design description that is imported from a commercially available CAD system through a standard (ISO 10303-214) interface format. For this reason, there are also opportunities to deliver shorter term impact on industry practice, for example, by developing a BoM editor that can import from and export to a range of specialist design tools, preserving consistency of all other data (e.g. shape information), or initialising new BoM structures that are consistent with an existing design structure produced by a different design tool. In this way, there is a viable route to market that does not involve adaptation of existing design tools.

Design information is critical to the effective and efficient manufacture, production and through-life support of engineered products. Engineering design informatics has delivered significant advances in support for shape-based design information since the 1960s [7]. Progress for non-shape-based information, such as the BoMs that engineers use to adapt design information for specific activities, has made less progress, resulting in sub-optimal experience-based solutions that lack the systemic foundations needed for the development of robust computational methods and tools. In this paper we have demonstrated that hypercube lattices can act as computational spaces within which BoMs can be configured. For the future, the lattices should remain in the background, as a part of the technical apparatus. The focus of future work will be on the assurance of consistency within technical data packages; this requires product description operations (such as deleting, dividing and fusing parts) that allow users to change product structures and the compositions of design descriptions.

Declaration of Competing Interest

The authors declare no conflict of interest.

Acknowledgement

The models of the robot arm presented in this paper were developed with support from Peter Hayward (University of Leeds) and based on files from a Zortrax online library. The University of Leeds Advanced Computing Team analysed early versions of the StrEmbed code that contributed to the significant speed improvements referred to in this paper.

Funding

This research was funded by the UK Engineering & Physical Sciences Research Council (EPSRC) under Grant Number EP/N005694/1, “Embedding design structures in engineering information”.

[4] Configuration management is identified as one of the eight core elements of a PLM system [33].

Appendix A. Supplementary material

Supplementary data to this article can be found online at https://doi.org/10.1016/j.aei.2019.100928.

References

[1] NIST, 2017, Model-Based Enterprise Summit 2017, Available from https://www.nist.gov/news-events/events/2017/04/model-based-enterprise-summit-2017 (accessed on 31st May 2019).
[2] A.K. Behera, A. McKay, H.H. Chau, A. de Pennington, M.A. Robinson, Embedding design descriptions using lattice structures: Technical requirements, user perspectives and implementation, Smart Innovation Syst. Technologies. pp 557–566 (2017) 557–566.
[3] C. Zhou, X. Liu, F. Xue, H. Bo, K. Li, Research on static service BOM transformation for complex products, Adv. Eng. Inf. 36 (2018) 146–162.
[4] R. Helmer, A. Yassine, C. Meier, Systematic module and interface definition using component design structure matrix, J. Eng. Des. 21 (6) (2010) 647–675.
[5] S. Kubler, K. Framling, W. Derigent, P2P Data synchronization for product lifecycle management, Comput. Ind. 66 (2015) 82–98.
[6] S. Frechette, P. Huang, M. Carlisle, Model Based Enterprise Technical Data Package Requirements, National Institute of Standards and Technology, US Department of Commerce, NIST, Maryland, USA, 2011.
[7] C. McMahon, Design informatics: Supporting engineering design processes with information technology, J. Indian Inst. Sci. 95 (4) (2016) 365–377.
[8] B. Kassel, The myth of the single authoritative source: Overlap and evolution of product data through the total lifecycle, 2017. https://www.nist.gov/file/361516 (slide presentation accessed on 31st May 2019).
[9] E. Brynjolfsson, L.M. Hitt, Beyond the Productivity Paradox: Computers are the Catalyst for Bigger Changes, Communications of the ACM, August, 1998.
[10] M. Liu, J. Lai, W. Shen, A method for transformation of engineering bill of materials to maintenance bill of materials, Rob. Comput. Integr. Manuf. 30 (2) (2014) 142–149.
[11] C.G. Yin, Y.S. Ma, Parametric feature constraint modeling and mapping in product development, Adv. Eng. Inf. 26 (3) (2012) 539–552.
[12] C. Li, C. McMahon, L. Newnes, Annotation in product lifecycle management: A review of approaches, in: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, ASME, 2010, pp. 797–806.
[13] L. Ding, A. Ball, J. Matthews, C. McMahon, M. Patel, Annotation of lightweight formats for long-term product representations, Int. J. Comput. Integr. Manuf. 22 (11) (2009) 1037–1053.
[14] M. Kashkoush, H. El Maraghy, Product design retrieval by matching bills of materials, J. Mech. Des. 136 (1) (2014) 011002–011010.
[15] G. Stiny, The algebras of design, Res. Eng. Des. 2 (1991) 171–181.
[16] G. Stiny, Shape: Talking about seeing and doing, The MIT Press. (2006) 167–169.
[17] D. Krstic, From shape computations to shape decompositions, in: J.S. Gero (Ed.), Design Computing and Cognition '16, Springer, Switzerland, 2016, pp. 249–266.
[18] H.H. Chau, A. McKay, C.F. Earl, A.K. Behera, A. de Pennington, Exploiting lattice structures in shape grammar implementations, AIEDAM 32 (2) (2018) 147–161, https://doi.org/10.1017/S0890060417000282.
[19] L. March, The smallest interesting world? Environ. Plan. B: Plan. Des. 23 (1996) 133–142.
[20] B. Ganter, R. Wille, Formal Concept Analysis, Springer, 1996.
[21] G. Grätzer, Lattice Theory: First Concepts and Distributive Lattices, San Francisco, 1971.
[22] G. Szász, Introduction to Lattice Theory, The Publishing House of the Hungarian Academy of Sciences, Budapest, 1963.
[23] A. McKay, G. Sammonds, S. Ahmed-Kristensen, A. Irnazarow, M.A. Robinson, Using embedded design structures to unravel a complex decision in a product development system, International Conference on Engineering Design, ICEDVancouver, Canada, 2017, pp. 149–158.
[24] A.A.G. Requicha, H.B. Voelcker, Solid modelling: Current status and research directions, IEEE Comput. Graphics Appl. 3 (7) (1983) 25–37.
[25] A. McKay, R. Baker, R. Chittenden, A. de Pennington, A framework for students to visualize the implications of design decisions in global supply networks, in: Engineering and Product Design Education, 6 & 7 September 2018, Dyson School of Design Engineering, Imperial College, London, 2018.
[26] A. Bouras, B. Eynard, S. Foufou, K.D. Thoben, Product Lifecycle Management in the Era of Internet of Things: 12th IFIP WG 5.1 International Conference, PLM 2015, Doha, Qatar, October 19-21, 2015, Revised Selected Papers, Springer International Publishing, 2016.
[27] E. Subrahmanian, C. Lee, H. Granger, N.D. Grp, Managing and supporting product life cycle through engineering change management for a complex product, Res. Eng. Des. 26 (3) (2015) 189–217.
[28] N.H. Mortensen, C.L. Hansen, L. Hvam, M.M. Andreasen, Proactive modelling of market, product and production architectures, in: International Conference on Engineering Design, ICED 2011Technical University of Denmark, 2011. pp. 133–144.
[29] D.J.L. Siedlak, O.J. Pinon, P.R. Schlais, T.M. Schmidt, D.N. Mavris, A digital thread approach to support manufacturing-influenced conceptual aircraft design, Res. Eng. Des. 29 (2) (2018) 285–308.
[30] S.J. Jung, T.W. Simpson, An integrated approach to product family redesign using commonality and variety metrics, Res. Eng. Des. 27 (4) (2016) 391–412.
[31] W. Chen, M. Fuge, J. Chazan, Design manifolds capture the intrinsic complexity and dimension of design spaces, J. Mech. Des. 139 (5) (2017) 051102–051110.
[32] P. Simons, Parts: A study in ontology, Clarendon Press, 2000.
[33] Atos Origin, Product Lifecycle Management white paper, Available from https://www.aras.com/papers/white-paper-atos-origin-enterprise-product-lifecycle-management.pdf (accessed on 31st May 2019).