APPLICATION OF TENSOR FACTORISATION TO ANALYSE SIMILARITIES IN CAD ASSEMBLY MODELS

Flavien Boussuge[1], Christopher M. Tierney[1], Trevor T. Robinson[1], Cecil G. Armstrong

[1]*Queen's University Belfast, Belfast, U.K., f.boussuge@qub.ac.uk*
[2]*Queen's University Belfast, Belfast, U.K., christopher.tierney@qub.ac.uk*
[3]*Queen's University Belfast, Belfast, U.K., t.robinson@qub.ac.uk*
[4]*Queen's University Belfast, Belfast, U.K., c.armstrong@qub.ac.uk*

ABSTRACT

Generating fit-for-purpose CAD models from complex assemblies is time consuming for analysts. Tedious tasks include to identify and isolate the components of interest for the analysis, remove duplicate components, or correct inconsistent components' interfaces are common for large assemblies during the product development process. In this paper a new approach to help engineers analyse the consistency of CAD assembly models is proposed. The method utilises a tensor factorisation technique developed for relational machine learning and applies it on B-Rep topological and geometrical relations. The generated decomposition is used to identify which entities in the assembly are similar (within a threshold) to a selected input entity. The factorisation model regards globally all input relationships, e.g. the connections between components, to identify similar entities based on their relationships in the relational domain. It is shown that a hierarchical clustering method can group entities based on the similarities of their attributes and relationships.

Keywords: CAD/CAE Integration, Relational Learning, Computer Aided-Design, Assembly Representation,

1. INTRODUCTION

The pre-processing of digital mock-ups for multi-physics simulation requires identification of components anticipated not to have an influence on the desired results and removing them from the simulation model. Before starting the geometric adaption and simplification of the CAD objects, for large assembly the user should initially extract and determine the components of interest for the analysis. Often, CAD assemblies extracted from Product Lifecycle Management systems contain data inconsistencies such as missing components, duplicated entities, misalignments of parts, or poor-quality B-Rep geometry and topology. Consequently, designers and analysts spend considerable time correcting the CAD input assembly in order to obtain a model that is fit-for-purpose for finite element analysis and ready to be meshed. In addition, identifying similar geometric configurations in assembly is highly beneficial to avoid repeating pre-processing tasks.

With the rapid growth of relational and network data in social media modeling, bioinformatics and the semantic web, relational machine learning methods [1] have been proposed to learn from information represented in the form of relations between entities. The objective of relational learning is to build a model of the relational domain, where the data can be incomplete, noisy or contain false information [2]. The data is in the form of a graph, where nodes represent entities and edges the relations between the entities. Typical applications are the prediction of links in social networks or the identification of duplicated or missing entities in incomplete knowledge bases. In particular, tensor factorization techniques [1,3] have shown capabilities to process large multi-relational knowledge bases from the semantic web Linked Open Data [4,5], consisting of millions of entities, hundreds of relations and billions of known facts on a standalone computer.

The CAD assemblies produced by most industrial CAD systems (e.g. CATIA or Siemens NX) are comprised of B-Rep component models which provide direct topological and geometrical information. These can be represented in the form of a graph. A B-Rep topological graph, as shown in Figure 1, can be transformed into RDF triples [6] (e.g. Face-is_bounded_by-Edge) and used as input data for relational learning methods. Geometrical attributes, such as face type, edge convexity, etc… can similarly be transferred as known facts (e.g. Edge - has_convexity - Convex). The combination of these topological and geometrical attributes with information on connections between components provides meaningful descriptors to interrogate assembly configurations. Thus, applying relational machine learning techniques on relational data from large CAD assembly models provides an opportunity to rapidly identify similar components and geometric regions as well as to visualize duplicated, missing or inconsistent relationships. Currently,

the size and complexity of large assembly structures make their analysis time-consuming. For example, a full mechanical aero-engine model can contain 5000 solids, 0.3M faces and 1M edges. This data will represent millions of entities and known facts making tensor factorisation, such as the RESCAL algorithm proposed by Nickel et al.[3], an ideal candidate to experiment with.

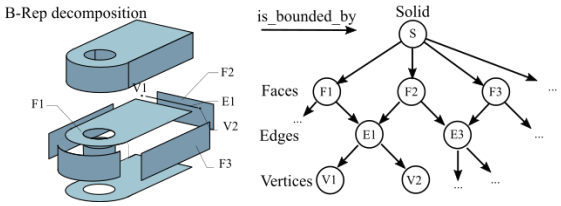


Figure 1. The topology of a Boundary Representation (B-Rep) Model can be represented as a graph.

This paper demonstrates the suitability of tensor factorization techniques as a relational learning approach for CAD assembly models. Our objective is to show how the scalability of these techniques can provide designers and analysts with rapid feedback on the configuration and consistency of the models they are generating or modifying.

Thus, it is illustrated how to learn efficiently from simple relational information available in CAD assembly models, e.g. adjacency between faces and solids, components' connectivity for tasks like:

- **Entity resolution:** (also known as object identification or instance matching). Here, the entity resolution task consists in identifying which entities (i.e. components, B-Rep faces and edges) in the CAD assembly refer to a similar input entity considering the B-Rep structure as well as the components connectivity within the assembly.
- **Link-based clustering:** entities (components, B-Rep faces and edges) are sorted into groups based on the similarity of their attributes and relationships. By iterating on the generated clusters, the user can create groups of similar components in order to filter the assembly. In addition, identical components which may not appear in the same cluster can expose assembly model inconsistencies where neighbouring relationships are not consistent between components.

Finally, by demonstrating how to efficiently look for similarity in the CAD models, the aim is to reduce the currently tedious and highly manual tasks when extracting and preparing simulation analysis models from a large assembly model.

2. RELATED WORK

2.1 3D shape retrieval

The approach in this paper is related to 3D shape retrieval methods. Literature on 3D model retrieval is vast and a comprehensive literature survey has be provided by Tangelder & Velkampt [7]. A brief overview on this topic is given here. Feature-based methods use shape descriptors to match similar shapes. For example, using spherical harmonics shape descriptors, the 3D search engine of Funkhouser et al. [8] retrieve similar objects from a database of 3D shapes. Robust and efficient feature-based methods convert the entire geometry model to global descriptors [7,9]. Kazhdan et al. proposed a rotation invariant descriptor based on spherical harmonics [10]. Graph-based methods evaluate the similarity between two 3D models by extracting a graph structure from the shape's geometry. The extracted graph links the components of the shape together. This graph, examples of which include a model graph [9], a reeb graph [11] or a skeleton graph [12,13], is used to compare shapes. Local model retrieval aims at finding similar subparts of different 3D models. For design reuse, Bai [9] extracts hierarchical descriptors from a CAD model feature tree to compare subparts of CAD models. Although this model graph method allows quick database queries, it relies on a CAD feature tree which is not unique for a given shape and is not always available [14]. To retrieve partial correspondences in CAD model, You and Tsai[15] define graphs corresponding to the B-Rep structure enriched with geometric attributes such as surface types, curve types and edge convexity. Although these 3D retrieval techniques could be applied to identify similar components in a CAD assembly, they do not consider the relationships between components within the assembly. Two components can share identical shapes but can be connected to completely different components in an assembly or have different connectivity configurations. Our objective is to identify similarities considering all available relationships in the CAD assembly.

2.2 CAD Assembly analysis

Assembly retrieval methods have been proposed to search and reuse complex mechanical assembly models. In [16], Chen et al. proposed a multilevel assembly descriptor to distinguish CAD assemblies. The descriptor is a combination of the hierarchical assembly structure, the interactions between components and global shape descriptors of components and sub-assemblies. A graph matching algorithm is then used to identify similar assemblies. To accelerate the matching process, an indexing mechanism is introduced. However, only the assembly level is considered (not the individual components' B-Rep structure). In addition, the hierarchical structure is not always available to an analyst for a large assembly [17] or not adapted to analysis requirements [18]. Hu et al. [19] uses a vector space model (mostly employed in document retrieval) for lightweight assembly retrieval. Although achieving interactive results and allowing partial matching, CAD components are transformed into simplified meshes (losing topological and geometrical information) and interactions between parts are not considered. Wang et al. [20] propose an assembly retrieval method efficiently comparing assembly models, represented as point sets using an Earth Mover's Distance-based matching method [21] to evaluate the dissimilarity between signatures, however, does not consider the relationships between parts in an assembly in the matching. The enriched assembly model EAM of Lupinetti et al. [22,23] contains (among other descriptors) patterns of repeated components as well as an interface layer

encoding the relationships between the different parts in an assembly model. EAMs, represented as graphs, are then compared by solving sub-graph isomorphism problems. The interface layer is also not considered when identifying the repeated components.

Regarding CAD assembly analysis, the tools available in CAD systems are mostly limited to clash/clearance analysis between components. Using Boolean operations, Shahwaan et al. [24] extracts and classifies interfaces between components to qualitatively identify the functional designation of components. This method considers the kinematic links between components to infer information, however, it requires a consistent model. Misaligned or missing components stop the application of inference rules. In addition, inferring on a large assembly is time consuming, making this approach difficult to use for quickly identifying inconsistencies. To overcome the scalability issue of Boolean operations in a CAD kernel, Jourdes et al. [25] has demonstrated how to efficiently extract assembly interfaces using a GPU ray casting approach. In the ontology-based approach of Vilmart et al. [18], information on repetitions of components and sub-assemblies serves as a basis to apply inference rules and deduce new assembly information, such as component designation. However, here too, components are grouped based on a one-to-one comparison using common shape descriptors such as symmetries. Due to the combinatory problem, such an approach is also difficult to scale on large assembly models. To the best of our knowledge, approaches to analyse the consistency of an assembly model which consider the relationships between components have not been proposed in literature.

2.3 Statistical Relational Learning

Statistical Relational Leaning (SRL) is concerned with domain models where entities are interconnected by multiple relations. In their review of Relation Machine Learning for Knowledge Graphs, Nickel & Murphy [1] classify SLR methods in two main classes: graph feature models and latent features models.

The first class captures the correlation using statistical models based on observable properties of the graph. For example, an unknown relation can be derived from the existence of a path in the graph. Initially, local similarity techniques were used to analyse the nearest neighbourhood of entities. In [26], Adamic & Adar proposed the frequency-weighted common neighbours index to identify similarity of entities by counting the common items between neighbours. As mentioned in [1], local similarity techniques scale well on large graphs, however, they are limited to single relationships where similarity is identified locally based on the direct neighbourhood. To consider that two entities can be similar without having common attributes, global similarity indices have been proposed. Among them, the Leicht-Holme-Newman index [27] analyses the ensemble of paths between entities. Although, the predictions are improved on graphs when relationships are non-local, these techniques might require more computation time [1,28]. For multi-relational knowledge graphs, where each edge is labelled to denote the type of relationship between the two vertices, Lao & Mitchell [29] propose a Path Ranking Algorithm to infer new beliefs in an imperfect knowledge base by predicting the probability of missing edges. Unlike latent feature models, this technique provides an easily interpretable model of the extracted features on the observable data. For further literature, Lu et al. [28] provides a survey on similarity indices used for link prediction, determining whether a particular relationship exists or retrieving relationships by their likelihood.

The second SLR class captures the correlation between the node/edges using latent features. Unlike graph-feature models which use features observed in the data, the latent features associated to entities have not been observed in the data, but are assumed to be hidden causes for the observables features [2]. Similar entities are derived from operations on these latent features. The objective of this SRL class is to infer the latent features automatically from the data. Tensor factorisation models have been proposed for learning from multi-relational knowledge graphs. Among them, Franz et al. [30] propose the TripleRank method to rank and produce richer description of linked data on the semantic web. The RESCAL factorisation model of Nickel et al. [2,3,31] has successfully demonstrated its ability to predict unknown triples on large knowledge bases consisting of millions of entities and known facts. RESCAL captures similarities of entities in the relation domain via interactions of the latent features. Jenatton et al [32] proposed a tensor factorisation model for highly multi-relational data, where the number of different relations is large. The reference paper of Kolda and Bader [33] provides an extensive review on tensor decomposition models and their applications.

Finally, as mentioned in [1], it is difficult to determine if a relational latent feature model or graph feature model is better for learning knowledge graphs. Authors in [1] agree that latent feature models are suited for data showing global relational patterns. Tensor factorization techniques are computationally efficient on a large database when relations can be explained using a small number of latent variables. Graph feature models are best suited when graphs patterns appear locally. For example, the Path Ranking Algorithm [29] is efficient when relationships can be explained from short paths in the graph.

In the context of this work, we propose to transcribe CAD models (described as a B-Rep model) as knowledge graphs to learn from. In this work, the RESCAL[3] tensor factorisation model is used as the learning model. This choice is based on the following assumptions:

- Similar entities (i.e. CAD components, B-Rep faces and edges) are not necessarily close in an adjacency graph. For example, standard components (bolts, nuts...) can appear in various locations in a mechanical structure. The path linking those components based on component adjacency might be long, thus making the graph feature model less efficient. On the other hand, the shared entity representation in RESCAL captures global dependencies due to the shared latent representations.
- One objective is to provide quick feedback to the designer/analyst on modelling errors, allowing quick design iteration to correct the CAD model. In [2], the authors demonstrated that the RESCAL model scales

linearly with the number of entities. Then, once the factorised model is computed, the user can quickly query if a specific relationship exists or not, essentially in real-time.

- Another objective is to automatically generate clusters of CAD components to help the user filter their assembly for a specific analysis. Depending on the input relationships selected by the user, the latent representations of entities can be used by clustering algorithms to automatically generate groups of components.

The novelty of this paper is the use of a latent-features method based on tensor factorisation to compare B-Rep entities (solid, faces and edges) in a CAD assembly. B-Rep entities and their connections within the assembly are extracted and stored in a large knowledge base to learn from. By capturing the interactions of the latent features, the tensor factorisation considers all the input relationships between the B-Rep entities. Entity similarities are derived from operations on these latent features, which are then used to detect design or assembly inconsistencies.

3. TENSOR FACTORIZATION OF CAD ENTITIES

3.1 Proposed pipeline

The method proposed in this work to analyse the consistency of CAD assembly models is decomposed into the following three main steps (refer to Figure 2).

Step 1. Pre-processing: extract data and fill the tensor

The first step extracts the data from the CAD assembly models. This data is used to populate a three-way tensor X of size n × n × m which will be factorised in Step 2. Similar to [3], the entries of the tensor X on the two first dimensions correspond to the combined entities to analyse. In this work, all the solids S, faces F and edges E of the B-Rep models B-Rep are defined as entities. Additional geometrical types (e.g. surface type, edge type), interfaces and convexity attributes are also entities.

The third dimension contains the m different types of relations between the entities. For each existing known relationship of a kth relation between the ith entity and jth entity, a tensor entry $x_{ijk} = 1$ is added to X. Otherwise, for non-existent relationships, the entry x_{ijk} is set to 0. Hence, for each kth relation, a frontal slice $X_k = \boldsymbol{X}_{:,:,k}$ is generated. This slice contains all the existing relationships (i.e. defined as semantic triples ith entity - kth predicate - jth entity in the RDF format) linking the entities through this kth relation.

In order to provide a first set of descriptors of the solid models, we propose to transfer the internal B-Rep structure into the slices of the tensor. As highlighted by You and Tsai [15], the benefit of using the B-Rep structure is its invariance to geometric transformation. Alignment of objects is not required before extracting the geometric signatures.

The following four main tensor slices are generated in this work:

- The first frontal slice X_0 of X contains the topological relations between solids, faces and edges of the B-Rep model. For example, if a topological relation "S_i - *is_bounded_by* - F_j" exists, an individual tensor entry $x_{ij0} = 1$ of X is generated linking the ith entity (solid S_i) to the jth entity (face F_j).

- The second slice contains the relations between the faces/edges and their geometrical and convexity type. For example, the fact that a face is planar is described by the relation "F_j - *is_type* - $GeomType_p$" translated by the entry $x_{jp1} = 1$ in the tensor slice X_1.

- The third slice contains the interface relationships between solids. If a solid S_i is touching/penetrating another solid S_l, or if there is a gap between the solids (smaller than a user-defined distance) the two relations "S_i – *has_interface* - I_o" and "S_l – *has_interface* - I_o" are added as new entry $x_{il2} = 1$ in the slice X_2. The relation between the interface and the interface type (interference, contact or gap) is also added.

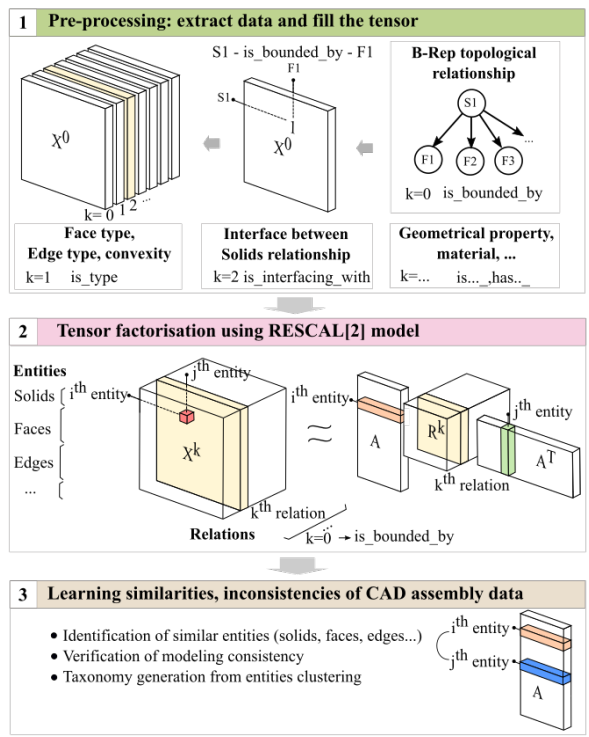


Figure 2 Overview of the proposed approach to learn similarities and inconsistencies in CAD assembly model using the RESCAL[3] tensor factorisation model.

The number of slices of the tensor model is not limited, new slices can be added (see discussions section 6). The objective of this paper is to demonstrate the applicability of the tensor factorisation on CAD assembly models. The current approach is limited to topological relationships and straightforward geometrical parameters which are directly available from the CAD system. Additional parameters generated by the user, such as material properties or

simulation attributes can be added in the model. Adding in-depth shape descriptors to better describe freeform surfaces or graph structures (e.g. medial object or reeb graph) is discussed in section 6 and left for future work.

Step 2. Tensor factorisation using RESCAL[3]

Given the tensor X of size n × n × m built in step 1, the RESCAL factorisation model proposed by Nickel et al. in [3] was used to compute a factorisation of X. RESCAL factorises each frontal slice X_k of X into the following matrix product:

$$X_k \approx AR_kA^T, \quad k \in 1..m$$

where A is a n × r factor matrix, R_k is r × r matrix which denotes the kth frontal slice of an adjacency R tensor. (RESCAL jointly factorises these adjacency matrices R_k, such that A is common for all frontal slices of X), r is a user-defined parameter defining the number of latent components (or common factors) in the matrix A. The matrix A can be viewed as an embedding of the entities in the r-dimensional latent space [2]. The matrices A and R_k are computed by solving a regularized minimisation problem as described in [3].

Step 3. Learning similarities, inconsistencies of CAD assembly data.

Given the factorization of the initial tensor X, the essential feature of the RESCAL method is that the latent space A reflects the similarity of entities in the relational domain [2]. Here the similarity of entities refers to the similarities of their relationships. For example: if two solids are bounded by the same type of faces, which are bounded by the same type of edges (slice X_0); if these two solids are also connected to other objects having similar topology, and so on…; there might be evidence that the two solids are identical within the assembly structure. Hence, two entities e_i and e_j can be compared by looking at their individual latent representations a_i and a_j in A. These latent representations not only measure the common attributes between the entities but also consider the similarity of related entities and relations involved in the relationships of the ith and jth entity.

The RESCAL model has been initially developed to perform relational learning tasks on large sets of relational data from the semantic web's Linked Open Data [31]. This paper illustrates how the approach can be used on large CAD assembly models to extract components or component entities in a range of scenarios. As described in Section 1, to demonstrate the applicability of tensor factorisation, the relational learning tasks of entity resolution and link-based clustering are performed. Section 4 develops these specific tasks and applies them to CAD assembly data.

4. USAGE SCENARIOS FOR CAD/CAE INTEGRATION

The objective of relational learning is to build a model of the domain from relational data that can be incomplete, noisy or even contain false information, thus avoiding the need for expensive user clean-up operations on CAD assembly data. From this relational model, specific learning tasks can be performed. In this section, the benefit of deriving a factorized model of a CAD assembly to analyse the consistency of the design will be described. The following usage scenarios are proposed in the context of CAD/CAE integration to help analysts understand and correct the input CAD models with a view to generating simulation models.

4.1. Entity resolution: retrieving similar entities in the CAD assembly

To simplify the approach, a CAD component is assumed to be modelled as a B-Rep solid. Following the data extraction of a CAD assembly (see Step 1 in Section 3.1) a tensor is generated containing the B-Rep solids, faces and edges. The tensor is then factorised (see Step 2 in Section 3.1) and produces a matrix A reflecting the relational similarity of entities.

The scenario is then to input an entity: a solid, a face or an edge and to identify which entities are similar to it in the CAD assembly. As mentioned in Section 3.1, similarity refers to the data contained in all relations in the initial tensor. As the topology relations of the CAD components are input in the tensor, together with the connection between the components, two CAD components (i.e. solids) are similar not only because they share a similar topological graph, but also because their similarity is propagated through their connections with the other components in the assembly which also have the same topological graph.

Hence, to compare an input entity e_i, given by the user, to any other entity e_j, a ranking of all entities is computed using their latent representation: the vectors a_i and a_j corresponding the ith and jth row in A.

Following the approach of Nickel [2], to compare two entities, the function k is calculated as such:

$$k(a_i, a_j) = \exp\left(-\frac{\|a_i - a_j\|^2}{\delta}\right),$$

where δ is an additional user-given parameter to exponentially scale the similarity value.

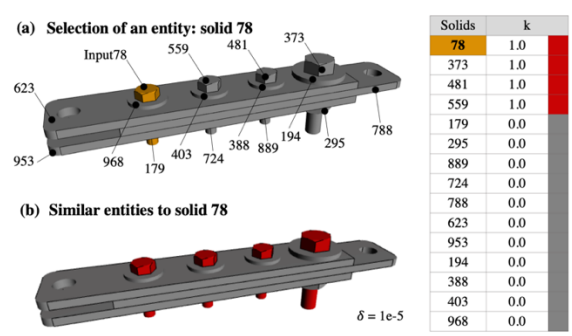


Figure 3 Retrieving similar solid entities. (a) The solid 78 is given as input and compared with all other solids in the assembly. (b) Solids are coloured based on their similarity value with solid 78. In Red, the solids 373, 481 and 559 are exactly similar to 78.

Figure 3 shows an example of the entity resolution on a simple model containing 15 solids. Solid 78 (one of the bolts) is given as an input and k is calculated for all the solids. For all the other bolts the same k value is returned. This result is expected as the bolts have the same topology, same face type, same edge type and convexity type. In addition, their faces have the same number of singularities. Most importantly, they are connected to the same components which are also identical. Their relations in the tensor X are symmetric which is reflected by having identical latent representations. Here, the data given as input in the tensor is not sufficient to distinguish the bolt 373 (largest bolt) being less similar than the same-size bolts 559 and 481. Indeed, depending on the designer/analyst need, additional shape characteristics should be added and evaluated.

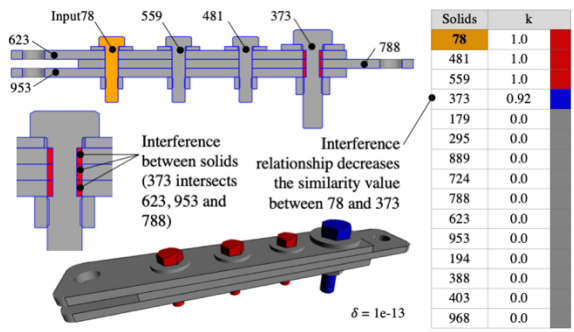


Figure 4 Example of the consideration of interference relationship between solids. The bolt 373 intersects the plates 623, 953 and 788. This configuration differs from the bolts 78, 559 and 481 having the same diameter as the plate holes (touching interface). 373 is less similar to 78 than 559/481 to 78 due to the consideration of the interference relationships.

In Figure 4 the bolted junction model is modified by changing the diameter of the holes in the plate bodies (623, 788 and 953) in order to introduce an interference between the bolt 373 and the plates (the diameter of bolt 373 being bigger than the holes diameter). These interferences modify the relationships of the bolt 373. Now, as shown in Figure 4, bolt 373 is still very similar to input 78 but slightly less similar than the bolts 373 and 481. This result illustrates how the RESCAL factorisation is able to consider all the relations as opposed to pairwise comparing entity to entity.

Once the factorisation is computed (see Section 0 for computation time), interrogating the A matrix is quick (<1s for an A matrix of size 175000 × n, corresponding to an assembly with 1000 solids, see Table 1 car engine model). Hence, the user can easily select a component in the assembly and ask which components are similar. By changing the δ parameter and adding a threshold for k (e.g. k>1e-10), the user can quickly filter the entities from the more similar to the less similar. Figure 5 illustrates entity resolution results for different values of the δ parameter. In the prototype implementation used to test the applicability of the proposed method, a slider component has been designed allowing the user to interactively discover similar objects for a given input. As shown in Figure 5(a), when setting δ to a low value, the method returns only the most similar bolt components. When increasing δ, more components are displayed until all components pass the threshold for k. Here, the scenario is to use this approach to quickly select similar components to keep/remove for the final simulation model. For example, in Figure 5(b), the 10 camshaft fingers (shaded component) are found distinctively even when δ is high (relative to this specific CAD model). This can be explained by the relatively distinct shape of the component within the assembly.

The input entities are not limited to solids but can also be faces or edges. On Figure 5(d) a planar face lying on the head of the bolt of the bolted junction model is selected. Similar faces are not only found on the same solid but also on all the similar bolts, showing that topological relations and components' interfaces are simultaneously considered in the factorisation.

4.2. Set of entity resolution: finding similar features

In this section, it will be illustrated how the entity resolution can be extended to sets of entities, still using the property of the A factor matrix in addition to the face adjacency graph from the B-Rep model. A usage scenario is to find similar features of a B-Rep solid, given one as an input. Here a feature refers to a set of B-Rep faces in a solid. Hence, the user inputs a set of adjacent faces and the algorithm returns a list of sets of faces considered similar. The following Algorithm 1 details the procedure. The returned list of features is ranked by summing k for each element of the feature (k has been previously calculated for each similar entity of each face of the given set of faces). Hence, the similar features can be visualized by the user in a similar manner to the entity resolution of Section 4.1.

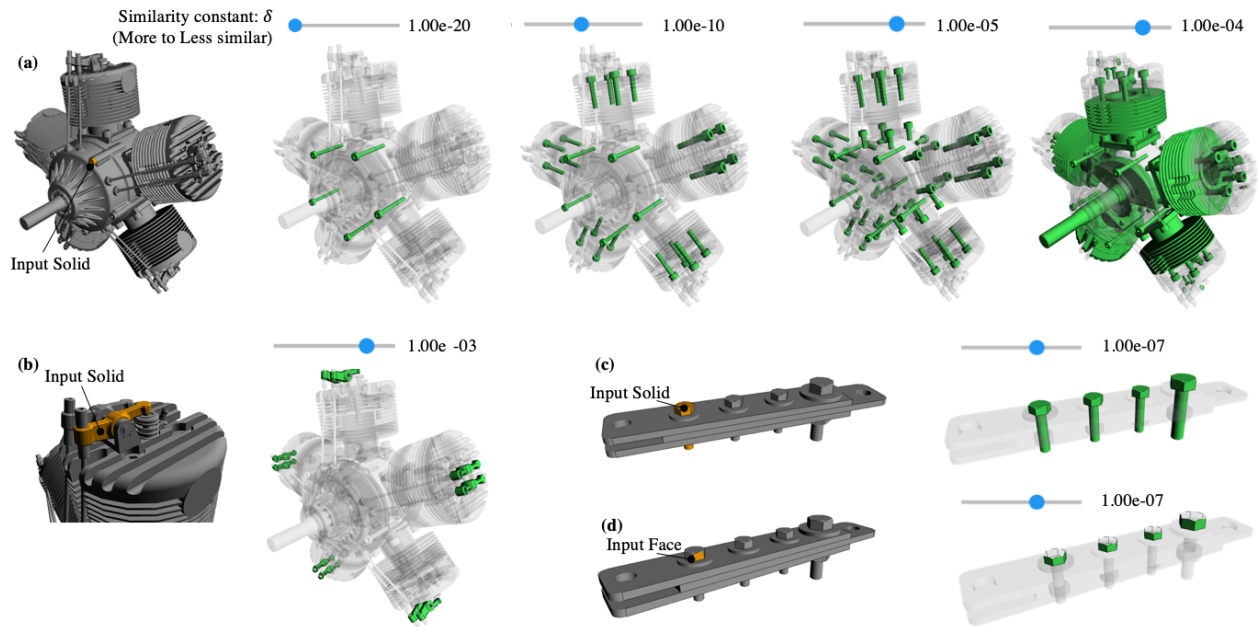


Figure 5 Examples of the entity resolution learning task. For a given solid or face entity, the user can modify the similarity constant in order to display more to less similar entities.

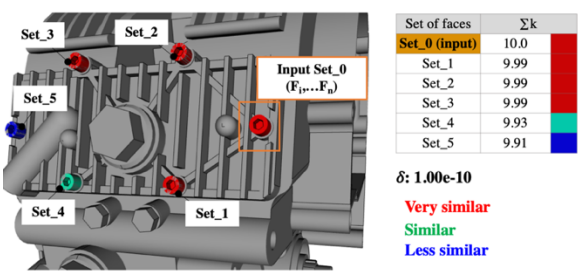


Figure 6 Entity resolution on set of faces. The user can visualize similar sets of faces on a B-Rep component given an input set of faces.

Figure 6 shows an example of entity resolution on sets of faces. The displayed model is a standalone B-Rep solid which has been created from the Boolean union of multiple individual components. The analysts' workflow is to remove small features, such as bolt heads considered as not relevant for the simulation they are to carry out. Given as an input the Set_0(F1,…Fn), shown highlighted in the box, the approach herein returns 5 similar features for a given δ. Similar to Section 4.1, the user can vary the value of δ to have more or less similar features returned. The processing time, however, is higher as more operations are performed compared to Section 4.1. For the example shown in Figure 6, Set_4 and Set_5 are ranked as "less similar" than 1, 2 and 3. Although, the latent factors in A cannot be interpreted directly, the difference can be explained by looking at the surrounding faces of the features in the model. Here Set 4 and 5 are connected to faces which are less similar to the faces surrounding the input Set 0 than the faces connected to Set 1, 2 and 3.

Algorithm 1 IdentifySimilarFeatures

Input: *fea /*given set of adjacent faces*

Function AddSimilarNeighbours (*Lfea* , *Rfea*)

/ add element of Rfea (ranked list of a similar) to the list of similar features*

for each *feature* **do**

add *element* of *Rfea* to *feature* if *element* shares a common edge with *feature*.

add *k* value of *element* to *kfea /* kfea is the sum of k of each element in the feature*

Result: *Lfea /* list of a similar features*

Initialize Lfea with the list of similar faces of the first face in fea

for each face in fea **do**

Algorithm 1 Identification of similar features given an input set of adjacent face.

4.3 Link-based clustering: building a taxonomy

In [31], authors have shown the capability of RESCAL to perform link-based clustering on large Linked Open Data databases containing millions of entities and known facts. As mentioned in Section 1, the objective is to partition entities into groups based on the similarities of relationships. In this section, a hierarchical clustering algorithm is used to cluster the entities using their latent representation in A. In this latent-component space, each row of A defines a vector of dimension r. This set of vectors is given as input to a

hierarchical clustering algorithm. In this work, the algorithm provided by the SciPy Python library [34] is used.

Figure 7 illustrates the result of the hierarchical clustering for the bolted junction model. The corresponding dendogram chart is shown. Here a threshold distance set to 0.05 returns 4 clusters corresponding to the 4 types of solids present in the model. By changing the threshold distance value, the user can visualize different cluster configurations.

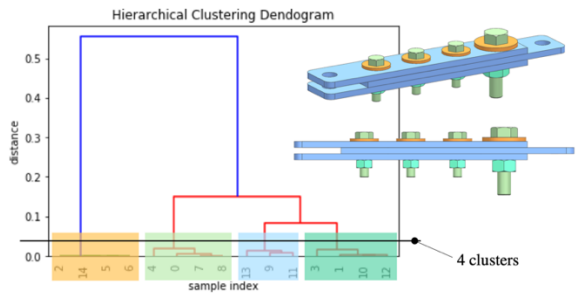
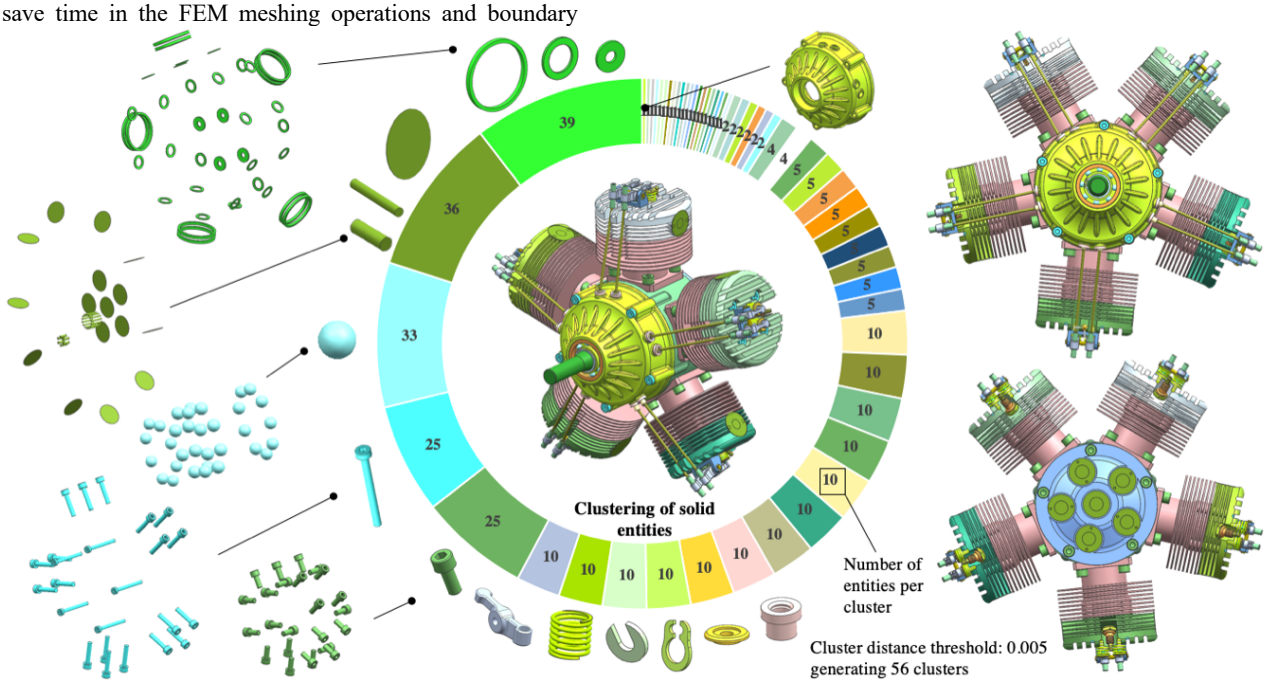


Figure 7 Hierarchical clustering of the bolted junction model. By setting a threshold, the user can quickly iterate different clusters configurations.

In a CAD/CAE context, the identification of similar components and similar regions of space is meaningful to save time in the FEM meshing operations and boundary condition application. Although a CAD assembly tree may exist, as explained by Vilmart [18], its structure might not be adapted for simulation purposes. Indeed, depending on the design protocol and user decisions, similar components can appear in multiple branches of the assembly tree. They are not part of the same instance. A typical example is the assembly of standard junctions. When multiple suppliers are involved in the assembly design, they will use their own junction models which will appear as different sub-assemblies in the final model. As mentioned in [31], an application of the link-clustering is the automatic generation of taxonomies. In the context of this work, under the supervision of the user, a CAD assembly model can be analysed and similar entities (components, faces, or edges) can be grouped. Figure 8 shows the result of the clustering for the radial engine model [35] (containing 374 solids) when the threshold is set to 5e-3. The largest clusters contain standard components which are repeated multiple times in the assembly. In this model generating these clusters helps an analyst to easily filter the assembly components. For assembly meshing requiring conformal meshes, when the components' interfaces imprints are also given as input, the user will need only to mesh one instance for each cluster and copy it to the groups of similar entities.


39
36
33
25
25
10
10
10

4.4 Identification of modelling inconsistencies

A practical application of our approach concerns the identification of modelling inconsistencies in a CAD assembly. Indeed, when displaying the generated clusters, some entities might not appear in the expected cluster. Figure 9 shows examples of inconsistencies found in CAD assemblies when analysing the clusters. In Figure 9(a), two bolts (in orange) are not identified as being part of the bolt cluster. This is due to the connectivity with neighboring components which is not consistent compared to the other bolts in the component. Their positionings generate interferences with the casings changing their relationships in the global tensor. In Figure 9(b), two groups of components having the same shape are connected differently to the casing. Here too, this configuration generates different relationships for the two groups of components. Although, clash management is usually handled by automatic scripts in CAD software, it obeys strict rules conventionally defined by the user. Here, this approach could complement the clash detection analysis as it doesn't follow any predefined rules and could be apply as a visual quality check tool.

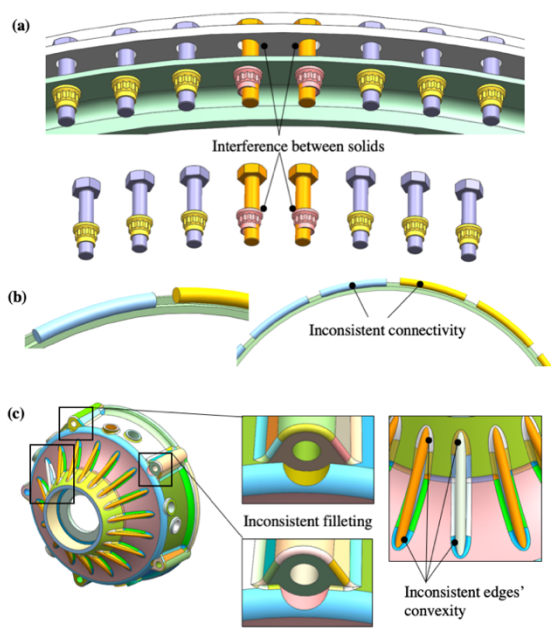


Figure 9 Example of modelling inconsistencies found following the link-based clustering task.

In Figure 9(c), the clustering is applied on face entities and displayed (one colour represents one cluster of faces). The symmetries of the model are not totally reflected in the colouring applied to the faces. An inconsistent filleting modifies the object's topology and an inconsistent edge convexity type modifies the convexity relationships of the cyclic stiffeners. These inconsistencies break the cyclic symmetry property. By using a relational learning approach, the user can visualize the similarities between the input entities and then identify the inconsistencies which need to be corrected. Having a consistent CAD assembly is highly valuable for simulation and manufacturing. For example, a finite element simulation of an assembly requires all the contacts between components to be defined. Running a process on a large assembly to automatically apply these contacts requires a clean input model, where all the components are correctly positioned. This approach provides a low-cost method to help the designer identify modelling issues upfront, without having to interrogate an entire assembly model component by component.

Figure 10 shows another example of modelling inconsistencies found when trying to identify similar features (see Section 4.2). Initially, a bolt head is given as the input set of faces. Choosing a small value for δ to display the most similar features, one of the bolt heads is not listed (see BoltHead_8 in Figure 10(a)). Increasing δ makes BoltHead_8 appear with a similarity k value significantly different from the other bolt heads. Looking closely at the CAD model for BoltHead_8, it appears that the bolt is intersecting the casing component before the Boolean union. This intersection results in BoltHead_8 having a slightly different topology from the other bolt heads, which is reflected in the latent factors of A.

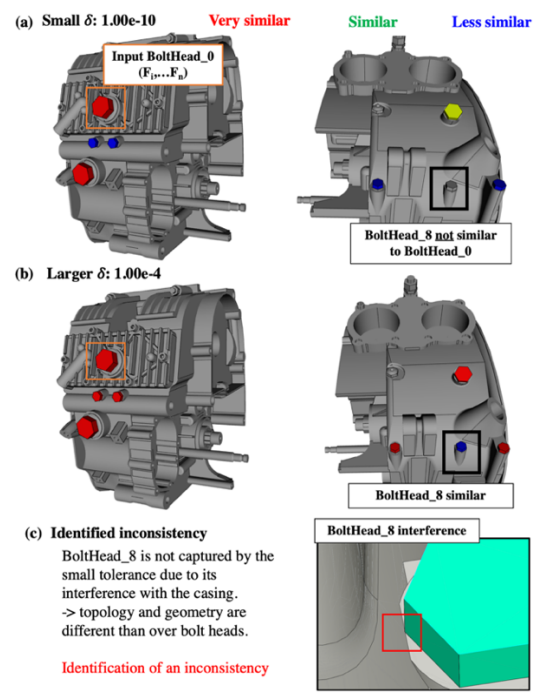


Figure 10 Inconsistency in the modelling of Bold Heads. Increasing the scale value of the similarity analysis reveals that the similarity of BoltHead_8 to BoltHead_0 is higher than the others bolt heads. (a) Entity resolution with small δ, (b) same entity resolution with a larger δ, (c) BoltHead_8 is intersecting the casing resulting in different geometry and topology.

Identifying inconsistent geometrical regions of a CAD component is relevant for simulation. It saves time for an analyst to filter the regions to simplify or remove before meshing for finite element modelling. In this paper, the

identification of similarities and the link-based clustering can be considered as an interactive tool for designer/analyst to verify the consistency of the CAD models. Ideally, processes could be developed to automate this identification.

5. EVALUATION ON LARGE CAD ASSEMBLIES

5.1 Scalability of the tensor factorisation

In [3], the authors designed the RESCAL factorisation model to be able to scale to large knowledge bases consisting of millions of entities, hundreds of relations and billions of known facts. In the context of this work, scalability is important as analysts tend to simulate increasingly larger assembly models containing thousands of components [17]. Even the design of standalone components can become highly complex in industry. For example, an intercase component in an aero-engine model contains around 10,000 CAD faces and 25,000 edges. It is essential to consider the running time of an analysis tool on such a large model.

RESCAL has been designed to scale linearly with the number of entities (see [2] giving details on the computational complexity). In this section, the scalability of the model is evaluated by running the proposed approach on various CAD assembly models. In this work, RESCAL was integrated with PythonOCC [36] (a Python version of the open-source geometric kernel OpenCascade [37]). The extraction of topological data and the interface detection is performed using the Siemens NX 11 API [38] and the Parasolid [39] library. Similar data extraction can be performed in other CAD systems.

The factorisation and clustering was carried out on a windows workstation with 4 Intel i7-6700 3.40 GHz CPUs, 32 GB RAM. Table 3 presents the computation time for different CAD models shown in Figure 16. The models in Figure 8 and Figure 16(a,e,f) are provided by the GrabCAD [35] community. These examples were used to verify the applicability of the proposed approach and its scalability to large models. As expected, the timing of the factorisation is proportional to the number of entities. For the largest model, the car engine, the factorisation time is less than 1 minute with a *r,* the number of latent components (or common factors) in the matrix *A,* equals to 100.

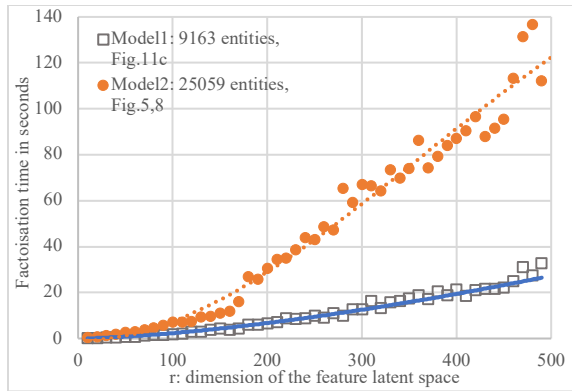


Figure 11 Evolution of the dimension of the feature latent space on the computation time to factorise the tensor.

As mentioned in [2], the tensor factorization computational complexity grows cubically with *r*. **Error! Reference source not found.** illustrates this evolution on two use-cases for different value of *r*. Depending on the size of the model, choosing this number can have an influence on the total computation time. The number of latent features *r* to consider is difficult to evaluate. Although in-depth evaluation has not be done in this work, our tests on the use-cases of Figure 16 using *r* equals to 100 allowed to generate clusters of expected similar shapes. Our observations suggest using a low rank (e.g. between 20 and 50) for small assembly (<200 solids) to avoid overfitting the model and a larger rank for larger assemblies to include more features and avoid underfitting. Once the factorization is computer, the clustering operation is almost instantaneous, making it possible to test different cluster variants. Figure 16 shows the cluster variant corresponding to δ = 1e-5 for each assembly model.

5.2 Comparison to graph-based and spherical harmonics-based methods

The tensor factorization approach enables the user to quickly interrogate the factorisation matrix when performing an entity resolution task. In this section we compared our approach for this task to two similar methods from the literature: the spherical harmonics approach of Kazhdan et al. [10] and the graph matching approach similar to [15] and [22,23]. For this evaluation, our use-case is the Vesta model: a large industrial CAD assembly provided by Rolls-Royce Plc and containing 6503 solid entities (see Figure 12 (b)). In Table 1, the runtime of the entity resolution task has been recorded for the three methods on five different components. For the spherical harmonics method, we use the executable of Kazhdan[40] with 17 spherical functions having 32 spherical harmonics. For the graph matching method, we extract the B-Rep topological face-edge graph of each solid and enrich it with the same geometrical information as the tensor slices, see Section 3.1, i.e. surface/edge type and edge convexity. To perform the sub-graph isomorphism, we use the VF2 algorithm implemented in the python library NetworkX[41].

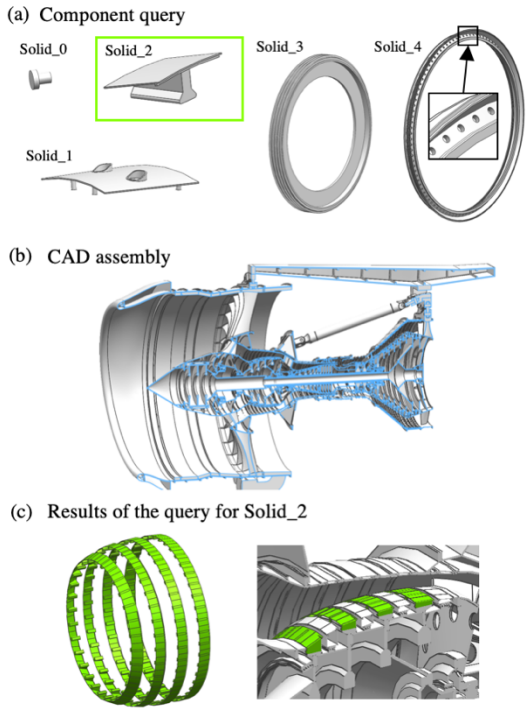


Figure 12 Entity resolution evaluation task on large assembly (Vesta model, courtesy of Rolls-Royce Plc). (a) query components, (b) the CAD assembly with 6503 solids, (c) example of a query using the tensor factorization-based method.

Table 1 Entity resolution timing comparison for the 5 components of Figure 12.

Method	**Global descriptor: Spherical Harmonics [10]**	**Graph matching: Subgraph Isomorphism**	**Relational Learning: Tensor factorisation**
B-Rep stats	Nb Solids: 6503, Nb Faces: 128 638, Nb Edges: 319 500		
CAD data extraction	3mins 01s		
Interface detection	10mins 15s		
Specific Data preparation task	Shape descriptors computation	Graph generation.	Tensor Factorisation (rank: 100)
	1min 42s	0.77s	2mins 18s
Entity Resolution task			
Solid_0 5 faces, 4 edges	0.14s	0.06s	0.07s
Solid_1 20 faces, 28 edges	0.12s	3.11s	0.07s
Solid_2 19 faces, 48 edges	0.14s	15mins 43s	0.07s
Solid_3 36 faces, 36 edges	0.11s	12mins 36s	0.07s
Solid_4 141 faces, 241 edges	0.12s	2mins 18s	0.07s

As shown in Table 1, the spherical harmonics approach requires a specific data preparation task to calculate the descriptor. Similarly, our approach requires to compute the factorisation of the tensor. The graph matching method can directly operate on the B-Rep graph, however, depending on the sub-graph configuration of the query component to match in the large assembly graph, the entity resolution task can be time consuming. This is particularly true when the combinatory is high, such as for the repeated components Solid_2 and Solid_3 corresponding to the numerous blades and seals present in the assembly. In this case, the matching algorithm has to look for a medium-size sub-graph pattern repeated several times into a large graph. The other two approaches, on the other hand, can directly compare the entities by looking at their factorised/shape descriptor vectors. Table 2 summarises the pros and cons of the three methods.

Table 2 Advantages and disadvantages of the evaluated methods

Method	**Pros**	**Cons**
Global descriptor: Spherical Harmonics [10]	Efficient to identify components having globally similar shapes. Query components can be stored in a database. Not sensitive to CAD topological variation.	Do not detect subpart (feature) in shape. Do not consider CAD topological segmentation. Do not consider interfaces between components.
Graph matching: Subgraph Isomorphism	Can match sub-shapes from a database to a new model.	Require an initial segmentation. Sensitive to CAD topological variation. Sensitive to the combinatory of different entities.
Relational Learning: Tensor factorisation (our approach)	Efficient to interactively compare entities within the same model. Able to analyse all relationships, considering interfaces between components. Quick query.	Require an initial segmentation. Cannot compare subpart stored in a database. Sensitive to CAD topological variation

To evaluate the clustering task performance, we built up a ground truth model of the Vesta assembly containing annotated components to compare with the clustering results obtained from our approach. Figure 13 (a-c) illustrates this model. 6218 solids out of 6503 have been grouped into 148 classes to represent a model an analyst would usually start with when setting up a FEM simulation on such large model. Indeed, an intuitive approach is to group similar repeated components along the aeroengine rotation axis. We then computed two cluster performance metrics: the adjusted rand index and the completeness scores using the scikit-learn python library[42]. The adjusted rand index measures from 0 to 1 (1 been the perfect score) the similarity between two clusters. The completeness metrics, measuring from 0 to 1 (1 been the perfect score), are divided into a homogeneity score (measuring how much each cluster contains only members of a single class) and a completeness score (measuring how much all members of a given class are assigned to the same cluster). The V-measure is the harmonic mean of the homogeneity and completeness scores[43].

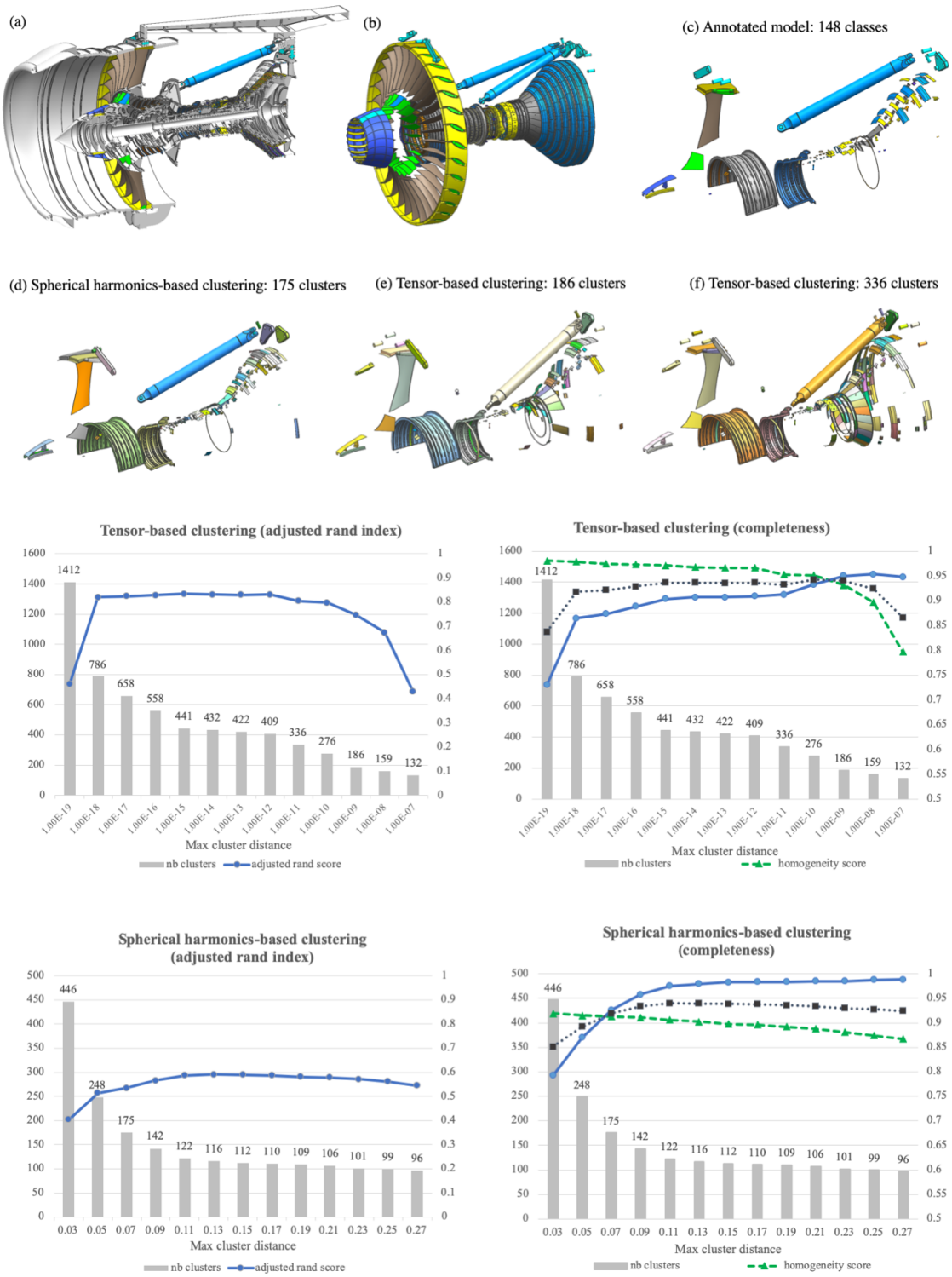


Figure 13 Evaluation of the clustering task on the Vesta model (courtesy of Rolls-Royce Plc) using the adjusted rand score and completeness score metrics: (a) the initial annotated model containing 6218 annotated solids (in color) of 6503; (b) display of the annotated solid; (c) Ground truth model containing 148 classes (one displayed component corresponds to one class), (d) clustering result based on spherical harmonics descriptors [10], (e-f) clustering result using our approach.

As shown in Figure 13, our approach scores over 0.7 for the adjusted rand index and over 0.9 for the V-measure. However, it never reaches the ground truth classes configuration. This can be explained by the fact that our approach does not consider the position of components along the axis of rotation. Indeed, the ground truth model separates blades and vanes components depending on their axial stage position in the aeroengine. This information is not given as an input to the tensor. However, by considering the interfaces between components, it starts separating the blade/vanes components into the compressors and turbines areas. This separation cannot be reached when using the spherical harmonics approach as it compares components individually. As discussed in the following section, future work is dedicated to improving these metrics by incorporating additional information in the initial tensor.

6. DISCUSSION AND FUTURE WORK

In this work, the advantages of the tensor factorization approach to learn similarities and inconsistencies of CAD assembly data are shown to be:

The factorisation model considers all relationships in the model and use this global information when creating groups of components. This is compared to the initial assembly tree structure where the components are grouped independently from their connectivity within the assembly.

The computation time scales linearly with the number of entities avoiding any combinatory scaling issues (see Table 1). The user can interactively explore similarities in the model, given the low computation time for clustering the matrix A.

Additional relations can be extracted from the data and added as slices in the initial tensor.

The approach relies on the boundary decompositions of the CAD models. The comparison of B-Rep shapes should be independent of the modeling process and topological constraints of geometric modelers [14,18]. As shown in Figure 14, a given shape can have different boundary decompositions. This may be due to the difference between the underlying architectures of different CAD/CAE packages, where periodic faces, such as cylindrical faces, will be represented with or without edges at the periodic seams. To make our approach less CAD modeler dependent, the virtual topology operators of Tierney [44] were used to merge faces and edges having the same underlying geometry. This is similar to the maximal topology concept of Vilmart et al. [18].

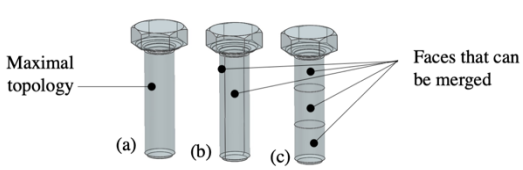


Figure 14 Examples of different boundary decomposition for the same shape. (a) The maximal topology (all faces and edges having same underlying geometry are merged). (b) Topology convention where cylinders are divided into half cylinders, (c) Particular segmentation to represent the bolt's thread.

One limitation of the proposed approach is that the factorisation model considers all relations as equal [3]. A relation containing a large number of relationships can influence the impact of an under-represented relation. Future research will look to improve the factorisation model to introduce weights to the relations while maintaining the scalability performance. This weighting will allow the user to adapt the influence of the relations for a particular application and provide a mechanism to automatically control and normalise the similarity threshold for different applications.

Another limitation of the approach is that it is challenging to make δ and k meaningful from an engineering perspective and therefore, depending on the shape descriptors given as input, determining the best threshold may necessitate multiple user iterations. However, these iterations are nearly instantaneous and finding a combination that works could be achieved through trial and error approaches.

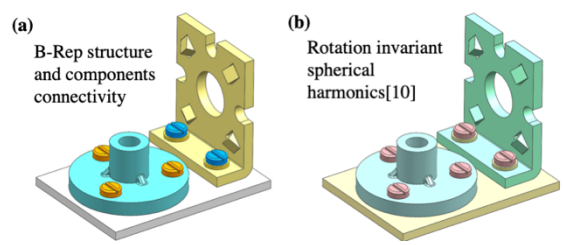


Figure 15 Cluster components (a) based on the B-Rep structure and the interfaces between components and (b) based on shape descriptors [10]

Extracting additional shape descriptors to enhance the identification of similarities is future work. Currently the topological and geometrical information from the B-Rep structure is used as shape descriptor. However, the current extracted information (see section 3.1) cannot capture all the characteristics of complex CAD models and is sensitive to small topological changes. For example, freeform surfaces will need additional descriptors to the ones proposed in this paper. Clearly, additional descriptors are needed to enhance the similarities. Because the factorisation model can deal with a large number of relationships, the challenges are to determine the correct set of shape descriptors to distinctively identify entities and to manage the conflict between different descriptors. Figure 15 shows the results of clustering components based on the current approach considering the connectivity between components (Figure 15a) and based on rotation invariant shape descriptor using spherical harmonics[10] calculated on each solid (Figure 15b). It can be seen that the interfaces between components does not impact the similarity identification and therefore the screws in Figure 15 (a) that were grouped into two similarity groups have been reduced to one group in Figure 15 (b). The shape descriptors using spherical harmonics are computationally efficient to identify similar shapes independently (see Table

1). Shape descriptors are usually described by discrete values, e.g. a grid of spherical harmonic coefficients for different radii of spherical functions[10]. This information could be integrated directly into the tensor as new entities or attributes (attributes can be added to the factorisation using RESCAL-ALS[2], an extended version of RESCAL). For example, an attribute will be added when two solids have the same coefficient for a particular harmonic of a particular spherical function. However, this direct integration results in a large number of additional relationships. For example, the hydraulic jack model would require 35422 new attributes or entities compared to the 255 interfaces between components. This number 35422 corresponds to the use of 17 spherical functions with 32 spherical harmonics. A tolerance of 1e-4 is used to consider two harmonics coefficients as equal. Decreasing this tolerance reduces the number of new relationships but decreases the similarity threshold between shapes. These new relationships globally influence the factorisation, i.e. the connectivity between components relationships could not be significant enough compared to the large number of shape descriptors relations. Integrating new descriptors is challenging and should be considered together with the introduction of weighting factors or to down-select relationships based on their overall contribution. This down-selection may vary for different applications, for example for feature removal applications certain relationships may be ignored that directly relate to feature insignificant for a particular analysis.

Graph models such as Reeb graphs[11] or skeletons [12,13] can be added to the initial tensor to better describe the internal structure of objects without adding a large number of attributes. Here the computation time to extract these should be investigated to not penalise the current efficient computing time for building the input to the tensor method. Indeed, although adding new shape descriptors will improve the similarities of entities, it is important to consider the scalability of the extraction algorithm for large assembly models. Nevertheless, the data extraction process does not require the intervention of the user and is independent from the factorisation. It can be performed once. The output data can be stored and updated for any modified components only.

Non-geometric information from the PLM system such as material properties can also be added as attributes to the factorisation.

Finally, the proposed approach using the proposed shape characteristics is efficient and scalable on B-Rep models. Tessellated models or FE results can also be an entry point for simulation. There is also an opportunity to apply a similar approach, given the right set of shape descriptors, to detect inconsistencies on such discrete models.

The work presented in this paper offers a number of potential benefits for meshing workflows:

1. Geometry preparation: Identifying inconsistencies in assemblies to aid downstream conformal meshing. The grouping of similar components enables geometry clean-up operations and boundary condition applications to be applied to only one component in the group and propagated to the remaining components, significantly reducing the amount of tedious manual operations.
2. Mesh generation: One instance in a group of similar components can be meshed with this mesh transformed to all instances. This enables identical meshes to be assigned to repeated components for certain analyses.
3. Post-processing: Knowing the relationships between components, sub-assemblies and assemblies will make it easier to visual, interpret and utilize results for very large assembly models, where subsets of space can be easily accessed by the used.

7. CONCLUSIONS

The preparation of fit-for-purpose CAD models for the efficient simulation of assemblies can be time consuming for the analyst. In this paper, the applicability of relational learning by tensor factorisation is shown to help engineers analyse the consistency of CAD assembly models. From this work, the following conclusions can be drawn:

- The RESCAL tensor model [3] is capable of generating a latent space reflecting the similarity of entities in the relational domain.
- The latent space is used to perform entity resolution on CAD assemblies: i.e. identifying which CAD entities (B-Rep solids, faces and edges) are identical to a given input entity. Entity resolution is extended to group of faces through an algorithm taking advantages of the latent features and face adjacency.
- Clustering of entities is performed in the latent space in order to generate groups of similar entities used to filter the components of interest for the analysis. The clusters can also be analysed by the user to identify inconsistencies between components, or to filter components considered as non-pertinent for a specific application.
- The proposed CAD model analysis approach can scale to large assembly models due to the scalability property of RESCAL. The user can quickly iterate a CAD assembly and test different cluster variants.

. ACKNOWLEDGEMENTS

The authors wish to acknowledge the financial support provided by the European Commission via the MUMPS project (704557), a H2020-MSCA-IF-2015 founding scheme and also the support provided by Innovate UK via GEMinIDS (project 113088), a UK Centre for Aerodynamics project. The authors acknowledge Rolls-Royce Plc for granting permission to publish this paper and for providing the Vesta aeroengine use-case.

Table 3 Statistics of the examples used in this paper. The RESCAL factorisation is performed with 10 iterations and a rank *r* of 100. *Shape descriptors[10] are here for timing comparison, they are not used in the factorisation

Model	B-Rep			Interfaces	Tensor entities	CAD data extraction	Interface detection	Shape described*[10]	RESCAL factorisation	Hierarchical Clustering	Clusters δ = 1e-5
	Solids	Faces	Edges								
Fused engine, *Fig.6, 10, 11a*	1	1636	4312	N/A	5949	3s	N/A	N/A	1.19s	N/A	N/A
Hydraulic pump, *Fig.11b*	71	1725	3603	180	5579	3s	3s	5s	1.25s	0.004s	26
Hydraulic jack, *Fig. 11c*	126	2620	6129	255	9130	4s	6s	8s	3.05s	0.002	54
Stapler, *Fig. 11d*	122	5697	14385	396	20600	7s	2s	7s	7.37s	0.004s	58
Radial engine, *Fig. 5, 8*	374	7963	16722	992	26051	10s	23s	27s	7.56s	0.014s	56
3D printer, *Fig. 11e*	878	32674	71778	3442	108772	55s	28s	42s	31s	0.06s	119
Car engine, *Fig. 11f*	997	49369	125448	2857	178671	1 min 42s	2mins 05s	1min 17s	47s	0.09s	142

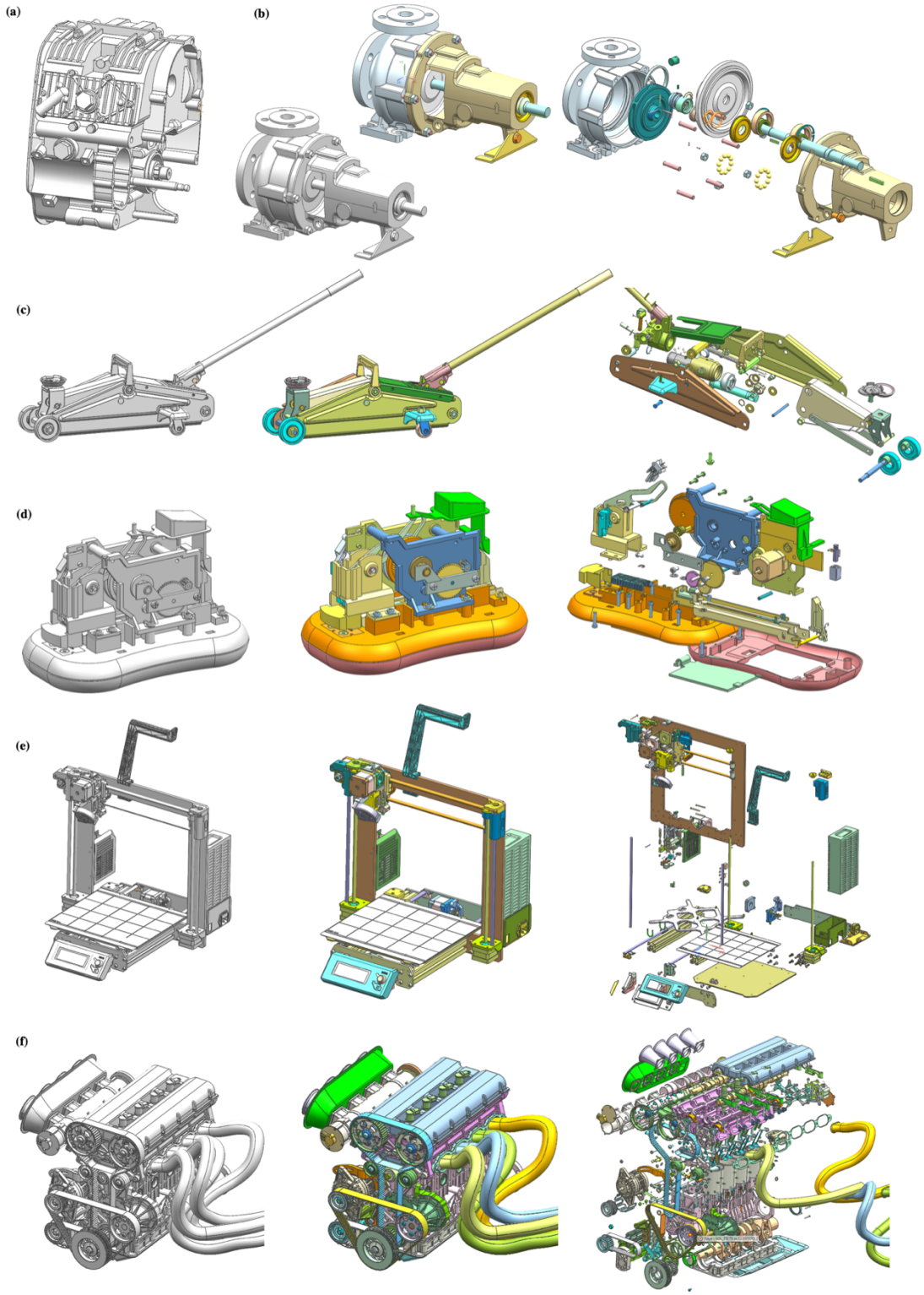


Figure 16 Results of the link-based clustering task on CAD assembly models. Left column: Initial models imported from Step files. Middle column: cluster variant with $\delta = 1e^{-5}$, one colour corresponds to one cluster. Right column: exploded view of the components' clusters.

REFERENCES

[1] M. Nickel, K. Murphy, V. Tresp, E. Gabrilovich, A review of relational machine learning for knowledge graphs, Proc. IEEE. 104 (2016) 11–33.

[2] M. Nickel, Tensor factorization for relational learning, Ludwig-Maximilians-Universität München, n.d.

[3] M. Nickel, V. Tresp, H.-P. Kriegel, A Three-Way Model for Collective Learning on Multi-Relational Data., in: ICML, 2011: pp. 809–816.

[4] C. Bizer, T. Heath, T. Berners-Lee, Linked data: The story so far, in: Semant. Serv. Interoperability Web Appl. Emerg. Concepts, IGI Global, 2011: pp. 205–227.

[5] W3C, Linked data, (n.d.). https://www.w3.org/standards/semanticweb/data (accessed February 1, 2019).

[6] W3C Working Group, Resource Description Framework Schema 1.1, (2014).

[7] J.W.H. Tangelder, R.C. Veltkamp, A survey of content based 3D shape retrieval methods, in: Shape Model. Appl. 2004. Proc., IEEE, 2004: pp. 145–156.

[8] T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, D. Dobkin, D. Jacobs, A search engine for 3D models, ACM Trans. Graph. 22 (2003) 83–105.

[9] J. Bai, S. Gao, W. Tang, Y. Liu, S. Guo, Design reuse oriented partial retrieval of CAD models, Comput. Des. 42 (2010) 1069–1084.

[10] M. Kazhdan, T. Funkhouser, S. Rusinkiewicz, Rotation invariant spherical harmonic representation of 3 d shape descriptors, in: Symp. Geom. Process., 2003: pp. 156–164.

[11] D. Bespalov, W.C. Regli, A. Shokoufandeh, Reeb graph based shape retrieval for CAD, in: ASME 2003 Int. Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf., American Society of Mechanical Engineers, 2003: pp. 229–238.

[12] H. Sundar, D. Silver, N. Gagvani, S. Dickinson, Skeleton based shape matching and retrieval, in: Shape Model. Int. 2003, IEEE, 2003: pp. 130–139.

[13] A. Tagliasacchi, T. Delame, M. Spagnuolo, N. Amenta, A. Telea, 3d skeletons: A state-of-the-art report, in: Comput. Graph. Forum, Wiley Online Library, 2016: pp. 573–597.

[14] F. Boussuge, J.-C. Léon, S. Hahmann, L. Fine, Extraction of generative processes from B-Rep shapes and application to idealization transformations, Comput. Des. 46 (2014) 79–89.

[15] C.-F. You, Y.-L. Tsai, 3D solid model retrieval for engineering reuse based on local feature correspondence, Int. J. Adv. Manuf. Technol. 46 (2010) 649–661.

[16] X. Chen, S. Gao, S. Guo, J. Bai, A flexible assembly retrieval approach for model reuse, Comput. Des. 44 (2012) 554–574.

[17] F. Boussuge, J.-C. Léon, S. Hahmann, L. Fine, An analysis of DMU transformation requirements for structural assembly simulations, in: Int. Conf. ECT2012, Proc. Eighth Int. Conf. Eng. Comput. Technol. Dubrovnik, Croat. 4-7 Sept. 2012, 2012.

[18] H. Vilmart, J.-C. Léon, F. Ulliana, From CAD Assemblies toward Knowledge-based Assemblies using an Intrinsic Knowledge-based Assembly Model, in: Proc. CAD'17, 2017: pp. 374–378.

[19] K.-M. Hu, B. Wang, J.-H. Yong, J.-C. Paul, Relaxed lightweight assembly retrieval using vector space model, Comput. Des. 45 (2013) 739–750.

[20] P. Wang, Y. Li, J. Zhang, J. Yu, An assembly retrieval approach based on shape distributions and Earth Mover's Distance, Int. J. Adv. Manuf. Technol. 86 (2016) 2635–2651.

[21] Y. Rubner, C. Tomasi, L.J. Guibas, The earth mover's distance as a metric for image retrieval, Int. J. Comput. Vis. 40 (2000) 99–121.

[22] K. Lupinetti, F. Giannini, M. Monti, J.-P. Pernot, Multi-criteria retrieval of CAD assembly models, J. Comput. Des. Eng. 5 (2018) 41–53.

[23] K. Lupinetti, F. Giannini, M. Monti, J.-P. Pernot, Automatic extraction of assembly component relationships for assembly model retrieval, Procedia CIRP. 50 (2016) 472–477.

[24] A. Shahwan, J.-C. Léon, G. Foucault, M. Trlin, O. Palombi, Qualitative behavioral reasoning from components' interfaces to components' functions for DMU adaption to FE analyses, Comput. Des. 45 (2013) 383–394.

[25] F. Jourdes, G.-P. Bonneau, S. Hahmann, J.-C. Léon, F. Faure, Computation of components' interfaces in highly complex assemblies, Comput. Des. 46 (2014) 170–178.

[26] L.A. Adamic, E. Adar, Friends and neighbors on the web, Soc. Networks. 25 (2003) 211–230.

[27] E.A. Leicht, P. Holme, M.E.J. Newman, Vertex similarity in networks, Phys. Rev. E. 73 (2006) 26120.

[28] L. Lü, T. Zhou, Link prediction in complex networks: A survey, Phys. A Stat. Mech. Its Appl. 390 (2011) 1150–1170.

[29] N. Lao, T. Mitchell, W.W. Cohen, Random walk inference and learning in a large scale knowledge base, in: Proc. Conf. Empir. Methods Nat. Lang. Process., Association for Computational Linguistics, 2011: pp. 529–539.

[30] T. Franz, A. Schultz, S. Sizov, S. Staab, Triplerank: Ranking semantic web data by tensor decomposition, in: Int. Semant. Web Conf., Springer, 2009: pp. 213–228.

[31] M. Nickel, V. Tresp, H.-P. Kriegel, Factorizing yago: scalable machine learning for linked data, in: Proc. 21st Int. Conf. World Wide Web, ACM, 2012: pp. 271–280.

[32] R. Jenatton, N.L. Roux, A. Bordes, G.R. Obozinski, A latent factor model for highly multi-relational data, in: Adv. Neural Inf. Process. Syst., 2012: pp. 3167–3175.

[33] T.G. Kolda, B.W. Bader, Tensor decompositions and applications, SIAM Rev. 51 (2009) 455–500.

[34] SciPy.org, SciPy - hierarchical clustering, (2018). https://docs.scipy.org/doc/scipy/reference/cluster.hie

rarchy.html (accessed February 1, 2019).
[35] Stratasys, GrabCAD, (2018). https://grabcad.com/library/radial-engine-163; https://grabcad.com/library/prusa-i3-mk3-solidworks-1%0D, https://grabcad.com/library/engine-2-0-liter-4-cylinder-88mm-bore-x-80mm-stroke-1%0D.
[36] T. Paviot, PythonOCC - 3D CAD for python, (2017). http://www.pythonocc.org/ (accessed February 1, 2019).
[37] OpenCascade - Opensource 3D CAD kernel, (2019). https://www.opencascade.com/ (accessed February 1, 2019).
[38] Siemens plm software, NX, (2018). http://www.plm.automation.siemens.com/en_us/products/nx/index.shtml (accessed June 5, 2017).
[39] Siemens plm software, Parasolid, (2018). https://www.plm.automation.siemens.com/en_us/products/open/parasolid/ (accessed June 5, 2017).
[40] Michael Kazhdan, Rotation Invariant Shape Descriptors, (n.d.). http://www.cs.jhu.edu/~misha/Code/ShapeSPH/ShapeDescriptor/ (accessed August 16, 2019).
[41] NetworkX - Isomorphism - VF2 algorithm, (2019). https://networkx.github.io/documentation/latest/reference/algorithms/isomorphism.vf2.html.
[42] Scikit-learn, Clustering evaluation, (2019). https://scikit-learn.org/stable/modules/clustering.html#clustering-performance-evaluation (accessed August 14, 2019).
[43] A. Rosenberg, J. Hirschberg, V-measure: A conditional entropy-based external cluster evaluation measure, in: Proc. 2007 Jt. Conf. Empir. Methods Nat. Lang. Process. Comput. Nat. Lang. Learn., 2007: pp. 410–420.
[44] C.M. Tierney, L. Sun, T.T. Robinson, C.G. Armstrong, Using virtual topology operations to generate analysis topology, Comput. Des. 85 (2017) 154–167. doi:https://doi.org/10.1016/j.cad.2016.07.015.