Extracting the Structure of Design Information From Collaborative Tagging

Information representation in engineering design is currently dominated by top–down approaches such as taxonomies and ontologies. While top–down approaches provide support for computational reasoning, they are primarily limited due to their static nature, limited scope, and developer-centric focus. Bottom–up approaches, such as folksonomies, are emerging as means to address the limitations of top–down approaches. Folksonomies refer to collaborative classification by users who freely assign tags to design information. They are dynamic in nature, broad in scope, and are user focused. However, they are limited due to the presence of ambiguities and redundancies in the tags used by different people. Considering their complementary nature, the ideal approach is to use both top–down and bottom-up approaches in a synergistic manner. To facilitate this synergy, the goal in this paper is to present techniques for using dynamic folksonomies to extract global characteristics of the structure of design information, and to create hierarchies of tags that can guide the development of structured taxonomies and ontologies. The approach presented in this paper involves using (a) tools such as degree distribution and K-neighborhood connectivity analysis to extract the global characteristics of folksonomies and (b) set-based technique and hierarchical clustering to develop a hierarchy of tags. The approach is illustrated using data from a collective innovation platform that supports collaborative tagging for design information. It is shown that despite the flat nature of the folksonomies insights about the hierarchy in information can be gained. The effects of various parameters on the tag hierarchy are discussed. The approach has potential to be used synergistically with top–down approaches such as ontologies to support the next generation collaborative design platforms. [DOI: 10.1115/1.3617447]

Keywords: folksonomies, tags, design classification, network analysis, hierarchical clustering, collective innovation

1 Frame of Reference—Classification in Engineering Design

1.1 Top–Down Knowledge Representation Using Taxonomies and Ontologies. Engineering design is a knowledge intensive activity [1]. Various techniques have been proposed in the engineering design literature to support knowledge representation, classification, search, and retrieval. Two of the popular approaches are taxonomies and ontologies. Taxonomies are hierarchically organized sets of concepts. Comprehensive design taxonomies have been developed for functions [2,3], components [4], and for mechanical design methods and theories [5]. Taxonomies classify information into hierarchical sets but are limited in their support for computational reasoning. With the increasing role of computers in the generation and communication of design information, ontologies are emerging as a dominant approach for knowledge representation in engineering design. An ontology is defined as an explicit specification of a shared conceptualization [6]. It consists of a set of concepts within a domain and the interrelationships between those concepts. It can be used to perform computational reasoning about information [7]. Ontologies have been used to support various aspects of design such as conceptual design [8–11], design optimization [12], and product family design [13]. Design taxonomies and ontologies have been used for developing online design repositories [14,15].

Ontologies and taxonomies are both top–down constructs because their development begins with the specification of a domain and the systematic identification of all the relevant concepts and relationships within the domain. Ontologies are engineered using a set of agreed-upon terms and logically founded constraints on their use [16]. Since ontologies are based on formal computational representations of knowledge, they are ideally suited for supporting communication and interoperability between computational design support systems. However, the limitations of such top–down constructs are that they are:

(a) static in nature,
(b) limited to the domain for which the ontology is developed,
(c) limited in supporting the evolutionary nature of the knowledge-base,
(d) generally developed by a single person or a focused team of experts and hence do not capture users’ viewpoints and
(e) suitable for capturing only a single way of organizing information.

Titus et al. [17] argue that a design representation scheme should be: (1) flexible, (2) network-like, (3) modifiable by the user, (4) allow users to access data from other users, and (5) mirror the emerging and evolving nature of design. In other words, a design representation scheme should also support the bottom–up nature of engineering design.
1.2 Bottom–up Classification Using Folksonomies. One such scheme that supports bottom–up classification of information is user-centric tagging. It is also known as a folksonomy, a word derived from the combination of “folk” and “taxonomy” indicating collaborative classification. Folksonomies are currently used in Web 2.0 applications for information search and retrieval. They have gained popularity during the past few years due to their use in self-organized information websites, photo-sharing websites, and collaborative tagging websites.3 Google 3D Warehouse uses collaborative tagging for classifying 3D models generated using the Google SketchUp tool. In these websites, the users freely tag objects such as web-pages, blogs, pictures, and videos using relevant keywords. These tags capture the users’ individual understanding of objects. The tags from all of the users are centrally managed and are available to all other users. The tags are used by users to find related content, people with similar interests and other related tags.

In contrast to the hierarchical nature of ontologies and taxonomies, folksonomies are flat. Tags are clustered as tag-clouds. Titus et al. [17] make a comparison between folksonomies and taxonomies. The folksonomies are bottom-up, flexible, user-defined, emergent, and quick and easy to use. In contrast, ontologies are top-down, predefined, developed by experts, static, and require significant domain knowledge to develop. Mathes [18] discusses the advantages and disadvantages of folksonomies. In addition to the advantages discussed above, folksonomies increase the chances of discovering new information and require less time, effort, and resources for development than an ontology or taxonomy. The disadvantages of folksonomies include inconsistencies due to ambiguity in the usage of words, incomplete tag assignment, redundancies due to the use of synonyms, low search precision, and poor resource navigation and retrieval [19].

1.3 Analyzing the Structures of Folksonomies. At a basic level, folksonomies can be used to identify related information objects by finding objects that share similar tags. In addition, folksonomies can also provide information about macrolevel organization of information within a domain. Researchers have used network-based techniques to extract properties of folksonomies. Cattuto et al. [20] use network measures including characteristic path length, clustering coefficient, and correlations in node connectivity to analyze the folksonomies in delicious.com and BibSonomy.4 Begelman et al. [21] and Simpson [22] utilize clustering algorithms for improving search and exploration using folksonomies. Van Damme et al. [23] present an approach to improve information retrieval from folksonomies by identifying appropriately annotated information. Goldar and Haberman [24] analyze the dynamic aspects of collaborative tagging.

Recognizing the top–down nature of ontologies, the bottom–up nature of folksonomies and the need for both top–down and bottom–up approaches; researchers have recently proposed the extraction of structured ontologies from free-form folksonomies [17]. Lin et al. [19] propose an approach for extracting ontological structures from folksonomies through data mining techniques with relevant terms from an existing ontology. Van Damme et al. [25] present a general approach for deriving ontologies by combining various techniques such as statistical analysis of folksonomies, semantic web, online lexical resources, ontology mapping, and matching. Liu and Guen [26] present an approach for the ontology evolution by the users, Chen and Qin [27] discuss an approach for deriving ontologies from folksonomies and controlled vocabularies.

All of the efforts discussed above are in the domain of web-based social networking. However, in the domain of engineering design, the use of folksonomies for classification has only been discussed at a conceptual level [17]. Quantitative analysis of folksonomies has not been addressed in the engineering design literature. In this paper, we present a step toward addressing this gap. Specifically, the focus is on using folksonomies to extract (1) global characteristics of the folksonomies and (2) hierarchies of tags to support the development of taxonomies and ontologies. The approach is illustrated using a function-based collective innovation (COIN) platform, discussed in Sec. 2. We use a network-based approach to extract the global characteristics of folksonomies (Sec. 3.1). The characteristics are extracted using a degree distribution, which provides information about the structure and evolution of a network (Sec. 3.2) and (b) k-neighborhood connectivity (KNC) analysis to understand the general nature of the hierarchy (Sec. 3.3). The hierarchy of tags is created through a combination of set-based approach (Sec. 4.1) and hierarchical clustering of tags and content nodes (Sec. 4.2). Finally, closing thoughts are presented in Sec. 5.

2 Overview of the COIN Platform

Collective innovation is defined as a connected, open, and collaborative process to generate, develop, prioritize, and execute new ideas [28]. The key characteristics that distinguish collective innovation from traditional product development processes are as follows:

(a) the participation of a large number of self-interested and autonomous individuals within and outside organizational boundaries,
(b) openness of borders to participation and knowledge,
(c) flat organizational structures, and
(d) bottom-up nature of product development.

Innovation approaches can be characterized based on two dimensions—participation and processes. Participation ranges from targeted groups to large crowds. Processes range from structured to self-organized. In well-structured processes, the alignment of product development activities and decisions is enforced by a hierarchical authority, whereas in self-organized processes, coordination emerges from the interactions between participants [29]. For example, open source software such as Linux [30] is developed through highly self-organized collective innovation processes. Open source software development is characterized by decentralized problem solving, self-selected participation, self-organizing coordination and collaboration, and free revealing of knowledge [29]. In contrast, crowdsourcing involves a structured competition for solving a well-defined problem. For example, Innocentive [31] provides a platform where organizations with specific tasks seek independent ideas from individuals in a broader community. Both open source development and crowdsourcing involve the participation of crowds. Open innovation refers to the use of inflow and outflow of knowledge across organizational boundaries to accelerate innovation [32]. The in-flow of knowledge can be from another organization or from a community of people, including expert networks, retired employees, customers, suppliers, or partners. All of these are examples of collective innovation processes.

In the domain of product development, systematic design approaches, involving strategically ordered successive, but also iterative, steps of information transformations have been shown to support designers to solve problems efficiently and effectively [33]. In the conceptual stages of design, function-based approaches have traditionally been very popular in engineering design [33]. As noted by Simon [34], discovering viable ways of decomposing a complex system into semi-independent parts corresponding to the system’s many functional parts is a powerful and widely adopted technique. In particular, solutions can be elaborated in a process that can be decomposed, communicated, and influenced collectively, while knowledge and relationships can be analyzed and synthesized in new ways. A function-based approach facilitates cooperation and collaboration in a collective innovation environment. Both the structured and self-organized
The process can either be carried out by targeted groups or by independent individuals constituting crowds.

The COIN platform is developed to utilize the value of traditional function-based systematic approaches in a collective environment. Through this platform a large number of participants can work together toward achieving greater innovation levels compared with focused teams that are limited by the knowledge and experience of the team members in traditional structured processes. In the COIN platform, the function-based systematic approach is instantiated through a sequence of mappings that can be carried out in a collective manner— from functions describing a problem to be solved, to principles embodying underlying solutions to accomplish specific functions, resulting in concepts representing emerging solutions to the initial problem. These mappings are facilitated through the creation of different types of nodes and relationships between them.

The primary node types supported in the COIN platform are as follows:

(a) Problems: open-ended obstacles that prevent a transformation from an undesirable initial state to a desirable goal state, usually characterized by complexity and uncertainty;

(b) Functions: solution-neutral descriptions of the functions that concepts fulfill to solve problems. Functions are modeled as verb–noun pairs representing energy/material/information input/output relationships by reference to purpose;

(c) Principles: quantitative descriptions using laws of physics and mathematics governing the quantities involved and/or means to embody such laws as physical structures;

(d) Concepts: ideas sufficiently developed to evaluate the functionality, principles, and associated structures for solving a problem;

(e) Methods: systematic procedures according to a detailed, logically ordered plan characteristic of a particular discipline or field of knowledge;

(f) Metrics: measures that are typically used to facilitate decision making;

(g) Projects: individual projects carried out by individual(s) from universities, research institutes, engineering/management/prototyping/industrialization service providers, etc.

Any user can instantiate any node type in any order. All nodes are created independent of each other, thereby allowing reusability of nodes for any problem. Nodes can be related to each other by specifying relationships between them. Descriptions, references, and mathematical/geometrical model attachments, etc. used to facilitate systematic mappings within the function-based systematic approach are tied to specific nodes. Users can freely assign tags (keywords) to each node. The users can either reuse the tags previously defined by others or create new tags for any node. These tags represent an alternate way of creating relationships between nodes. Nodes that share common tags are related to each other. Hence, tags help to create higher-level evolving relationships between the content nodes. A set of tags assigned to the nodes represent the bottom-up folksonomy.

The COIN platform is implemented using Drupal [35] which is an open-source modular framework and content management system that allows rapid development of community-driven websites. The Drupal core provides the basic functionality of user management, administration tools, blogs, etc. All other functionality is added using modules that can be downloaded separately from the Drupal website. For example, Drupal has a taxonomy module, which allows users to develop different taxonomies to organize content. The platform provides the flexibility for assigning different access rights to different team members. This feature allows the users to control the level of self-organization and structure in the design process. JOOMLA [36] or similar content management systems could also be used for the development of similar community-based websites.

So far, the general content has been populated from the following sources (1) a dissertation on integrated product and materials design at the Georgia Institute of Technology (Atlanta), (2) cooperative Bachelor thesis projects on populating concepts of different types of mechanical components, supported by Freudenberg INNOVATIONCENTER (Wenheime, Germany), (3) class assignments at Washington State University (Pullman) in a 60 student strong machine element course, and (4) related ongoing semester projects in about 40 student strong machine element and materials handling courses at the University of Applied Sciences Giessen-Friedberg (Friedberg, Germany). The diversity of design data populated through different sources makes it suitable for investigating the bottom-up folksonomy generated by collaborative tagging.

Ideally, top-down taxonomies and ontologies and bottom-up tagging are complementing in nature. On one hand, bottom-up tagging often overcomes limitations of top-down domain-specific taxonomies and ontologies, particularly when dealing with multi-domain evolving knowledge-bases in dynamic, self-sustained networks. On the other hand, top-down taxonomies may support specific users in organizing information hierarchically for targeted search or eliminate ambiguity in the usage of words or synonyms. In this paper, the bottom-up user-centric tagging of nodes in COIN platform is utilized as an illustrative example to establish emergent classifications.

3 Extracting Global Characteristics of a Folksonomy

3.1 Modeling Networks of Content Nodes and Tags

The structure of information on the COIN platform is modeled as an undirected unweighted bipartite graph [37] with two types of nodes—content nodes and tags. A bipartite graph G = (S1 ⊔ S2, E) consists of two disjoint sets of nodes S1 and S2 and a set of edges E such that each edge in E connects a node in S1 to a node in S2. An example of a bipartite graph is shown in Fig. 1, where S1 = {1, 2, 3, 4} and S2 = {a, b, c, d, e, f, g}. In the information network under consideration, S1 is the set of content nodes and S2 is the set of tags. The edges in the graph represent the assignment of tags to the content nodes. The bipartite graph can be represented as a matrix M with rows as nodes S1 and columns as nodes S2. Each entry Mij is a Boolean entry representing whether there is a link between nodes S1 and S2.

The bipartite graph G can be transformed into two weighted undirected graphs G1 = (S1, E1) and G2 = (S2, E2) consisting of content nodes (S1) and tags (S2), respectively. Figure 1 provides an illustration of graphs G1 and G2 derived from a bipartite graph.

Two content nodes in G1 are connected by an edge if both of them share at least one tag. Similarly, two tags in G2 are linked if they are assigned to at least one common content node. The weights associated with edges E1 represent the number of common tags shared by a set of content nodes. Similarly, the weights in graph G2 represent the number of common content nodes shared by two tags. The weights on edges in G1 and G2 represent the similarities between different types of nodes (content and tags, respectively).

These graphs can be represented using adjacency matrices with rows and columns corresponding to nodes and the matrix elements corresponding to the weights of edges between different nodes. The matrices are symmetric due to the undirected nature of the graphs. The matrices corresponding to G1 and G2 are represented as M1 and M2, respectively. The matrices corresponding to the illustrative graphs are shown in Fig. 1. M1 and M2 can be derived from the matrix M as follows:

\[(M_1)_{ij} = \sum_k M_{ik} M_{kj}\]

\[(M_2)_{ij} = \sum_k M_{kj} M_{ik}\]

For the COIN platform, M1 and M2 correspond to the weighted networks of content nodes and tags, respectively. The snapshot of
data from the COIN platform that is analyzed in this paper contains 1036 content nodes and 564 tags. Hence, the sizes of $M$, $M_1$ and $M_2$ are $[1036 \times 564]$, $[1036 \times 1036]$ and $[564 \times 564]$, respectively. Matrix $M$ is a sparse matrix with 4356 links corresponding to the tag assignments. In the Secs. 3.2 and 3.3 we analyze the properties of these three matrices to determine the structure of the information contained in COIN platform. Specifically, the following aspects are analyzed: degree distribution and k-neighborhood connectivity.

The starting point for the extraction of information from a folksonomy is the bipartite graph of tags and content nodes. The hierarchy of tags generated is solely dependent on (a) the bipartite graph and (b) the parameters chosen for hierarchical clustering and the set-based process. Insights for choosing the parameters are provided in Sec. 4. In addition to these parameters, the topology of the bipartite graph also greatly affects the resulting hierarchy. The effect of the topological characteristics of the bipartite graph on the overall hierarchy is an issue for future exploration. In this section, we discuss two tools to analyze the global characteristics of the bipartite matrix. These include degree distribution (Sec. 3.2) and K-neighborhood connectivity analysis (Sec. 3.3). Using these tools, we provide some additional insights on the use of folksonomies for design information, with pointers for further research.

3.2 Degree Distribution. In a graph, a node’s degree is defined as the number of other nodes linked to it [37]. A node’s degree is a microlevel (local) property. The degree distribution, on the other hand, is a global property of the graph, which provides insights about a graph’s topology and its growth mechanisms. Different topologies of networks have characteristic degree distributions. For example, a random network has binomial degree distribution which takes the form of a Poisson distribution for large number of vertices [38,39]. An exponential graph has an exponential degree distribution and a scale-free graph follows a power-law distribution [40,41]. Degree distribution has been used to study various complex networks such as the world wide web, Internet, power-line networks, citation networks, ecological networks, and protein networks [42].

The degree of a node in the bipartite graph also provides information about local clustering between different nodes. As shown in Fig. 1, if there is a node in $S_1$ that is connected to $n$ nodes in $S_2$, the resulting subgraph of $G_2$ with those $n$ nodes will be completely connected. Such a completely connected set of nodes is called a clique [43]. In other words, a node in $S_1$ with degree $n$ in the bipartite network results in a clique with $n$ nodes in graph $G_2$. For the network corresponding to COIN platform, a tag assigned to $n$ content nodes will result in a clique with $n$ nodes in the content network. Hence, tags act as means for clustering the content nodes, which in turn act as means for clustering the tags. The degree distribution serves as an indicator of the size of these clusters.

The degree distributions of the tags and content nodes for the bipartite graph $(G)$ of the COIN platform data are shown in Fig. 2. The degree distribution of tags is plotted on a log–log scale. Based on this plot, it is observed that a few tags have links with a large number of content nodes, while a majority of the tags have only few links with the content nodes. The degree distribution of the content nodes is shown in Fig. 2(b). Note that the degrees of all the content nodes are significantly lower than the tags, with an average of four tags per content node. This can be attributed to the tendency of the users to assign only a few tags to each content node.

![Fig. 1 An example of bipartite graphs and two weighted graphs derived from it](image)

![Fig. 2 Degree distribution of (a) tags and (b) content nodes in the bipartite graph](image)
The degree distribution of the tags also provides valuable insight about the topology of the network. The tags follow a degree distribution similar to a power-law distribution. As mentioned above, a power-law distribution is a characteristic of scale-free networks. On the other hand, the content nodes follow a degree distribution closer to a Poisson distribution, which characterizes random networks. The question that arises is: why does the degree distribution of tags follow that of a scale-free network, whereas the degree distribution of content follows that of a random network? One possible explanation for such a distribution is the inherent structure of the domain information. If the content nodes and tags are such that a clear distinction between higher domain-level information and content-specific information can be made, then tags corresponding to higher level information are assigned to a large number of content nodes, and the tags corresponding to content-specific information are assigned to smaller numbers of content nodes. This results in a distribution similar to Sec. 4. Tags that connect to a small number of content nodes describe a higher-level classification and can be termed as domain tags. Examples of such tags from the COIN platform include “functions,” “concept,” “design,” “market,” and “analysis-based methods.” These tags create large clusters of content nodes. These tags are also higher-level tags in the tag-hierarchy generated in the COIN platform, preferential attachment is a general principle that exists in most folksonomies. Further investigation of the evolution of the COIN folksonomy and other folksonomies from engineering design platforms is necessary to understand whether such a degree distribution is common to all platforms or is unique to the COIN platform.

### 3.3 K-neighborhood Connectivity (KNC) Analysis

In K-neighborhood connectivity (KNC) analysis [45], the content and tag graphs (G1 and G2, respectively) are transformed into new graphs by removing all the links with weights less than k. In the transformed content graph, two content nodes are connected (i.e., are k-neighbors) if they share at least k tags. Similarly, two tags are k-neighbors if both of them are assigned to at least k content nodes. In a KNC plot, the number of clusters and maximum cluster size are plotted as a function of k. The plot is used to study the degradation of connectivity as a function of k [45].

As k increases, weaker links between content nodes and tags are eliminated from the graphs. Hence, the graphs become progressively more fragmented with increasing k. The size of the largest components of the graphs decreases monotonically and the number of components increases monotonically. The rates of decrease in the size of the largest components and the growth of the number of fragments provide insights into the hierarchical organization of information: If the rate at which the maximum cluster size reduces is high, then a large number of nodes are linked by a few links at a higher level. On the other hand, if there is a gradual drop in the maximum cluster size, it implies that many nodes are strongly linked. The KNC plots for content and tags are shown in Fig. 3. In the case of COIN platform, the rate of decrease in maximum cluster size is similar for tags and content nodes (the rate of decrease being slightly greater for tags than for content nodes).

In Fig. 3(b), the number of components in k-neighborhood graphs with at least three nodes is plotted as a function of k. Note that if all the components of the k-neighborhood graph are considered, the plot is monotonically increasing because the individual unconnected nodes are also counted as a component. However, in this plot, we ignore the unconnected nodes and the components with only two nodes. The number of components with three or more nodes can increase, because larger components are decomposed into smaller components as links are eliminated with increasing k. The number of components can also decrease as k increases if the new components have only one or two nodes. In the case of the COIN platform, it is observed that for the tag network, the number of such components reduces monotonically. For lower values of k, the reduction is fast but for higher values of k, the reduction is gradual. From this trend, it can be inferred that

![Fig. 3 (a) Maximum size and (b) number of components of k-neighborhoods](image-url)
for lower values of \( k \), large numbers of components of the graph have smaller numbers of nodes. These small components are eliminated as \( k \) is increased. At the other end of the plot, the components corresponding to higher values of \( k \) have larger numbers of nodes. This can be attributed to the star-like topology of the tag network with some strongly connected core tags and a large number of weakly connected tags. In contrast, the numbers of clusters in the content network changes nonmonotonically as \( k \) increases. The plot shows a sharp peak at \( k = 4 \). The number of clusters is significantly higher for \( k = 4 \) as compared with \( k = 3 \) and \( k = 5 \). This implies that as we go from \( k = 3 \) to \( k = 4 \), the graph becomes suddenly fragmented by the decomposition of the large clusters into many smaller clusters. The network at \( k = 3 \) is held together by a number of links with a weight of 3. The subsequent sudden drop in the number of clusters at \( k = 5 \) indicates that many of the clusters found for \( k = 4 \) have only a few nodes. The value of \( k \) can be used to determine the hierarchy level, particularly when using the co-occurrence metric. A higher value of \( k \) corresponds to a higher value of hierarchy level, which is discussed in Sec. 4.

K-neighborhood analysis can also be used to compare the global characteristics of different folksonomies.

4 Extracting the Hierarchy of Information From Folksonomies

The approach adopted here for extracting the hierarchy of information from folksonomies involves a combination of set-based approach and hierarchical clustering. The set-based approach is used to organize the tags into a hierarchy, and hierarchical clustering is used to address some of the limitations of folksonomies such as ambiguity and redundancies in the use of tags. These two components of the approach are discussed in Secs. 4.1 and 4.2, respectively.

Fig. 4 Parts of the hierarchy of tags generated by the proposed approach

4.1 Set-Based Approach for Extracting the Hierarchy of Tags. The degree distribution and K-neighborhood analysis are primarily used to study the overall topology of the bipartite graph and the size of clusters. However, they do not answer the following question: “How can the information content be organized into a hierarchical structure?” Recently, there has been some progress in approaches for generating hierarchies of tags. Wu et al. [46] utilize a probablistic supervised method to allocate a set of tags into a set of parallel clusters. Zhou et al. [47] present a probabilistic unsupervised method using deterministic annealing to recursively split the set of tags into a hierarchy. Heymann and Garcia-Molina [48] build the tag hierarchy based on the centrality of tags in a tag-similarity graph. The tag-similarity graph is an unweighted graph in which two tags are connected if the similarity is above a threshold. A tag is added as a child of the most similar tag if its similarity to that tag is greater than some threshold. Lin et al. [19] integrate the application of data mining techniques to folksonomies with existing upper-level ontologies such as WORDNET.

In this paper, we build upon the set-based approach from Mika [16] and Barla and Bielikova [49] where the fundamental assumption behind hierarchical organization of tag is that “in an ideal situation, the tag \( t_0 \) is a parent of tag \( t_i \) if the set of entities (persons or items) classified under \( t_0 \) is a subset of the entities under \( t_i \)” [49].

For example, in the sample network shown in Fig. 1, where the set \( S_1 \) represents content nodes and the set \( S_2 \) represents tags, tag “d” is associated with all the content nodes. All the other tags are associated with a subset of the content nodes. Hence, tag d is a parent of all the other tags. This provides a set of parent–child relationships between tags that can be used to build the hierarchy.

A snapshot of the tag hierarchy for the COIN platform created using the fundamental assumption listed above is shown in Fig. 4. The hierarchy is created and stored as an XML file displayed in a tree form. At the topmost level, an artificial “root” tag is created.
which helps in creating a connected tree. It is observed that using
the fundamental assumption involving strict subsets, it is possible
to extract key elements of the hierarchy. For illustration, five sam-
ple tags (gear, tropism, chromium, functions, and use-functions)
are shown in the expanded form. Up to five levels of hierarchy are
observed.

While the hierarchy of some of the tags is well developed, it is
also observed that a large number of tags (not shown in the figure)
are children of the root node, giving rise to a flat hierarchy. This is
attributed to the free-form and collaborative nature of folksono-
maries where tag assignment is generally incomplete. For example,
the different types of gears (worm, helical, and spur) may not all
be assigned the tag “gear”. Hence, the fundamental assumption
of subsets is not always valid. To address this limitation, we use
near-perfect subsets [16]. The fundamental assumption is modi-
fied for the near-perfect subsets as follows: For a pair of tags \( t_a \)
and \( t_b \) associated with sets of content nodes \( C_a \) and \( C_b \), respec-
tively, tag \( t_b \) is a parent of \( t_a \) if

\[
\frac{n(C_a \cap C_b)}{n(C_a)} \geq P \quad \text{and} \quad \frac{n(C_a \cap C_b)}{n(C_b)} \leq Q
\]

where \( n(C) \) is the number of elements in set \( C \); \( p_{ab} \) represents the
fraction of \( C_a \), that overlaps with \( C_b \); \( P \in [0 \ 1] \) and \( Q \in [0 \ 1] \) are
threshold parameters. If \( C_b \) is a proper subset of \( C_a \) then \( p_{ab} = 1 \),
and if \( C_a = C_b \) then \( p_{ab} = p_{ba} = 1 \). Tag \( t_b \) is a parent of \( t_a \) if
\( p_{ab} \geq P \) and \( p_{ba} \leq Q \). Similarly, tag \( t_b \) is a parent of \( t_a \) if \( p_{ab} \geq P \)
and \( p_{ba} \leq Q \). There is a possibility that both these conditions are
satisfied simultaneously if \( P \leq Q \), and \( p_{ab} \) and \( p_{ba} \) are within the
range \([P \ Q] \). To avoid this situation, \( P \) is always chosen to be
greater than \( Q \).

The steps for creating hierarchy of tags are as follows:

1. The threshold parameters \( P \) and \( Q \) are chosen such that
\( P > Q \).
2. For each pair of tags \( t_a \) and \( t_b \), the sets of content nodes \( C_a \)
and \( C_b \), their intersection \( C_a \cap C_b \), and \( p_{ab} \) and \( p_{ba} \) are
determined using the bipartite matrix, \( M \).
3. A “root” node representing the top of the hierarchy is
created.
4. For each tag, the parent nodes and the children nodes are
determined using the modified assumption listed above. A
tree of all the tags is created.

The threshold parameters are primarily used to modify the
structure of the hierarchy. To study the effects of these parameters
on the structure of the hierarchy, the tag hierarchy is generated
using various combinations of \( P \) and \( Q \). The number of tags
directly below the \( \text{root} \) node, referred to as Level 1, and the num-
ber of tags in Level 3 are plotted in Figs. 5(a) and 5(b), respec-
tively. It is observed that the number of tags in Level 1 decreases
as \( P \) decreases and \( Q \) increases. On the other hand, the number of
tags in Level 3 increases as \( P \) decreases and \( Q \) increases. This can
be explained as follows. As \( P \) is reduced, the condition for a tag to
be a parent of another tag is relaxed. Similarly, the condition is
relaxed as \( Q \) is increased. In both cases, the tags which share sub-
sets (whose sizes are defined by the values of \( P \) and \( Q \)) of content
nodes have a higher chance of being related through a parent–
child relationship. An increase in the number of tags related by
parent–child relationships results in a taller hierarchy which has
fewer tags right below the root. Hence, reducing \( P \) or increasing \( Q \)
results in fewer tags at Level 1 and more tags at Level 3 (i.e.,
taller hierarchy). The number of levels in the hierarchy may also
increase.

While the hierarchy becomes taller by reducing \( P \) and increasing
\( Q \), \( P \) should be greater than \( Q \). Further, a very small \( P \)
increases the likelihood of tags at the same level being classified
as parent–child relationships, thereby diluting the hierarchical
structure. Hence, an appropriate balance between these thresholds
is required. We assert that the best values of \( P \) and \( Q \) are depend-
ent on the folksonomy under consideration. Currently, these values
are determined based on experimentation and experience. Choosing the best threshold values is an open issue requiring fur-
ther investigation.

As a summary of this section, the set-based approach with near-
perfect subsets is proposed for creating a hierarchy of tags. The
structure of the hierarchy can be varied using the two threshold
parameters. The assumption of near-perfectness addresses one of
the inherent limitations of folksonomies—incomplete tag assign-
ment. However, it does not address the limitations associated with
the presence of synonyms and ambiguity in the usage of words.
These limitations are addressed using hierarchical clustering.

### 4.2 Hierarchical Clustering to Refine the Hierarchy of Tags

Due to their inherent nature, folksonomies are associated
with many sources of noise such as the use of similar terms for
different objects and different terms for the same objects. Hier-
archical clustering is a data-mining technique commonly used for
reducing such noise and improving precision in statistical data.
Clustering involves assigning a set of objects into subsets (clus-
ters) such that the objects within a cluster are closer to each other
as compared with the objects in different clusters [50]. Hierarchi-
cal clustering involves recursive clustering using previously
assigned clusters [51]. At the highest level of the hierarchy, all
objects are within one cluster. At the lowest level, each object is
its own cluster and the number of clusters is equal to the number
of objects. Between the highest and lowest levels, various inter-
mediate levels of clusters are generated based on the similarity or
distance between different objects. Hierarchical clustering is used
in statistical data analysis, pattern recognition, and data mining
applications.

Hierarchical clustering algorithms can be divided into agglom-
erative (bottom-up) and divisive (top-down) algorithms. Agglom-
erative clustering algorithms involve starting with nodes as
individual clusters, which are progressively merged into larger
clusters whereas divisive algorithms start with a single cluster
consisting of all nodes that are sequentially split into smaller
clusters.
Hierarchical clustering generates a set of clusters at different levels ranging from the top-most level where all nodes are within a cluster to the bottom level where all nodes are individual clusters. For example, one set of clusters found in the content graph is shown in Fig. 7. The cluster shown in the figure is a subgraph of the content network \( G_1 \) with link-weights greater than or equal to three. Hence, any two linked nodes in the figure share at least three tags. The clusters correspond to different types of entities including ball bearings, welds, fasteners, gears, couplings, design methods, functions, etc. Other sets of clusters can be identified by changing the level in the hierarchy. It is important to note that these hierarchical clusters are based on relatedness and may represent synonyms or any other type of relatedness such as parent-child. Hence, this hierarchy of clusters is fundamentally different from the hierarchy of tags developed in Sec. 4.1. The key challenge is to develop the right sets of clusters and to interpret them to refine the hierarchy of tags.

As discussed by Gemmell et al. [54], following parameters need to be chosen appropriately in order to achieve the desired behavior from hierarchical clustering: (a) similarity measure and (b) hierarchy level. The similarity measures are used to progressively generate the clusters. The hierarchy level refers to the level of the cluster hierarchy at which the clusters are observed. At the bottom level (hierarchy level = 0), all nodes are separate clusters and at the top level (hierarchy level = 1), all nodes are under one cluster. The effect of these parameters is discussed in the Secs. 4.2.1 and 4.2.2.

4.2.1 Similarity Measures. Cattuto et al. [53] compare tag co-occurrence, cosine similarity of co-occurrence distributions, and FolkRank and conclude that different metrics are suitable for different goals. The authors suggest that cosine similarity is suitable for extracting synonyms and co-occurrence is more suitable for extracting taxonomical relationships. To evaluate the effect of

Hierarchical clustering generates a set of clusters at different levels ranging from the top-most level where all nodes are within a cluster to the bottom level where all nodes are individual clusters. For example, one set of clusters found in the content graph is shown in Fig. 7. The cluster shown in the figure is a subgraph of the content network \( G_1 \) with link-weights greater than or equal to three. Hence, any two linked nodes in the figure share at least three tags. The clusters correspond to different types of entities including ball bearings, welds, fasteners, gears, couplings, design methods, functions, etc. Other sets of clusters can be identified by changing the level in the hierarchy. It is important to note that these hierarchical clusters are based on relatedness and may represent synonyms or any other type of relatedness such as parent-child. Hence, this hierarchy of clusters is fundamentally different from the hierarchy of tags developed in Sec. 4.1. The key challenge is to develop the right sets of clusters and to interpret them to refine the hierarchy of tags.

As discussed by Gemmell et al. [54], following parameters need to be chosen appropriately in order to achieve the desired behavior from hierarchical clustering: (a) similarity measure and (b) hierarchy level. The similarity measures are used to progressively generate the clusters. The hierarchy level refers to the level of the cluster hierarchy at which the clusters are observed. At the bottom level (hierarchy level = 0), all nodes are separate clusters and at the top level (hierarchy level = 1), all nodes are under one cluster. The effect of these parameters is discussed in the Secs. 4.2.1 and 4.2.2.

4.2.1 Similarity Measures. Cattuto et al. [53] compare tag co-occurrence, cosine similarity of co-occurrence distributions, and FolkRank and conclude that different metrics are suitable for different goals. The authors suggest that cosine similarity is suitable for extracting synonyms and co-occurrence is more suitable for extracting taxonomical relationships. To evaluate the effect of
similarity measures for the COIN folksonomy, we compare the clusters of tags generated using cosine similarity and co-occurrence measure for a fixed hierarchy level of 0.5. We observe that using cosine similarity, various synonymous tags including {'half moon key,' "woodruff key"}; {'mitred half lap," "mitred"'}, {'butts joints," "butts"'} are merged into clusters much earlier in the agglomerative clustering process. Hence, these synonyms are considered as one tag in the tag hierarchy developed using the set-based process, discussed in Sec. 4.1. Using co-occurrence measure, on the other hand, these synonymous tags are different tags during the corresponding steps of the agglomerative clustering process. For example, both "half moon key" and "woodruff key" are children of the parent tag "Key". Hence, our observation is consistent with that of Cattuto and co-authors that cosine similarity is better for extracting synonyms than co-occurrence measure.

It is important to note that while cosine similarity has a higher likelihood of extracting synonyms its effect is not solely on clustering synonyms. Just like co-occurrence measure, it also clusters related tags and concepts. For example, all bearing types are clustered as one node, and all brazing types are clustered as one node using both the similarity measures. We also observe that both the similarity measures failed to cluster tags with typographical errors. For example, "Intuition-Based Methods" and "Intuition-Based Methods" were treated as separate nodes. To identify such typographical errors, other techniques such as the use of existing lexical databases (e.g., WORDNET) are required.

Finally, the type of similarity metric chosen for concepts does not have a significant impact on the hierarchy of tags. Since the focus is on determining the hierarchy of tags only, we use the simple co-occurrence metric for clustering concepts. Hence, we use cosine similarity for extracting synonymous tags and co-occurrence measure for clustering concepts. Clustered tags are treated as a single tag and clustered concepts are treated as a single concept. The corresponding bipartite matrix, M, is updated by combining the clustered tags and nodes. The procedure listed in Sec. 4.1 is then used to create the tag hierarchy.

4.2.2 Hierarchy Level. The hierarchy level determines the number and sizes of clusters. As a cluster is combined into a single node, the resulting number of nodes reduces as the hierarchy level is increased from 0 to 1. The effect of the hierarchy level on the number of resulting nodes and clusters is shown in Fig. 8. In the figure, a comparison between cosine similarity and co-occurrence measure is also shown. As expected, the number of tags and concepts reduces with the increase in the hierarchy level. However, the rate of reduction is different for cosine and co-occurrence measures. Cosine similarity results in a greater reduction of the number of nodes than the co-occurrence measure. In other words, more nodes are combined into clusters using cosine similarity.

Fig. 9 Percentage of tags at Level 1 for different hierarchy levels of tags and content nodes

To ensure that the clustering of synonymous tags was a characteristic of the cosine similarity and not a result of the clustering of greater number of tags, we compared two different hierarchy levels for the two similarity measures that resulted in the same number of tags. The results were similar. The synonyms listed in Sec. 4.2.1 were still clustered using cosine similarity but not clustered using co-occurrence.

The hierarchy level also affects the final tag hierarchy generated using the set-based approach. The percentage of tags at level 1 (right below the root node) in the final hierarchy for different levels of clustering hierarchy is plotted in Fig. 9. It is observed that as the number of tags combined into clusters increases with increasing hierarchical level, the percentage of tags in Level 1 increases. Hence, hierarchy increasingly becomes flatter. For example, at a hierarchy level of 0.2, all bearings are combined into one tag, all luminescence types are combined into one tag, etc. Hence, the hierarchy tends toward flatter structures. The hierarchy level of content nodes has negligible impact on the cluster of tags between values 0.0 and 0.8. As the hierarchy level of content nodes becomes close to 1, the tag hierarchy becomes flatter. At the extreme value of 1.0, all the content nodes are combined into one node and the tag hierarchy becomes totally flat (i.e., all the tags are direct children of the root node).

Hence, as long as the hierarchy of concepts is sufficiently less than the maximum value, its effect on the tag hierarchy can be ignored. The hierarchy level of the tags can be adjusted to get the desired hierarchy. One possible strategy is to carry out the process iteratively. Instead of picking a single value of the hierarchy level, the process can be initiated with a higher hierarchy value to get a flatter tag hierarchy. From that flat tag hierarchy, progressively reduced hierarchy levels can be chosen for subsets of tags to refine parts of the tag hierarchy. This could potentially improve the precision and also make the process computationally efficient. An investigation into this iterative process is a topic for future investigation.

5 Closing Comments

In this paper, we show that despite the flat nature of folksonomies, taxonomic relationships between tags can be learnt. We present an approach for extracting a hierarchy of tags from folksonomies. In this approach, content nodes and tags are modeled as bipartite graphs, from which individual content and tag networks are derived and analyzed. Network characteristics such as degree distribution and KNC plots are used to derive specific insights about the information. Through a combination of set-based approaches and hierarchical clustering, parent–child relationships between tags are determined and a hierarchy is created. The approach involves choosing two threshold parameters for near-
perfect subsets and two parameters associated with hierarchical clustering, namely, similarity measure and hierarchy level. The effect of each parameter on the resulting tag hierarchy is discussed in the paper. The focus is on the approach and the COIN framework is primarily used as an illustrative example.

It is emphasized that our objective in this paper is not to present an automatic, unsupervised approach for the extraction of hierarchy. Rather, the approach requires careful analysis and appropriate choice of parameters. We believe that every folksonomy is different in terms of the types of users and the information content. Hence, a detailed exploration of the effect of different parameters is presented so that it can be customized for different folksonomies. In addition to the future research avenues discussed throughout the paper, the proposed approach could be extended by pursuing the following directions:

1. Dynamic aspects: One of the key features of folksonomies is their evolutionary nature. The focus in this paper is on the structure of the folksonomy at a given point in time. Future research would involve studying the structure of the networks at different points in time. The study of the dynamic nature of networks would provide insight into how the domain information grows and how the information structure evolves in a bottom–up manner. Further research is required in different network topologies and their impact on network properties such as degree distribution and clustering coefficients. The relationship between degree distribution of nodes and domain interaction hierarchy needs further exploration.

2. Integration of top–down and bottom–up approaches: Future research on the use of folksonomies in engineering design will involve exploring the synergistic use of folksonomies with top–down approaches (taxonomies and ontologies). An important step in combining the benefits of top–down and bottom–up approaches is the investigation of the use of existing ontologies to scaffold the evolution of folksonomies. The use of bottom–up classification is also well-suited for design approaches that map knowledge across multiple diverse domains.

3. Community-related aspects: The user community is an important aspect of the development of folksonomies. In this paper, we only discuss a bipartite model with content nodes and tags. But a tripartite model which includes people who annotate with tags can also be used to enhance the model. For example, Mika [16] extends the traditional bipartite model of ontologies (concepts and relations) by incorporating actors in the model. The inclusion of people in the data is also helpful from various perspectives. In addition to the extraction of information structure, folksonomies can also be used to detect community structures [55]. Folksonomies can be used for personalizing navigation through tag clustering [54]. Also, consideration of the types of users (novice-experts) can greatly impact the folksonomies. For example, as discussed in Sec. 3.2, there is a tendency of the users of COIN platform to assign about four tags per content nodes. It is unclear whether this is a characteristic of the current user community or whether it could be rectified by using experienced engineers to populate the system. Folksonomies can also be used for capturing the evolution of shared representation as the community evolves [56,57].

4. Commonalities: Recently, many evolutionary systems have been found to display commonalities in their network structures. For example, the World Wide Web, the Internet, collaboration networks, citation networks have all been shown to have similar scale-free topologies [41]. Elements of that topology are also observed in the COIN platform. The question is: Do all folksonomies for engineering design display similar topologies? If so, what implications does that have on the design of bottom–up classification systems for engineering design? If not, what other topologies can result? Further, do specific topologies have advantages in terms of information search and retrieval? If so, how can such topologies be fostered?

Answers to these questions may provide new directions to the design of next generation information systems for engineering design.

Acknowledgment

The authors gratefully acknowledge the financial support from the National Science Foundation through the CAREER Grant No. 0954447.

References


