Formal Ontology in Information Systems
M. Donnelly and G. Guizzardi (Eds.)
IOS Press, 2012

doi:10.3233/978-1-61499-084-0-147

Ontology Content "At A Glance"

Gökhan COSKUN [a], Mario ROTHE [a] and Adrian PASCHKE [a]
[a] *Freie Universität Berlin*

Abstract. In the field of software engineering component-based development and appropriate documentation are established methods to support reuse. While modular development is tackled in various work regarding ontology engineering, it is an open problem how documentation of ontologies should be created. After analyzing existing ontology documentations we identified grouping concepts as a very helpful technique to simplify the understandability and thus improve the reusability of ontologies. In this paper, we present a technique to group concepts for ontology documentation by applying community detection algorithms on the graph structure of ontologies. Using the manually created concept groups from existing documentations as reference we demonstrate that this technique is able to create appropriate concept groups automatically.

Keywords. ontology documentation, concept grouping, ontology reuse

Introduction

Ontology reuse attracts the interest of the Semantic Web community and its current pragmatic version the Linked Data community where ontologies are considered as shared knowledge and are interlinked. Even though ontology reuse is part of various ontology engineering methodologies, there are no best practice solutions which describe how existing ontologies should be analyzed for their (re)usability. In the field of software engineering, component-based development and appropriate documentation are established methods to support reuse. While the adoption of the former method to ontology engineering has been addressed in various scientific publications within the ontology engineering community (e.g. [9], [7]) the latter has not been tackled in depth yet. Therefore, only a few of the many ontologies which are published and available online are well documented.

The lack of good documentation makes reuse difficult because the decision process of the applicability of a candidate ontology becomes time-consuming. On the other hand, the process of documentation is an additional effort for the ontology developer which still lacks of an appropriate support system. Aiming at creating such a support system we analyzed existing hand-made ontology documentations and identified grouping of concepts as a proper means to provide an overview of an ontology's content. In case of large ontologies with thousands of concepts it is intuitively comprehensible that some kind of complexity reduction is necessary to understand an ontology. But even the Friend

of a Friend (FOAF) vocabulary[1], which is a small ontology, uses a grouping of concepts in its specification (see Figure 1), in order to provide the reader with an easier way to understand the vocabulary.

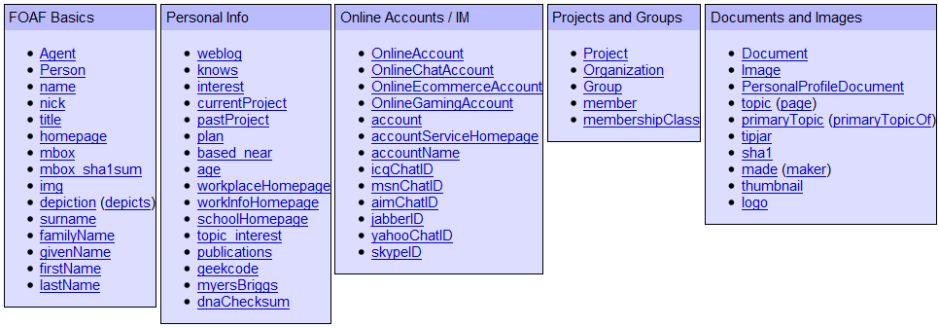

Figure 1. Concept groups of the FOAF vocabulary in the specification (version 0.97)

The application of this method for describing an ontology in other documentations like the Music Ontology[2], the Atom Activity Streams Ontology[3], and the Semantic Web Conference Ontology[4] as well as the Vocabulary for biographical information[5] which are about the same size as FOAF proves how important adequate visualization of meaningful concept groups is. Keeping the rapidly growing Semantic Web [6] in mind and the fact that the large number of ontologies are lightweight and small-sized [5] issues like reusability regarding those ontologies seem to be more urgent. For that reason we analyzed the documentations of the mentioned ontologies and extracted some trends in creating such concept groups. Additionally, we investigated the applicability of community detection algorithms on the ontology structure in order to identify concept groups automatically or at least semi-automatically. An appropriate concept grouping system is expected to be a useful support tool for the ontology engineer to create a proper documentation.

The remainder of this paper is structured as follows: in Section 1 a state-of-the-art of current ontology documentation tools is presented along with a discussion of how some ontology engineers have extended their documentations and grouped the concepts. In Section 2 we present our structure-based approach to grouping concepts. The application of this approach on concrete ontologies and the results are presented in Section 3. We finally conclude this paper with Section 4 and provide an outlook on future work.

1. State of the Art

Inspired by the success of JavaDoc for code written in Java, OWLDoc[6] is a tool that generates frame-based HTML pages with three areas. It allows to navigate quickly to a

[1] http://xmlns.com/foaf/spec/
[2] http://musicontology.com/
[3] http://xmlns.notu.be/aair/
[4] http://data.semanticweb.org/ns/swc/ontology
[5] http://vocab.org/bio/0.1/.html
[6] http://www.co-ode.org/downloads/owldoc/

specific resource and obtain information about it like comments, labels, type etc. When a class is chosen the main frame shows information such as superclasses and disjoint classes, while in case of properties information as superproperty, domain and range is shown. This kind of representation is useful to get detailed information about a single concept and its connections to other concepts. However, it does not provide an overview about the ontology and its structure as a whole.

More recent documentation tools such as Neologism [1], SpecGen[7] and VocDoc[8] create one HTML page containing detailed information about the classes and the properties. Additionally, these HTML pages also contain meta information like version information, changelog, authors, namespaces, license information and referenced external ontologies. This kind of information is either at the beginning of the document or at the end. The details about the ontology and its concepts are at the main part of the document. Before the main part begins there is a short section which is called "overview" or "at a glance" which contains an alphabetically sorted list of classes and properties. Neologism extends this section with a graphical visualization, which is very useful in case of very small ontologies. With increasing number of concepts this visualization gets confusing very quickly.

In the documentations of FOAF, the Music Ontology (MO), the Atom Activity Streams Vocabulary (AAIR) and the Semantic Web Conference Ontology (SWCO) the "at a glance" section is extended with a manually created grouping of concepts. Such a group is also part of the documentation of the Vocabulary for biographical information[9] (BIO). In the opinion of the authors this is a very good means to provide an overview of the subdomains which are covered by the ontology. Concepts, which are more related to each other and describe one subdomain, should be grouped together.

It is a good introduction so that the reader can understand what the ontology is about and can decide very quickly if the content covers relevant concepts for her or his purpose. Therefore this illustration addresses most likely users who are looking for a reusable ontology and want to decide if a closer look makes sense. In consideration of the fact that the documentation of FOAF comprises about 40 printed pages it becomes clear how important such a support is and how much time it can save. Additionally, it emphasizes that even in case of rather small ontologies there is a need for breaking down the complexity for documentation purposes, where concept grouping is one promising means to do this.

After analyzing the aforementioned documentations, we made some interesting observations:

- In most cases there are concepts within the grouping which are not up-to-date. Although the ontology development was continued the documentations were not adapted to the updates. In some cases only parts of the documentations which are generated automatically like the alphabetically sorted classes and properties lists were updated. It is obvious that documentations do not get enough attention by the developers, because even automatically generatable parts were not updated after every change.

[7] http://forge.morfeo-project.org/wiki_en/index.php/SpecGen
[8] http://kantenwerk.org/vocdoc/
[9] http://vocab.org/bio/0.1/.html

- Even though the number of ontologies and groups are not significant to extract some numerical best practices there are some trends that are worth mentioning. The arithmetic means for the concept group size is about eight, while most groups contain five concepts. Three documentations contain five groups (FOAF, BIO, SWCO) while one documentation contains four groups (AAIR). Only the documentation of MO contains much more groups, namely 23. It might make sense to create about 5 groups for ontologies which are about the same size as the mentioned ontologies. Figure 2 illustrates the distribution of the group size.

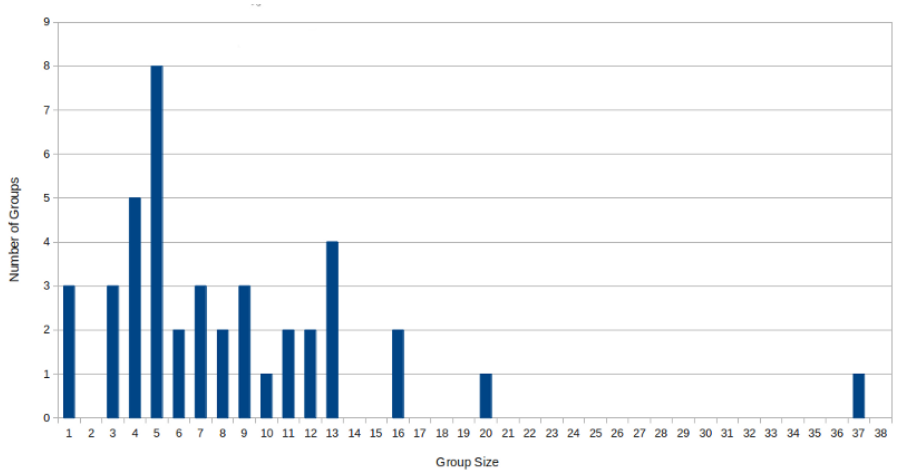

Figure 2. Distribution of the group size (five ontology documentations with 42 groups)

- In each documentation some of the concepts within the ontology were not assigned to any group at all. Only the concepts which are considered to be important by the ontology engineer were grouped. In case of SWCO and AAIR only classes were used to create the groups. On the other hand, MO, BIO as well as FOAF contain some groups which consist only of properties.

2. Structure-based concept grouping

In the Semantic Web ontologies are mostly represented by the Web Ontology Language (OWL) based upon the Resource Description Framework (RDF)[10]. RDF allows representing information as triples following the form (Subject, Predicate, Object). The graph syntax of RDF[11] maps triples to graphs where the subjects and the objects are nodes and the predicates are directed edges (from subject to object). At this level the inherent semantics of OWL ontologies are not taken into consideration.

Since the subjects as well as the predicates of RDF statements need to be resources and objects might also be resources, it is impossible to organize the edges and nodes

[10] http://www.w3.org/TR/owl-semantics/mapping.html describes how OWL is mapped to RDF
[11] http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/

into disjoint sets. This is because a resource, which is a subject or an object in one statement might be used as a predicate in another statement. This problem of the RDF graph representation of triples can be avoided if every named entity of the ontology is represented as a node (even the predicate becomes a node, which is connected with the subject and the object). However, since typically the number of properties is much less than the number of resources which are used as subjects and objects, this graph representation would lead to a graph structure in which the properties are central nodes with high degree values. Some predicates such as "hasLabel" or "hasComment" would have a high centrality value. Hence, it is important to filter and remove such concepts, which have a significant impact on the graph structure analysis of an ontology, but which are not necessary in order to understand the content of an ontology. Furthermore, it is important to take different namespaces into consideration.

We made use of three basic approaches on how to represent an ontology as a graph. Firstly, the RDF graph syntax is used as it is, that means the subject and object of each statement are nodes, while the predicate is the connecting edge, directed from the subject to the object (variant V1). Secondly, the predicates are also represented by nodes, where two unlabeled directed edges are created. One edge is directed from the subject to the predicate, while the second is directed from the predicate to the object (V2). With this variant each named entity appears only once in the whole graph and is represented by a node. Thirdly, a graph is created where only classes are represented as nodes connected by properties as edges, where the direction is from the domain class of the property to the range class of the property (V3). This variant corresponds to the typical graph representation of OWL ontologies and is based on the idea, that classes are the major objects of an ontology, while the properties can be seen as extensions of those classes.

There are also two different extensions of these variants. In the first extension (named as VxL) the literals are filtered during the graph creation process. This filter is enhanced by the second extension (named as VxLX) by excluding concepts with external namespaces. Summing up, for our analysis we used nine different graph variants for each ontology. The different variants are shown in Table 1.

Table 1. Different graph representations for ontologies

Variant name	Description
V1	Plain RDF graph
V1L	like V1 without literals
V1LX	like V1L without external namespaces
V2	Plain RDF graph, but predicates are represented as nodes
V2L	like V2 without literals
V2LX	like V2L without external namespaces
V3	Class graph
V3L	like V3 without literals
V3LX	like V3L without external namespaces

2.1. Concept Groups as Communities

There are different terms in the literature which are used to describe more or less the same process that we call concept grouping. Network partitioning, graph partitioning,

clustering and segmentation are some examples for such terms. We define the process which we refer as concept grouping as identifying the groups of concepts based on the network structure of the ontology in such a way, that the concepts within a group are belonging stronger to each other than to the concepts of another group.

The mathematical approach (mostly named as graph partitioning) looks for subgraphs which are about the same size in such a way that the connections between these subgraphs are minimized. For ontologies this approach does not seem to be suitable because ontologies model various parts (subdomains) of a domain in different levels of detail. E.g. the concept groups of the FOAF ontology are of different size (see in Figure 1). This work is based on the assumption that different subdomains of a domain are reflected in an ontology in such a way that the concepts belonging to one subdomain are building a more densely connected graph partition - a community. For that reason a social network analysis approach for detecting communities seems to be more suitable for identifying concept groups within ontologies.

In order to investigate the applicability of different community detection algorithms to the ontologies, we applied the following algorithms on the different graph representation variants of the ontologies which are listed in Table 1. (Based on the findings of our previous work [4] we focused on the three most promising algorithms and decided to omit the Edge Betweenness Community algorithm as well as the Leading Eigenvector Community algorithm.)

Fast Greedy Community The Fast Greedy Community (FGC) algorithm introduced in [3] identifies communities by optimizing a Modularity [11] score, which is a network property and a specific proposed partitioning of that network into communities. It evaluates the partitioning, in the sense that in a good partitioning there are many edges within communities and only a few between them. This algorithm is an agglomerative clustering algorithm, which means that the communities are built step by step by merging vertices into communities. This algorithm is optimized for large networks and is called *fast* because does not check modularity after each merge.

Walktrap Community Pons and Latapy are proposing an algorithm in [13] which is based on the same community idea as the FGC algorithm. It is stated that "random walks on a graph tend to get "trapped" into densely connected parts corresponding to communities." For that reason this algorithm is called Walktrap Community (WTC). As FGC this is also an agglomerative clustering algorithm.

Spin Glass Community The Spin Glass Community (SGC) algorithm for community detection was proposed by Reichardt and Bornholdt in [14,11]. It makes use of the model of a spin glass and simulated annealing. The community structure of the network is interpreted as the spin configuration that minimizes the energy of the spin glass with the spin states being the community indices.

2.2. *Weight function for properties*

For each of the mentioned algorithms we created a second version (named WTC_w, SGC_w, FGC_w) that is extended with a weight function for the edges of the graph. This is the first step towards a more semantically sensitive approach. The motivation for the weighting of the edges is that they represent different kinds of the semantics of an ontology. A link between two classes, that represents a property, does not necessarily contribute as much

to the membership of an element to a group as a hierachical link. We use the weights shown in Table 2. (The default value for edges which are not listed in the table is 1. If the superclass is *Thing* the *subClassOf* edge value in the table is not used.)

Table 2. Weights for properties

Property	**Weight**	**Property**	**Weight**
equivalentClass	20	comment	0.2
subClassOf	10	seeAlso	0.2
subPropertyOf	10	isDefinedBy	0.2
domain	5	label	0.2
range	5		

Classes that are semantically equivalent (connected via the property equivalentClass) should always be placed in the same group, therefore the edges between them have the highest weight. Usually such edges exists only between classes of different namespaces to link some vocabularies. If classes from external namespaces are not relevant, one can use one of the VxLX variants (see Table 1) of the graph representation which filters elements from external namespaces.

According to [15], a categorization of classes in a taxonomy provides maximum information with the least cognitive effort. The same principle is used for ontologies where hierachical relations of the classes and properties include much of the semantic content. Based on the inheritance of properties it is stated in [12] that hierachical links highly contribute to a module's degree of internal connectedness. The importance of the hierachical organization of classes is also reflected in the fact, that the subClassOf relation is introduced in RDFS, which is one of the basic levels of the Semantic Web stack.

The increased weights of the domain and range edges compared to the default value represents the idea that classes connected via a property are semantically closer than two classes that both share the same type (for example owl:Class). Edges used to connect to literals (datatype properties) have been set below the default weight of 1, because they are not part of the semantic model. Besides, this fixes some centrality problems that occur with the common edges in variant V2.

2.3. Evaluating Concept Groups

It is not possible to decide how good an ontology is without knowing the context in which it is intended to be used, because most ontologies are built in an application dependent manner. And even if the context is known there are always different ways to create a conceptual model of a domain. The ontology module evaluation techniques which are proposed in literature are based on the structure of the ontology. The common idea behind these techniques is that information provided by different modules, should be - as far as possible - independent and disjoint. That means that each module represents a subdomain of the domain which is modeled by the whole ontology.

Previous work like [9] and [16] make use of very simple structural information as the number of modules, average module size, size variance, and the connectedness be-

tween the modules to evaluate ontology modularization. Calmet et. al. propose in [2] a distance measure for two concepts within an ontology based on the notion of entropy in order to measure the similarity between two modules. This approach is extended in [8] by distinguishing between language level edges and domain level edges, so that two different entropies are calculated, namely the language level entropy and the domain level entropy. By distinguishing between two kinds of edges a first step towards a semantically sensitive technique has been made.

A pure structure-based measure is not adequate to evaluate the structure-based modularization techniques, since the modularization can be always optimized in such a way that the evaluation score is increased. Instead, we use as a "gold standard" (reference model for the evaluation of the modularization technique) the existing concept groupings as they have been designed and described by the ontology engineers. The rationale for this evaluation approach is that the quality of an ontology module usually depends on the application where it is going to be used. As our purpose for the technique is to create concept groups for documentation support, we use existing groupings which have been introduced by the ontology engineers in documentations. This means that for evaluating the result of the algorithms we just need to compare the reference model with the created concept grouping and calculate the similarity. For that purpose we make use of the F-Measure which is a pairs-based approach and can be found in e.g. [17]. It is calculated with

$$F = \frac{2 * precision * recall}{precision + recall} \tag{1}$$

where "the precision of a partition is defined as the ratio of intra-pairs in the generated partitioning that are also intra-pairs in the optimal partitioning." and "the recall of a partition is defined by the ratio of intra-pairs in the optimal partitioning that are also in the generated one." [17].

3. Analysis

For our analysis we implemented a lightweight web application which uses R[12] with the igraph[13] library for the implementation of the algorithms. Before calculating the groups, the ontology documents are loaded with Jena[14] and are converted into GraphML[15] files according to the variants which are shown in Table 1. Before this process is started the ontologies are loaded in two different ways. Firstly, with inactive inference and secondly, with active inference[16]. Inference has a significant influence on the structure of an ontology, which can be seen on the increasing number of statements in Table 3.

As we use hand-made concept grouping to evaluate our results following the techniques presented in section 2.3 we searched for ontologies, which are divided into concept groups in their documentations. We found the aforementioned ontologies FOAF,

[12] http://www.r-project.org

[13] http://igraph.sourceforge.net

[14] http://jena.sourceforge.net/

[15] http://graphml.graphdrawing.org/

[16] In contrast to a preceding work [4], we only use the RDFS reasoning support of the Jena OWL Reasoner (OWL MEM RDFS INF)

MO, AAIR, SWCO and BIO. Table 3 provides an overview about the size of these ontologies and the concept groups from the documentations.

Table 3. Properties of the different Ontologies

Ontology	Classes in			Properties in			RDF Statements	
	Ontology	Doc	Groups	Ontology	Doc	Groups	normal	inferred
FOAF	13	13	12	60	61	51	613	972
MO	59	53	48	163	137	85	2092	3576
AAIR	41	41	39	26	26	0	437	771
SWCO	29	29	29	16	16	0	848	1632
BIO	42	42	37	33	33	27	968	1702

3.1. Analysis results

The Tables 4 to 8 show the scores for the results of the algorithms combined with the graph variants for each ontology. In each cell the evaluation result (multiplied by 100) is listed for the corresponding combination of graph variant (row) and algorithms (column). Some values are missing, because the corresponding variants produce graphs which cannot be processed by some algorithms.

The Atom Activity Streams Vocabulary (AAIR) (analysis results in Table 4) defines concepts to describe activities within social networking sites, while the Semantic Web Conference Ontology (SWCO) (analysis results in Table 5) is a vocabulary that allows to describe academic conferences. In both cases the number of classes are about twice the number of properties. The distribution of the properties to the classes is rather balanced. The basic concepts are mainly refined with subclasses. The focus on classes gets clear after looking at the documentations of AAIR and SWCO, because both contain only groups of classes without properties. This explains why the class-centric graph representation (variant V3) leads to the best scores in both cases.

Table 4. Analysis results for AAIR

	no Inference							Inference						
	FGC	**FGC$_w$**	**SGC**	**SGC$_w$**	**WTC**	**WTC$_w$**	**AVG**	**FGC**	**FGC$_w$**	**SGC**	**SGC$_w$**	**WTC**	**WTC$_w$**	**AVG**
V1	36	60	—	—	27	62	**46**	52	59	66	58	58	88	**63**
V1L	36	58	—	—	71	55	**55**	57	59	56	52	62	88	**62**
V1LX	56	54	—	—	—	—	**55**	58	64	—	—	—	—	**61**
V2	43	51	38	39	51	51	**46**	37	51	36	37	51	51	**44**
V2L	49	51	36	36	39	51	**44**	41	51	43	44	43	51	**45**
V2LX	41	47	36	35	49	44	**42**	49	51	38	45	51	51	**48**
V3	39	64	42	47	9	64	**44**	56	85	50	48	43	88	**62**
V3L	51	64	50	66	54	64	**58**	62	85	78	73	72	88	**76**
V3LX	64	64	67	63	62	62	**64**	87	75	87	79	68	63	**77**
AVG	**46**	**57**	**45**	**48**	**45**	**57**		**55**	**65**	**57**	**55**	**56**	**71**	

The vocabulary for biographical information (BIO) (analysis results in Table 6) is a vocabulary that defines concepts to describe biographical information about people.

Table 5. Analysis results for SWCO

	no Inference							Inference						
	FGC	**FGC$_w$**	**SGC**	**SGC$_w$**	**WTC**	**WTC$_w$**	**AVG**	**FGC**	**FGC$_w$**	**SGC**	**SGC$_w$**	**WTC**	**WTC$_w$**	**AVG**
V1	52	92	—	—	—	—	**72**	66	79	79	79	79	79	**77**
V1L	60	97	—	—	—	—	**79**	68	79	79	79	79	79	**77**
V1LX	87	91	—	—	—	—	**89**	96	80	—	—	—	—	**88**
V2	26	28	27	30	28	31	**28**	28	28	27	27	31	31	**29**
V2L	27	28	27	27	31	31	**29**	27	28	19	27	31	31	**27**
V2LX	28	28	47	48	31	31	**36**	28	28	48	45	31	31	**35**
V3	76	100	90	83	3	97	**75**	61	79	74	67	69	79	**71**
V3L	95	100	95	95	95	100	**96**	79	79	79	79	79	79	**79**
V3LX	93	100	93	93	92	100	**95**	100	76	92	92	65	69	**82**
AVG	**61**	**74**	**63**	**62**	**47**	**65**		**62**	**62**	**62**	**62**	**58**	**60**	

Table 6. Analysis results for BIO

	no Inference							Inference						
	FGC	**FGC$_w$**	**SGC**	**SGC$_w$**	**WTC**	**WTC$_w$**	**AVG**	**FGC**	**FGC$_w$**	**SGC**	**SGC$_w$**	**WTC**	**WTC$_w$**	**AVG**
V1	67	70	84	84	79	77	**77**	80	88	85	86	82	88	**85**
V1L	82	70	87	85	82	66	**79**	84	77	84	80	85	89	**83**
V1LX	84	60	70	71	84	68	**73**	78	81	87	87	83	86	**84**
V2	71	83	78	83	54	79	**75**	56	83	78	78	54	85	**72**
V2L	42	83	80	43	56	79	**64**	42	80	62	60	54	78	**63**
V2LX	63	86	41	52	71	85	**66**	63	83	67	75	81	85	**76**
V3	16	56	26	26	33	59	**36**	28	58	52	58	50	58	**51**
V3L	56	56	56	56	59	59	**57**	64	56	58	58	64	58	**59**
V3LX	59	56	51	54	59	56	**56**	60	58	46	46	31	31	**45**
AVG	**60**	**69**	**64**	**61**	**64**	**70**		**62**	**74**	**69**	**70**	**65**	**73**	

As it describes the events during the life of a person the basic concepts are *Person* and *Event*, which are also the domain classes of all properties. The number of classes and the number of properties at all are not that different in the documentation. However, the main concepts are also mainly refined by subclasses. The documentation has only one group for classes and three groups for properties. Because of this distinction variant V1 leads to the best score.

FOAF is a vocabulary (analysis results in Table 7) that allows to express and interlink personal information. It contains much more properties than classes (see Table 3), where *Agent* and its subclass *Person* are the most important classes, as they are the domain classes of the most properties of this vocabulary. In fact, the main focus of FOAF is on the definition of these two classes. The documentation contains four groups with properties and classes and one group with only properties, which are mainly properties of *Person*. The unbalanced distribution of the properties to the classes along with the mixture of classes and properties within the groups explain why the score is at a low level.

The Music Ontology (MO) (analysis result in Table 8) provides concepts to describe and link music information. It is the biggest ontology in this analysis and contains much more properties than classes. In contrast to the other ontologies, the number of concept groups within the documentation is much higher and the groups are not disjoint. There are groups with only classes, groups with only properties and also groups with classes and properties as well as groups with only one concept. The size of the groups and the

Table 7. Analysis results for FOAF

	no Inference							Inference						
	FGC	**FGC$_w$**	**SGC**	**SGC$_w$**	**WTC**	**WTC$_w$**	**AVG**	**FGC**	**FGC$_w$**	**SGC**	**SGC$_w$**	**WTC**	**WTC$_w$**	**AVG**
V1	27	36	31	32	37	33	**32**	28	38	33	33	32	28	**32**
V1L	32	39	31	35	34	30	**33**	37	40	40	37	32	30	**36**
V1LX	36	40	—	—	—	—	**38**	45	45	—	—	37	31	**39**
V2	34	32	34	35	34	34	**34**	34	34	34	34	34	34	**34**
V2L	34	32	35	34	34	33	**34**	33	34	36	36	34	34	**34**
V2LX	26	30	26	25	33	32	**29**	36	30	20	28	34	34	**30**
V3	37	38	37	37	30	38	**36**	35	35	34	35	31	35	**34**
V3L	37	38	36	35	32	37	**36**	31	35	30	31	30	34	**32**
V3LX	35	35	35	35	31	34	**34**	31	34	33	30	34	35	**33**
AVG	**33**	**36**	**33**	**33**	**33**	**34**		**34**	**36**	**33**	**33**	**33**	**33**	

distribution of properties are not balanced which explains why the score is at a similar level as the score for FOAF.

Table 8. Analysis results for Music Ontology

	no Inference							Inference						
	FGC	**FGC$_w$**	**SGC**	**SGC$_w$**	**WTC**	**WTC$_w$**	**AVG**	**FGC**	**FGC$_w$**	**SGC**	**SGC$_w$**	**WTC**	**WTC$_w$**	**AVG**
V1	30	31	26	27	14	19	**25**	28	27	24	25	13	21	**23**
V1L	26	30	26	28	27	29	**28**	18	32	20	20	20	28	**23**
V1LX	22	29	22	25	23	34	**26**	23	33	25	20	19	32	**25**
V2	14	17	13	13	11	14	**14**	11	17	13	13	11	16	**14**
V2L	13	16	14	14	13	13	**14**	14	17	13	12	13	16	**14**
V2LX	16	22	19	20	15	16	**18**	13	19	16	15	16	17	**16**
V3	22	28	19	21	16	27	**22**	23	28	29	23	21	29	**25**
V3L	23	29	21	20	20	29	**23**	25	28	26	26	22	29	**26**
V3LX	27	28	30	30	19	28	**27**	27	28	27	27	25	31	**27**
AVG	**21**	**26**	**21**	**22**	**18**	**23**		**20**	**25**	**21**	**20**	**18**	**24**	

4. Conclusion and Outlook

The analysis described in the previous section leads in three of five cases to very good results. In case of SWCO it was possible to completely reconstruct the grouping from the documentation. The application of community detection algorithms on ontologies produce good results if concepts are mainly refined with subclasses and the distribution of properties to the classes is balanced. This approach seems to be best to create vertical modules [10] of an ontology which was exactly the expectation, as the main motivation for creating concept groups was to allow an overview on the subdomains.

Scores at a low level for FOAF and MO seem to be caused by the characteristics of these vocabularies and their groupings within the documentation. In both cases the central concepts are mainly refined with properties, which is the reason why they contain much more properties than classes and most groups consist of a mixture of properties and classes. After an additional look at the documentations of MO and FOAF (and the latest version of FOAF) the main idea by creating the concept groups in the documentations

seem to be the provision of different levels of detail for one domain. This means, that the main goal is to create horizontal modules with different levels of abstraction.

Finally, an important observation is that for each ontology the best score was reached with either FGC_w or with WTC_w. The introduction of the weight functions (see Section 2.2) improved the results. The findings of the analysis demands further exhaustive investigation on the relation between different qualitative aspects of an ontology and the concept groups created by the algorithms. It is necessary to do more analysis with other ontologies and to discuss the role of complex expressions, which was ignored in this work. Additionally, it is important to analyze the quality of the existing groupings in the documentations, although the assumption of this work was, that they can be used as a gold standard, due to the fact, that they were created manually. We are also planning to investigate the applicability of community detection algorithms on modular build ontologies in order to reproduce their modularity after a merge. The main goal is to understand the relations between different modules of an ontology and to extract structural trends in modularizing ontologies not for concept groups but for modular ontologies.

Acknowledgement

This work has been partially supported by the "InnoProfile-Corporate Semantic Web" project funded by the German Federal Ministry of Education and Research (BMBF) and the BMBF Innovation Initiative for the New German Länder - Entrepreneurial Regions.

References

[1] Cosmin Basca, Stphane Corlosquet, Richard Cyganiak, Sergio Fernndez, and Thomas Schandl. Neologism: Easy vocabulary publishing. In *Proceedings of the Workshop on Scripting for the Semantic Web, in conjunction with ESWC 2008*, 2008.

[2] Jacques Calmet and Anusch Daemi. From entropy to ontology. In *AT2AI-4 - Fourth International Symposium "From Agent Theory to Agent Implementation" at the 17th European Meeting on Cybernetics and Systems Research (EMCSR)*, pages 547–551, Vienna, Austria, 2004.

[3] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical Review E - Statistical, Nonlinear and Soft Matter Physics*, 70(6):1–6, December 2004.

[4] Gökhan Coskun, Mario Rothe, Kia Teymourian, and Adrian Paschke. Applying community detection algorithms on ontologies for indentifying concept groups. In *Proceedings of the Fifth International Workshop on Modular Ontologies (WoMO 2011)*, pages 12–24, Ljubljana, Slovenia, August 2011.

[5] Mathieu d'Aquin, Claudio Baldassarre, Laurian Gridinoc, Sofia Angeletou, Marta Sabou, and Enrico Motta. Characterizing knowledge on the semantic web with watson. In *Evaluation of Ontologies and Ontology-Based Tools: 5th International EON Workshop*, volume 329 of *CEUR Workshop Proceedings*, pages 1–10. CEUR-WS.org, 2007.

[6] L. Ding and Tim Finin. Characterizing the semantic web on the web. In *Proceedings of the 5th International Semantic Web Conference (ISWC 2006)*, pages 242–257, Athens, GA, USA, 2006. Springer.

[7] Paul Doran, Valentina Tamma, and Luigi Iannone. Ontology module extraction for ontology reuse: an ontology engineering perspective. In *CIKM '07: Proceedings of the sixteenth ACM Conference on information and knowledge management*, pages 61–70, New York, NY, USA, 2007. ACM.

[8] Paul Doran, Valentina A. M. Tamma, Terry R. Payne, and Ignazio Palmisano. An entropy inspired measure for evaluating ontology modularization. In *5th International Conference on Knowledge Capture (KCAP'09)*, pages 73–80, Redondo Beach, CA, USA, September 2009. ACM.

[9] Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. Modular reuse of ontologies: Theory and practice. *Journal of Artificial Intelligence Research (JAIR)*, 31:273–318, 2008.

[10] Frank Loebe. Requirements for logical modules. In *Proceedings of the 1st International Workshop on Modular Ontologies, WoMO'06, co-located with the International Semantic Web Conference, ISWC'06*, Athens, GA, USA, 2006. CEUR-WS.org.

[11] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E - Statistical, Nonlinear and Soft Matter Physics*, 69(2):413–421, 2004.

[12] Sunju Oh, Heon Y. Yeom, and Joongho Ahn. Evaluating ontology modularization approaches. In *Proceedings of the 8th International Conference on Frontiers of Information Technology*, FIT '10, pages 6:1–6:6, New York, NY, USA, 2010. ACM.

[13] Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks. *Journal of Graph Algorithms and Applications*, 10(2):191–218, 2005.

[14] Jörg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Physical Review E - Statistical, Nonlinear and Soft Matter Physics*, 74(1):016110, 2006.

[15] E. Rosch. Principles of categorization. In E. Rosch and B. Lloyd, editors, *Cognition and categorization*, pages 27–48. Erlbaum, Hillsdale, New Jersey, 1978.

[16] Anne Schlicht and Heiner Stuckenschmidt. A flexible partitioning tool for large ontologies. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT '08)*, pages 482–488, Sydney, Australia, December 2008. IEEE Computer Society.

[17] Heiner Struckenschmidt. Network analysis as a basis for partitioning class hierarchies. In *ISWC 2005 Workshop on Semantic Network Analysis (SNA'05)*, pages 43–54, Galway, Ireland, 2005.