Extracting Sequential Patterns from Collaborative Ontology-Engineering Projects

Technical Report

Simon Walk[1]

with
Philipp Singer[1]
Markus Strohmaier[2,3]
Tania Tudorache[4]
Natalya F. Noy[4]
Mark A. Musen[4]

[1] Knowledge Technologies Institute
Graz University of Technology, Austria
[2] Computational Social Science
GESIS - Leibniz Institute for the Social Sciences, Cologne, Germany
[3] Department of Computer Science
University of Koblenz-Landau, Germany
[4] Biomedical Informatics Research Center
Stanford University, California, USA

March, 2014

Abstract. This technical report summarizes the results and work that was conducted by me during my research visit at Stanford University from September to December 2013. During my stay, I performed a detailed analysis and evaluation of sequential patterns in five different collaborative ontology-engineering projects from the biomedical domain. In ongoing work, I will implement a plug-in for WebProtégé in cooperation with researchers at Stanford University, which leverages the results of the sequential pattern analysis to create task recommendations.

Parts of this technical report have already been submitted to the Journal of Biomedical Informatics.

Biomedical taxonomies, thesauri and ontologies, such as the International Classification of Diseases (ICD) or the National Cancer Institute Thesaurus (NCIt), play a critical role in acquiring, representing and processing information about human health. With increasing adoption and relevance, biomedical ontologies have also significantly increased in size. For example, the 11^{th} revision of the International Classification of Diseases (ICD-11), which is currently under active development by the World Health Organization (WHO) contains nearly $50,000$ classes representing a vast variety of different diseases and causes of death. This evolution in terms of size was accompanied by an evolution in the way ontologies are engineered. Because no single individual has the expertise to develop such large-scale ontologies, ontology-engineering projects have evolved from small-scale efforts involving just a few domain experts to large-scale projects that require effective collaboration between dozens or even hundreds of experts, practitioners and other stakeholders. Understanding the way these different stakeholders collaborate will enable us to improve editing environments that support such collaborations. In this paper, we uncover how large ontology-engineering projects such as ICD-11 unfold by analyzing usage logs of five different biomedical ontology-engineering projects of varying sizes and scopes. We discover intriguing patterns that suggest that large collaborative ontology-engineering projects are governed by a few general principles that determine and drive development. From our analysis, we identify commonalities and differences between distinct projects that have implications for project managers, ontology editors, developers and contributors working on collaborative ontology engineering projects and tools in the biomedical domain.

Table of Contents

Acknowledgements

This work was generously funded by a Marshall Plan Scholarship issued by the Marshall Plan Foundation with support from Graz University of Technology. I would also like to express my gratitude to Prof. Markus Strohmaier, who introduced me to the Prof. Mark Musen from the Biomedical Informatics Research Center at Stanford University. Additionally, I would also like to thank everyone at the BMIR, in particular Natasha Noy Ph.D., Tania Tudorache Ph.D. and Prof. Mark Musen, for supporting me and my research and making me feel welcomed at their institute.

1 Introduction

Today, biomedical ontologies play a critical role in acquiring, representing and processing information about human health. For example, the International Classification of Diseases (ICD) is used in more than 100 countries to encode patient diseases, to compile health-related statistics and to collect health-related spending statistics. Similarly, the National Cancer Institute's Thesaurus (NCIt) represents an important vocabulary for classifying cancer and cancer-related terms.

With their increase in relevance, biomedical ontologies have also significantly increased in size to cover new findings and to extend and complement their original areas of application. For example, the 11^{th} revision of the International Classification of Diseases (ICD-11), currently under active development by the World Health Organization (WHO), consists of nearly $50,000$ classes representing a vast variety of different diseases and causes of death. This growth was accompanied by a need to adapt the way these ontologies are developed as no single individual or small group of domain experts have the expertise to develop such large-scale ontologies. New tools and processes have to be developed in order to coordinate, augment and manage collaboration between the dozens or hundreds of experts, practitioners and stakeholders when engineering an ontology.

Understanding the ways in which such a large number of participants – e.g., more than 100 experts contribute to ICD-11 – collaborate with one another when creating a structured knowledge representation is a prerequisite for quality control and effective tool support.

Objectives: Consequently, we aim at understanding how large collaborative ontology engineering projects such as ICD-11 unfold. We approach this problem by analyzing usage logs of five biomedical ontology-engineering projects of varying sizes and scopes. For this analysis we employ Markov chain models for investigating whether we can use the history of user actions for the task of predicting the next action that the user is going to perform (e.g., the user who will change a class next). The analyzed datasets range from large-scale datasets such as ICD-11 to smaller ones such as the Ontology for Parasite Lifecycle (OPL). We aim to explore differences and similarities in patterns among different biomedical ontology-engineering projects (e.g., we look at sequences of users who edit a class). Furthermore, we aim to identify and discuss features of these projects that potentially affect the patterns that we observed. This analysis can be seen as a stepping stone for collaborative ontology-engineering project managers to devise infrastructures and tool support to augment collaborative ontology engineering.

Findings: We discover new insights on social interactions and editing patterns that suggest that large collaborative ontology engineering projects are governed by a few general principles that determine and drive development. Specifically, our results indicate that general edit patterns can be found in all investigated datasets, even though they (i) represent different projects with different goals, (ii) use variations of the same ontology-editors and tools for the engineering process and (iii) differ in the way the projects are coordinated.

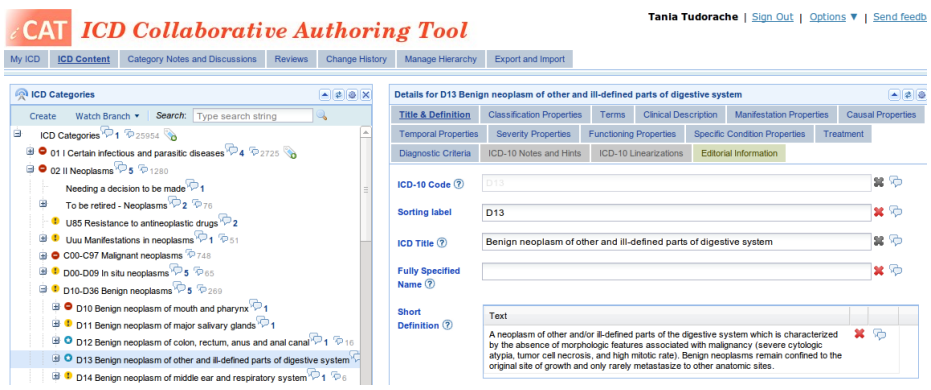


Fig. 1. A screenshot of iCAT, a custom tailored, web-based version of WebProtégé, developed for the collaborative engineering of ICD-11. The left part of the interface visualizes the ICD-11 class hierarchy, the class titles, the number of annotations each class has received (speech bubbles) and its overall progress (color and symbol before the class title). The right part of the interface shows the different user-interface sections (e.g, *Title & Definition* or *Classification Properties*), listing all properties and property values for each class.

We find first evidence that users collaborate in micro-workflows, where information about previous users contains information about which user is likely to edit a given class next (Section 4.1). We show that specific roles – e.g., the so-called gardener, a contributor focused on pruning ontology classes and fixing syntactical errors – of an ontology [1]) exhibit different transition probabilities and features from regular contributors. Users tend to move along special ontological relationships and do not follow a single, but a combination of multiple strategies when editing ontologies (e.g., *bottom-up*, *top-down*, *breadth-first* or *depth-first*) (Section 4.2). Finally, we find that users tend to change the same properties across different classes. When investigating which properties are changed for a class, we find out that (different) contributors consecutively change the same properties multiple times, rather than multiple properties for each class (Section 4.3).

Contributions: To the best of our knowledge, the work presented in this paper represents the most fine-grained and comprehensive study of patterns in large-scale collaborative ontology-engineering projects in the domain of biomedicine. In addition, our analysis is conducted across five datasets of different sizes, which have been developed using different versions of Collaborative Protégé (Table 1). Our results reveal that (i) general social patterns and sequential patterns can be found across all five projects and (ii) there are commonalities and differences between extracted patterns in our datasets.

2 Collaborative Ontology Engineering

According to Gruber [2], Borst [3], Studer et al. [4] an ontology is an explicit specification of a shared conceptualization. In particular, this definition refers to a machine-readable construct (the formalization) that represents an abstraction of the real world (the shared conceptualization), which is especially important in the field of computer science as it allows a computer (among other things) to "understand" relationships between entities and objects that are modeled in an ontology.

Collaborative ontology engineering is a new field of research with many new problems, risks and challenges that we must first identify and then address. In general, contributors of collaborative ontology-engineering projects, similar to other collaborative online production systems (e.g., Wikipedia), engage remotely (e.g., via the internet or a client–server architecture) in the development process to create and maintain an ontology. As an ontology represents a formalized and abstract representation of a specific domain, disagreements between authors on certain subjects can occur. Similar to face-to-face meetings, these collaborative ontology-engineering projects need tools that augment collaboration and help contributors in reaching consensus when modeling topics of the real world.

Indeed, the majority of the literature about collaborative ontology engineering sets its focus on surveying, finding and defining requirements for the tools used in these projects [5, 6].

The Semantic Web community has developed a number of tools aimed at supporting the collaborative development of ontologies. For example, Semantic MediaWikis [7] and its derivatives [8, 9, 10] add semantic, ontology modeling and collaborative features to traditional MediaWiki systems.

Protégé, and its extensions for collaborative development, such as WebProtégé and iCAT [11] (see Figure 1 for a screenshot of the iCAT ontology-editor interface) are prominent stand-alone tools that are used by a large community worldwide to develop ontologies in a variety of different projects. Both WebProtégé and Collaborative Protégé provide a robust and scalable environment for collaboration and are used in several large-scale projects, including the development of ICD-11 [12].

Pöschko et al. [13], Walk et al. [14] have created *PragmatiX*, a tool to visualize and analyze a collaboratively engineered ontology and aspects of its history and the engineering process, providing quantitative insights into the ongoing collaborative development processes.

Falconer et al. [15] investigated the change-logs of collaborative ontology-engineering projects, showing that users exhibit specific roles, which can be used to group and classify users, when contributing to the ontology. Pesquita and Couto [16] investigated if the location and specific structural features can be used to determine if and where the next change is going to occur in the Gene Ontology[5]. Strohmaier et al. [17] investigated the hidden social dynamics that take place in collaborative ontology-engineering projects from the biomedical

[5] `http://www.geneontology.org`

domain and provides new metrics to quantify various aspects of the collaborative engineering processes. Wang et al. [18] have used association-rule mining to analyze user editing patterns in collaborative ontology-engineering projects. The approach presented in this paper uses Markov chains to extract much higher detailed user-interaction patterns incorporating a variable number of historic editing information.

The only requirement to perform the pattern analysis that we present in this paper is the availability of a structured log of changes that can be mapped to the underlying ontology. The majority of the discussed collaborative ontology-engineering environments provide such a log, allowing for a similar analysis. For example, the Semantic MediaWikis store all the changes to the articles, and thus the ontology, allowing to expand the application of Markov chains to analyze sequential patterns as shown in this paper.

3 Materials & Methods

For the analysis conducted in this paper we concentrated our efforts on five ontology-engineering projects in the biomedical domain. Each of the projects (i) has at least two users who contributed to the project, (ii) provides a structured log of changes and (iii) represents knowledge from the biomedical domain. In section 3.1 we provide a brief history for each dataset and in section 3.2 we describe the sequential path analysis. To aid readers in understanding the analyses conducted in this paper and its implications we provide a very brief overview of Markov chains and the involved model selection methodology in section 3.3.

Table 1. Detailed information of the datasets used for the sequential pattern analysis to extract beaten paths in collaborative ontology-engineering projects.

		ICD-11	ICTM	NCIt	BRO	OPL
Ontology	classes	48,771	1,506	102,865	528	393
	changes	439,229	67,522	294,471	2,507	1,993
	DL expressivity	$\mathcal{SHOIN}(\mathbf{D})$	$\mathcal{SHOIN}(\mathbf{D})$	$\mathcal{SH}$	$\mathcal{SHIF}(\mathbf{D})$	$\mathcal{SHOIF}$
Editor	tool	iCAT	iCAT-TM	Collaborative Protégé	WebProtégé	Collaborative Protégé
Users	users	109	27	17	5	3
	bots (changes)	1 (935)	1 (1)	0 (0)	0 (0)	0 (0)
Duration	first change	18.11.2009	02.02.2011	01.06.2010	12.02.2010	09.06.2011
	last change	29.08.2013	17.7.2013	19.08.2013	06.03.2010	23.09.2011
	observation period (ca.)	4 years	2.5 years	3 years	1 month	3 months

3.1 Datasets

Table 1 lists the detailed features and observation periods for the following five datasets that we used in our analysis.

The International Classification of Diseases (ICD)[6] is the international standard for diagnostic classification used to encode information relevant to epidemiology, health management, and clinical use in over 100 United Nations countries. The World Health Organization (WHO) develops ICD, and publishes new revisions of the classification every decade or more. The current revision in use is ICD-10, which contains over 15,000 classes. The 11th revision of ICD,[7] **ICD-11**, is currently taking place and brings two major changes with respect to previous revisions. First, ICD-11 is developed as an OWL ontology using a much richer representation formalism than previous revisions. ICD-11 contains very detailed descriptions of several aspects of diseases, mostly represented as properties in the ontology. Second, the development of ICD-11 takes place in a Web-based collaborative environment, called iCAT (see Figure 1), which allows domain experts around the world to contribute and review the ontology online. ICD-11 is planned to be finalized in May 2017.

The International Classification of Traditional Medicine (ICTM) is a WHO led project that aimed to produce an international standard terminology and classification for diagnoses and interventions in Traditional Medicine.[8] ICTM is represented as an OWL ontology, which tries to unify the knowledge from the traditional medicine practices from China, Japan and Korea. Its content is authored in 4 languages: English, Chinese, Japanese and Korean. More than 20 domain experts from the three countries developed ICTM using a customized version of the iCAT system, called iCAT-TM. The development of ICTM was stopped in 2012, and a subset of ICTM is also included as a branch in the ICD-11 ontology.[9]

The National Cancer Institute's Thesaurus (NCIt) [19] has over 100,000 classes and has been in development for more than a decade. It is a reference vocabulary covering areas for clinical care, translational, basic research, and cancer biology. A multidisciplinary team of editors works to edit and update the terminology based on their respective areas of expertise, following a well-defined workflow. A lead editor reviews all changes made by the editors. The lead editor accepts or rejects the changes and publishes a new version of the NCI Thesaurus. The NCI Thesaurus is an OWL ontology, which uses many OWL primitives such as defined classes and restrictions.

The Biomedical Resource Ontology (BRO) originated in the Biositemaps project,[10] an initiative of the Biositemaps Working Group of the NIH National Centers for Biomedical Computing [20]. Biositemaps is a mechanism for researchers working in biomedicine to publish metadata about biomedical data,

[6] `http://www.who.int/classifications/icd/en/`
[7] `http://www.who.int/classifications/icd/ICDRevision/`
[8] `http://tinyurl.com/ictmbulletin`
[9] The ICD-11 dataset used in our analysis did not include the ICTM branch.
[10] `http://biositemaps.ncbcs.org`

tools, and services. Applications can then aggregate this information for tasks such as semantic search. BRO is the enabling technology used in Biositemaps; a controlled terminology for describing the resource types, areas of research, and activity of a biomedical related resource. BRO was developed by a small group of editors, who use a Web-based interface (WebProtégé) to modify the ontology and to carry out discussions to reach consensus on their modeling choices.

The Ontology for Parasite Lifecycle (OPL) models the life cycle of the *T.cruzi*, a protozoan parasite, which is responsible for a number of human diseases. OPL is an OWL ontology that extends several other OWL ontologies. It uses many OWL constructs such as restrictions and defined classes. Several users from different institutions collaborate on OPL development. This ontology is much smaller and has far fewer users than NCIt, ICD-11, or ICTM.

3.2 Sequential Paths

For our sequential pattern analysis we analyze three different kinds of paths, which all represent actual interactions with the underlying ontology. A sequential path is represented by the chronologically ordered list of extracted states for either one user or one class (see Figure 2). For example, a sequential property path for one user (user-based) consists of a chronologically ordered list of all properties (e.g., *title*, *definition* etc.), which have been changed by that user, while a sequential property path for one class (class-based) consists of a chronologically ordered list of properties that were changed on that class.

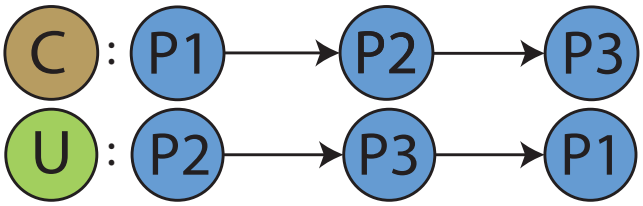


Fig. 2. The top row of the figure depicts an exemplary **class-based** sequential property path ($P1$ to $P3$) for class C. This means that for class C the property $P1$ was changed first, then property $P2$ and most recently changed was property $P3$. The bottom row of the figure depicts the sequential property path ($P1$ to $P3$), however this time for a user U (**user-based**). Analogously, user U has first changed $P2$, continued to change property $P3$ and most recently changed $P1$.

User-Sequence Paths: First, we analyze activity patterns within the collaborative ontology-engineering project. This means that we identify sequences of users who change a class. We want to investigate if and to what extent historic editing information (memory) can be used to predict the next user who is going to change a class. We will also inspect and describe the different sequential patterns (the structure) that can be extracted from the change-logs of the investigated collaborative ontology-engineering projects

Structural Paths: Analogously to the User-Sequence Paths, we investigate edit-strategies, such as *bottom-up* or *top-down* development, that users follow. Is it possible to predict at what depth level a user is going to contribute to the ontology next? In addition to development-strategies, we look at the relationships (e.g., parent, child, sibling etc.) between the current and the next class a user is going to contribute to.

Property Paths: On a content-based level, we investigate the series of property-changes users perform on. In particular, we want to know if we can model and predict the next property *a user* is going to change? Can we model and predict which property is going to be changed next for *a given class*?

3.3 Markov chain Model Selection

For the analysis conducted in this paper we are adopting the methodology presented by Singer et al. [21] and mapped to collaborative ontology-engineering change logs by Walk et al. [22] to determine if and to what extent sequential patterns can be identified in and extracted from change-logs and how well they perform in our prediction task.

For a better understanding of the collected results, we will provide a short description of Markov chains and the evaluation used to determine the appropriate order of a Markov chain to use. For an in-depth description of our methodology we point to Singer et al. [21], Walk et al. [22].

Among other fields of application, Markov chain models have been used for modeling navigation on the web. In general, a Markov chain consists of a finite *state-space* and the corresponding *transition probabilities* between these states. For our analysis, we will make use of the transition probabilities to identify likely transitions for a variety of different states. To be able to do so, it is important to understand the nature of Markov chains. Formally, a finite and discrete (in time and space) Markov chain can be seen as a stochastic process that contains a sequence of random variables. One of the most well known hypotheses about Markov chains is the so-called *Markovian assumption* that postulates that the next state of a sequence depends only on the current state and not on a sequence of preceding ones of the investigated data.

We are also interested in higher order Markov chains, meaning that in a k-th order Markov chain the next state depends on k previous ones. This means that given k previous changes of a user, we can predict, for example, what kind of property that user is most likely to change next.

For easier understanding, one could think of a first-order Markov chain model as a matrix, where each column and row correspond to a state of the *state-space* and the elements within the matrix represent the transition probabilities to and from each state towards the corresponding other states. For higher order Markov chain models, the states would include the combinations of all states, which drastically increases the state-space and therefore the complexity of the resulting Markov chain. When looking at the sequence of properties and a Markov chain of first order, we would calculate the transition probabilities between all properties. On a second-order Markov chain (and analogously higher-order Markov chains)

we would calculate the transition probabilities between all property pairs (as in 2 properties) and every single property. For example, what is the probability of a user changing *property c* next if that same user has last changed *property a* and then *property b*?

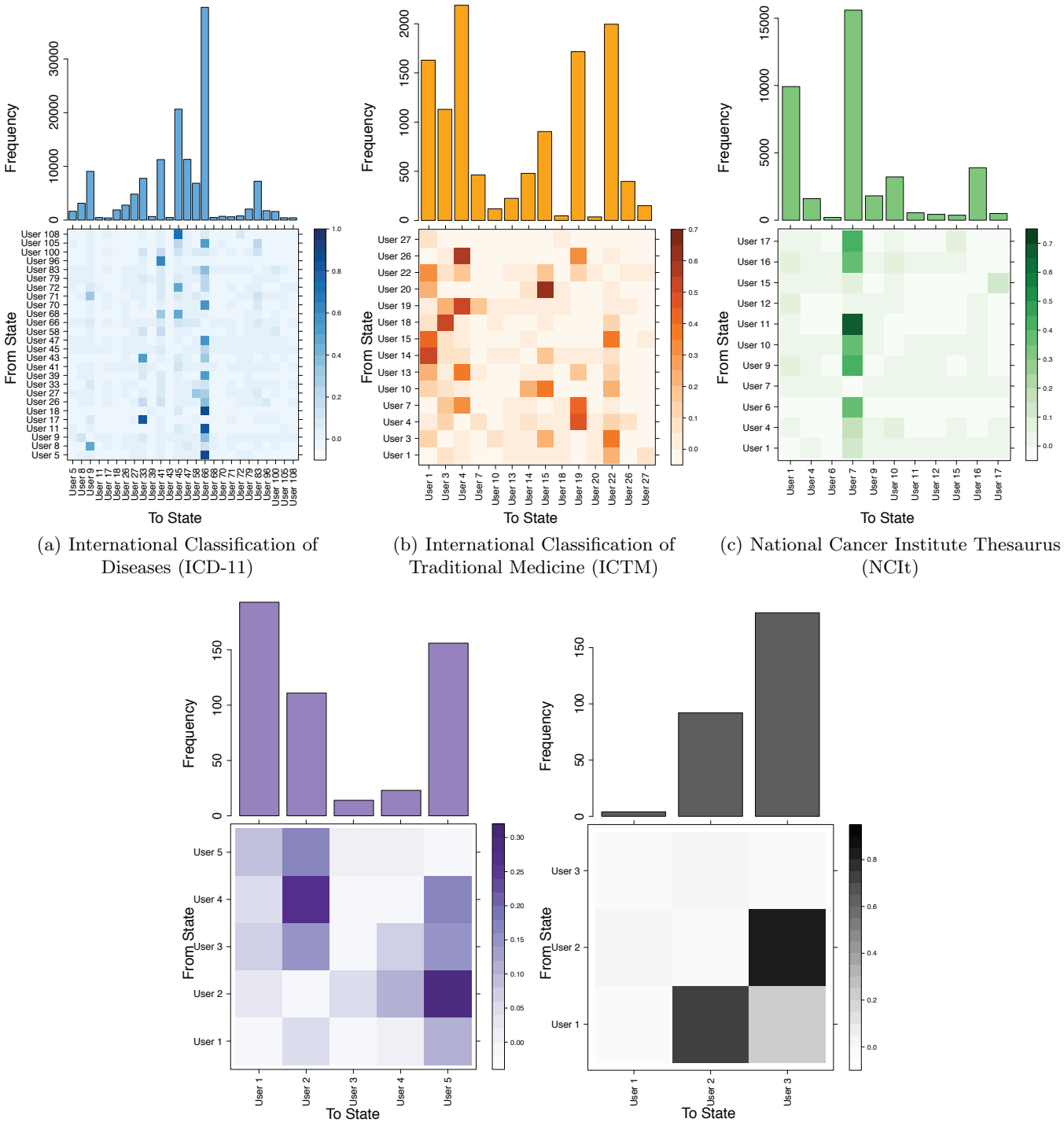
It is important to understand that if a Markov chain of at least first-order can be detected and put to practical use, transitions between the investigated states are not random, meaning that at least the current state holds information about the next state to occur.

4 Results

In this section, we report the results from our experiments on *User-Sequence Paths*, investigating if and to what extent it is possible to predict the user who is going to change a class next. In the *Structural Paths* section we analyze if and to what extent users move along the hierarchy of the ontology when contributing. In the *Property Paths* section we investigate if we can predict the property a user is going to change next as well as the property that is going to be changed next for a class.

4.1 User-Sequence Paths

In the *User-Sequence Paths* analysis we investigate whether we can predict the user who is going to contribute to a class next, given n previous contributors.



(a) International Classification of Diseases (ICD-11)

(b) International Classification of Traditional Medicine (ICTM)

(c) National Cancer Institute Thesaurus (NCIt)

(d) Biomedical Resource Ontology (BRO)

(e) Ontology for Parasite Lifecycle (OPL)

Fig. 3. Results for the *User-Sequence Paths* analysis: The columns and rows of the transition maps (**bottom area** of Figures 3(a) to 3(e)) represent the transition-probabilities between the users of each dataset for a first-order Markov chain, where rows are *source users* and columns are *target users*. A sequence (or transition-probability) is always read *from row to column*. Darker colors represent higher transition-probabilities while lighter colors indicate lesser transition-probabilities. Absolute probability values are dependent on the number of investigated rows and columns, hence relative differences are of greater importance. Darker colored columns identify gardeners. The histograms (**top area** of Figures 3(a) to 3(e)) show the amount of changes performed by each user (again for a first-order Markov chain) within the five ontologies in alphabetical order. Note, that the y-axes for all histograms are scaled differently for each dataset. All datasets have a few users who contributed the majority of changes, while the rest of the users (the long-tail) only contributed a very small number of changes. It is important to understand that the transition-probabilities depicted in the transition maps are relative numbers for each column and row individually. Thus, an inspection of the transition maps **and** histograms is necessary for proper interpretation. To increase readability we have removed users from the plots who have contributed only a very limited amount of changes for ICD-11, ICTM and NCIt.

Analyzing the chronologically ordered list of contributors for each class of the five investigated datasets provides the necessary information to identify users who perform changes on classes after (or before) other users. It is important to understand that this analysis on its own, without regarding additional factors, such as the changed property or the performed change-action, does not provide information about actual collaboration. The results of this analysis can be used to identify users who work on the same classes, however, we do not know if they actually collaborate with or just clean up (i.e., a *gardener*) after other users.

Path & State Descriptions: To analyze user sequences, we iterated over each class of our datasets and extracted a chronologically ordered list of contributors. As we are interested in uncovering the transitions between users, we merged multiple consecutive changes by the same user into a single change to avoid the detection of longer Markov chains based on frequent and consecutive loops between the same states.

As the name of this analysis suggests, the states of the Markov chain correspond to the usernames of the users who contributed to each project. Due to reasons of privacy we obfuscated the actual usernames and replaced them with generic names.

Results: When investigating the transition probabilities (representing a Markov chain of first order) between contributors (see bottom area of Figures 3(a) to 3(e)) we can identify very active users by looking at darker colored columns of the transition maps. As we have merged all consecutive changes of the same user into one single change the diagonal, representing the transition probabilities between the same users, is 0. Note, that the absolute transition probabilities, depicted next to each transition map, are dependent on the absolute amount of observations and states, thus are to be interpreted relatively to each other for each row individually. When looking at the probabilities between very active users and all target states in ICD-11 we can see that the probabilities are very evenly distributed across all users. Nonetheless, we can clearly identify *User 66* to be the most likely following state after nearly all other states. This suggests, that *User 66* may represent a gardener in ICD-11.

For NCIt we can clearly observe that *User 7* appears to be a *gardener*, who is checking all the changes contributed by all other users. For BRO *Users 2* and *5* are prominent target states, evident in the high transition probabilities as *To State* (dark columns). Interestingly, the user with the highest amount of changes (*User 1*) exhibits very low and evenly distributed transition probabilities (row) and is not necessarily the user that most likely changes a class after other users. This shows us that there does not need to be a necessary connection between the overall activity of users and their activity as a gardener. For OPL we can observe that *User 3* frequently changes the same classes after *User 2*. A similar observation can be made for *Users 1* and *2*. However, one has to keep in mind that *User 1* has contributed a limited number of changes, rendering the observed transition probabilities less useful as they rarely occur.

The histograms (see top area of Figures 3(a) to 3(e)) indicate that a small number of users contribute the majority of changes (similar to a long-tail distri-

bution). This distribution can be seen most dominantly for ICD-11 and NCIt. A larger share of users contributed the majority of changes for ICTM, BRO and OPL, which is also connected to a larger number of gardeners in these ontologies.

Interpretation & Practical Implications: The transition probabilities for a first-order Markov chain unveil the roles of certain users and can help to identify users or even groups of users who frequently change the same classes. Users that are prominent target states (i.e., exhibit high transition probabilities in their columns) were identified by us as actual gardeners, curators and administrators of the corresponding projects. If certain users always change the same classes after specific other users, it could be worthwhile for project administrators to investigate if these users are actually collaborating, for example by looking at the changed properties and property values, or if one user is always cleaning up after the other user. In all datasets we were able to observe at least one user who contributed a high amount of changes, with evenly distributed transition probabilities to all remaining users. This observation indicates that in all projects, gardeners, curators and administrators are assigned (directly or indirectly) certain parts of the ontology; otherwise the transition probabilities between the very active users would be higher.

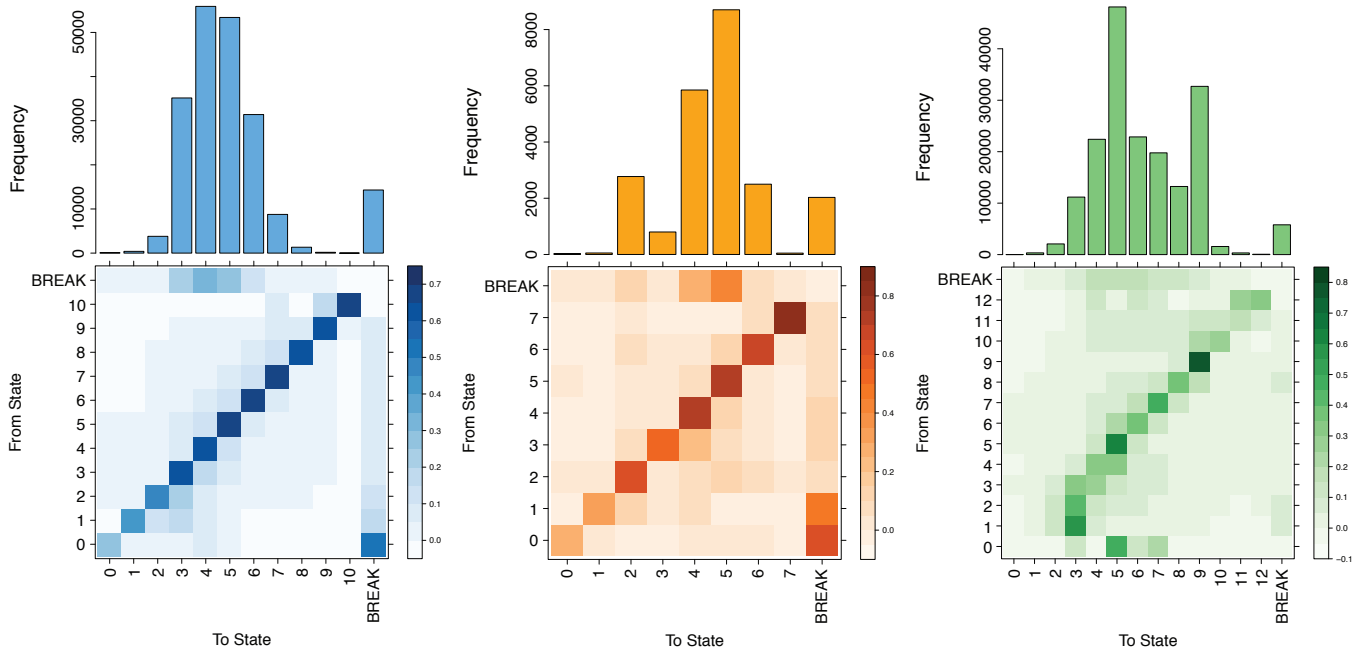
The ability of predicting who is going to change a specific class next, as well as the classes that a user is most likely to change next can be used by project administrators to help users finding and identifying classes (and thus work) of interest. Classes for users can be easily identified by using the calculated Markov chain probabilities, and instead of just predicting the next user for a class, we use the information of the most likely next contributor to infer the classes that user is most likely to change next. Note, that using this approach we are not guaranteed to find classes for all users. On the other hand the information about the next, most probable contributor for a class, can even be used to create automatic class recommender systems to suggest work to users, which could help to increase participation. In particular for projects the size of ICD-11 and NCIt, mechanisms to automatically identify and assign work are highly useful as it is still very time-consuming to identify pending work and suitable users with the necessary knowledge to address the identified work-tasks.

4.2 Structural Paths

The investigation of *Structural Paths* involves an analysis of different aspects regarding how and where users contribute to the ontology, such as the depth level of the class that users contribute to next (Section 4.2) as well as looking at the relationship distances between consecutively changed classes (Section 4.2).

Depth-Level Paths In this analysis, we investigate if users concentrate their efforts on specific depth levels of the ontology and if there are certain depth levels that receive frequent transitions and less concentrated workflows. The gathered results can be used to implement pre-fetching mechanisms to minimize the loading and waiting times for contributors. Furthermore, we can determine

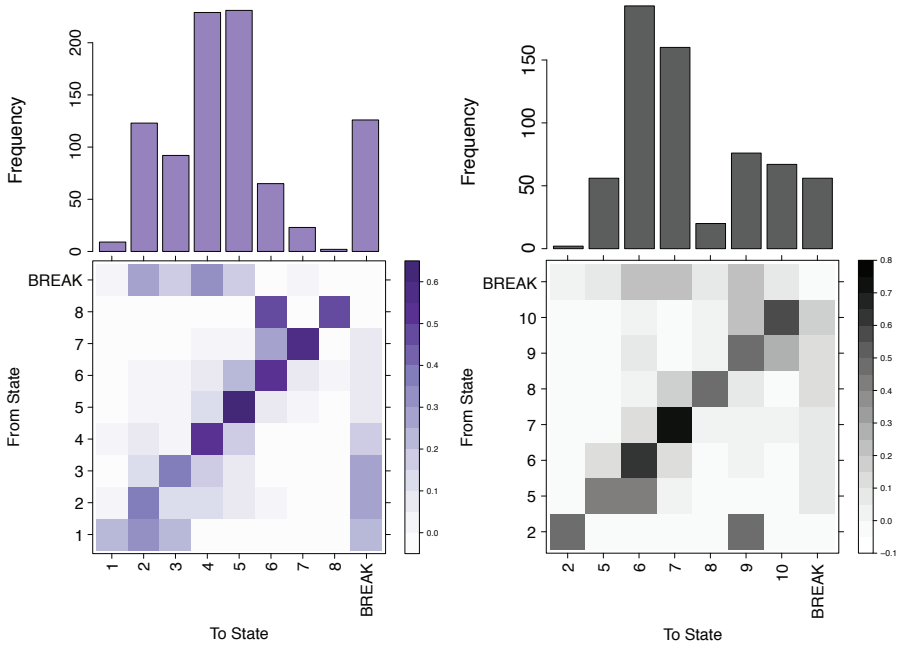
whether users move along the structure of the underlying ontology when editing classes.



(a) International Classification of Diseases (ICD-11)

(b) International Classification of Traditional Medicine (ICTM)

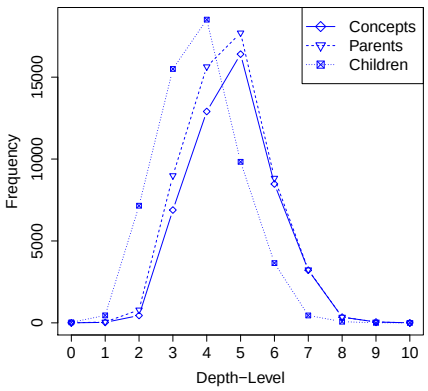
(c) National Cancer Institute Thesaurus (NCIt)



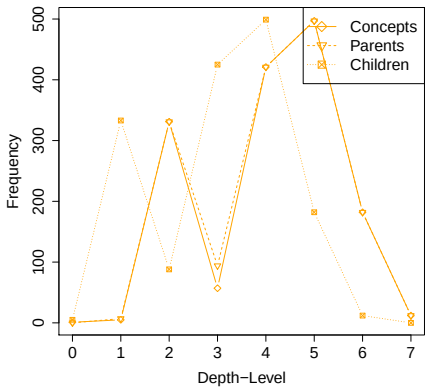
(d) Biomedical Resource Ontology (BRO)

(e) Ontology for Parasite Lifecycle (OPL)

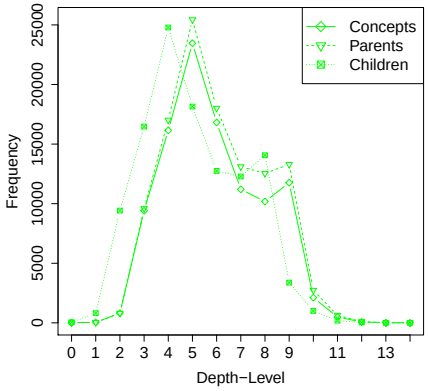
Fig. 4. Results for the *Depth-Level Paths* analysis: The columns and rows of the transition maps (**bottom area** of Figures 4(a) to 4(e)) represent the transition probabilities of a first-order Markov chain between depth levels, where rows are *source depth levels* and columns are *target depth levels*. A sequence (or transition probability) is always read *from row to column*. Darker colors represent higher transition probabilities while lighter colors indicate lesser transition-probabilities. Absolute probability values are dependent on the number of investigated rows and columns, hence relative differences are of greater importance. For classes closer to root a *top-down* editing manner can be observed, while this is reversed for classes further away from root. The histograms (**top area** of Figures 4(a) to 4(e)) show the amount of changes performed in each depth level aggregated over all users of the respective projects (again for a first-order Markov chain). Throughout all projects, classes located between the first and last few depth levels (in the middle) are changed substantially more frequently than others, suggesting that work is concentrated on some depth levels while others receive none to very few changes at all. Note, that the y-axes for all histograms are scaled differently for each dataset. For the x-axes (and column/rows of the transition maps) we only display depth levels which exhibit at least one change, thus, the depth level sequences are not necessarily continuous from lowest to highest depth level.



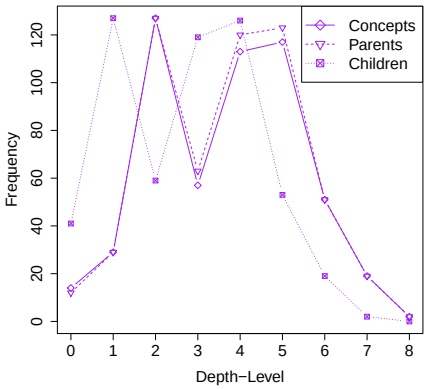
(f) International Classification of Diseases (ICD-11)

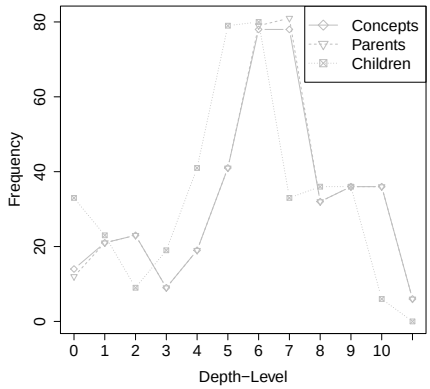


(g) International Classification of Traditional Medicine (ICTM)



(h) National Cancer Institute Thesaurus (NCIt)





(i) Biomedical Resource Ontology (BRO) (j) Ontology for Parasite Lifecycle (OPL)

Fig. 5. The **Figures 5(f) to 5(j)** depict the absolute numbers (y-axis; Frequency) of classes as well as the number of edges (*isA*) to classes on the immediate higher (*parents*; closer to root) and lower (*children*; further away from root) depth level for all depth levels (x-axis; Depth-Level). According to Figures 5(f) to 5(j) the transition probabilities depicted in the transition maps correlate with the total number of edges to children and parents for each depth level across all datasets.

Path Extraction & State Descriptions: For this analysis, we stored the chronologically ordered depth levels of each changed class for each user. The depth level of a class is the length of the shortest path between the *root node* of the ontology and the corresponding class. We merged consecutive changes that were conducted by the same user on the same class into one single transition between the same depth levels. This approach helps us to investigate actual transitions between states while still retaining the notion of users consecutively editing the same classes.

Results: First, the histograms (see top area of Figures 4(a) to 4(e)) show that work is concentrated on certain levels of the ontology, with the highest and lowest levels not receiving as much attention as the levels in-between.

As depicted in the transition maps (bottom area of Figures 4(a) to 4(e)), users have a high tendency to edit classes in the same depth levels, visible in the darker colored diagonal. In ICD-11, for the first 5 depth levels, users appear to have a tendency towards *top-down* editing, while this tendency turns around at a depth level of 6 and higher, and appears to be strictly limited to surrounding depth levels. For ICTM (see Figure 4(b)), we can observe a similar trend, again with the tendency towards *top-down* editing appearing to be minimally more dominant. For NCIt, when only looking at the transition map, we can identify a trend towards *bottom-up* editing. However, when we consider the absolute amount of changes, depicted in 4(c), we can see that the levels with higher transition probabilities to lower depth levels (further away from root) occur more frequently. Thus, when users are not changing the same classes, they still exhibit a preference towards *top-down* editing. Given the short observation periods for BRO and OPL it is hard to infer actual edit strategies. However, similar to the other projects, we can observe a concentration on the same depth levels with alternating preferences towards higher and lower depth levels. Similar to ICD-11, all datasets exhibit higher transition probabilities between the immediately surrounding depth levels.

Furthermore, we investigate whether the total number of classes as well as the total number of links to the immediate higher (children; edges to classes one level further away from root) and lower (parents; edges to classes one level closer to root) depth level correlate with our findings (Figures 5(f) to 5(j)). For example, the transition map for ICD-11 (see Figure 4(a)) shows that contributors exhibit a *top-down* editing behavior for the first five depth levels, with level 5 exhibiting first signs of *bottom-up* editing. Figure 5(f) shows a higher number of possible transitions from children than parents, indicating that users are in general likelier to follow *top-down* editing-strategies when changing classes, following relationships by chance, of the first four levels. This changes for ICD-11 at level 5, with a higher number of transitions to parents than to children, and continues until level 10. Resulting in a higher probability of users performing *bottom-up* editing-strategies when changing classes from levels 6 to 10. The same observations can be made for all other datasets, indicating that the class hierarchy influences the edit behavior of contributors.

In all datasets, after taking a *BREAK* (representing an artificially introduced session break when two consecutive changes of the same user are more than 5 minutes apart; for more information see section 5.4), users exhibit a clear tendency towards changing classes on certain depth levels (e.g., levels 3 to 5 for ICD-11, levels 4 to 5 for ICTM, levels 4 to 7 for NCIt, levels 2 to 4 for BRO and levels 6 to 9 for OPL).

Interpretation & Practical Implications: The results of this analysis show if, to what extent and where work is conducted and concentrated within the ontology. This information can be used in a variety of ways, for example by ontology-engineering tool developers to pre-fetch potential classes a user is likely to work on next, or to adapt the interface of the ontology-engineering tool dynamically to display specific classes after users return from a *BREAK*. Project managers can adapt milestones and project progress reports to reflect the underlying editing strategies (e.g., *top-down* editing), for example by aligning progress with created branches (opposed to complete coverage). Across all projects we can observe that classes close to and very far away from the *root* of the ontology are not edited as frequently as other classes. One explanation for this observation could be that classes in lower depth levels (closer to *root*) are mainly used as content dividers and are usually created in the beginning of a project. Thus, they may be more stable and less frequently updated. Classes at the higher depth levels (further away from *root*) on the other hand most likely require extensive expert knowledge. Hence, only a small number of users have the necessary expertise to contribute to these classes. Additionally, the absolute number of classes in the higher and lower depth levels is much lower in all investigated datasets. Note that absolute values of depth levels are less important for the interpretation of the results, rather than their relative position (i.e., closest to root, furthest away from root etc.). For example, a class at level 6 can exhibit different behaviors in ontologies with 6 or 10 levels.

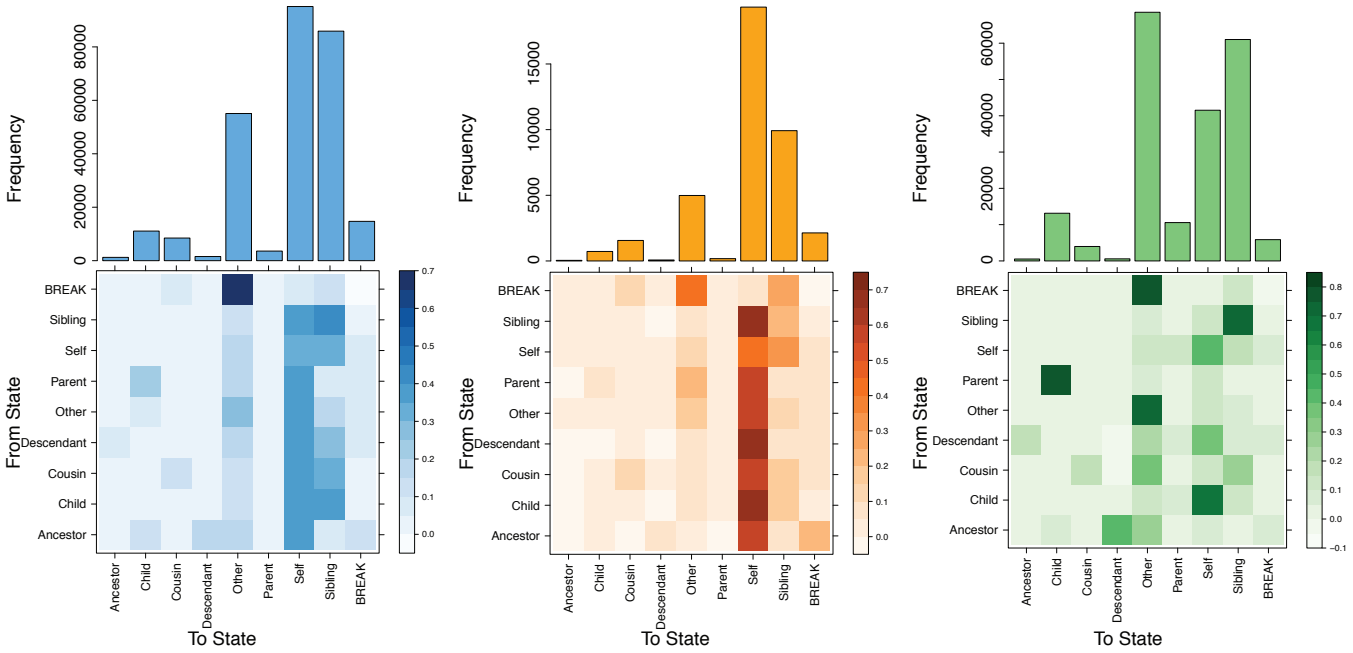
In all projects, except for NCIt, the depth levels where users start to edit the ontology after they return from a *BREAK* are similar to the ones where they stop editing before taking a *BREAK*. To be able to make that observation we have to take the absolute numbers of changes on each depth level (bottom area of Figure 4) into account when looking at the transition probabilities (top area of Figure 4). NCIt is the only dataset where users appear to be similarly likely to take a *BREAK* after changing classes across all depth levels, except for 0 and 12.

When we combine the results of this analysis with the results of the *User-Sequence Paths* (Section 4.1) we can develop automatic mechanisms to curate and delegate work to users. For example, if we know that a specific user is most probably going to contribute to a class on level 3 and we have a set of classes on that level where that specific user is the most probable next user to contribute to, determined by the *User-Sequence Paths* analysis, we can combine these two observations to create class (and thus work) suggestions for users.

Relative Relationship Paths Given the high number of observed transitions between the same depth levels in the *Depth-Level Paths* analyses (Section 4.2; bottom area of Figure 4), we conducted an additional analysis investigating the shortest-paths and relationships between the changed classes for all users. To further strengthen our observation that users are actually moving along the ontological hierarchy when contributing to an ontology, we analyzed the actual relative relationships between the changed classes for each user. For example, when traversing the shortest-path distance of 2, multiple different nodes can be reached, such as a grandparent (i.e., 2 times up), a grandchild (i.e., 2 times down), a sibling (i.e., 1 time up, 1 time down) or even some other relationship (e.g., 1 time down, 1 time up).

Path Extraction & State Descriptions: By combining the information from the *Depth-Level Paths* and the relative movement between depth levels, we inferred the relative relationships between two consecutively changed classes. For example, if the difference between the depth levels of the investigated classes would be exactly the size of the shortest-path between them (with the shortest-path being > 0), the latter-changed class could either be a *Child*, a *Parent*, an *Ancestor* or a *Descendent* of the first-changed class. Given a relative *DOWN* movement (to a lower depth level) value, depending on the shortest-path value, the second class could be classified as *Child* (shortest-path of 1) or *Descendent* (shortest-path > 1). Analogously follows the definition of a *Parent* and *Ancestor* with a relative *UP* movement. A *Sibling* is defined as the two classes being (i) connected via the same parent with (ii) a shortest-path distance of 2 and (iii) both classes are located on the *SAME* depth level. The *Cousin* state is defined as two classes on the *SAME* depth level being connected by the same grand parent while exhibiting a shortest-path distance of 4. Every other possible combination of depth level and shortest-path was classified as *Other*. The state *Self* indicates that the same class that was changed last time was changed again, thus, no transition to another class occurred. For example, a transition from *Sibling* to *Self* means that a change was first performed on a class that is a sibling of the previous class (not displayed in this example) and then another change was performed on the same class, however now the relationship changed to *Self* as no new class was involved.

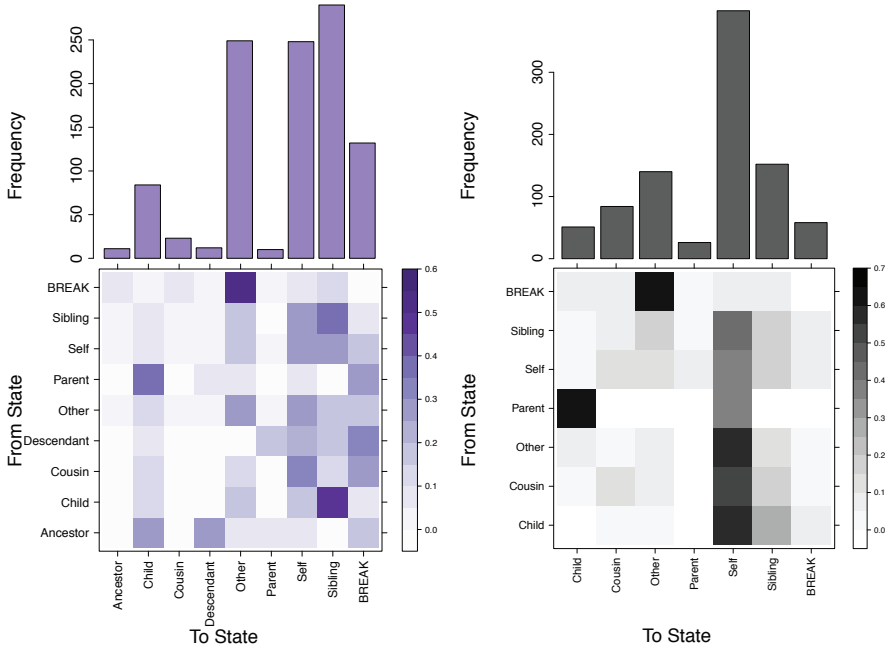
Again, consecutive changes on the same class by the same user have been merged into one *self-loop*, meaning that multiple (more than 2) consecutive transitions of the same user on the same class have been merged into one transition from the state *Self* to *Self*.



(a) International Classification of Diseases (ICD-11)

(b) International Classification of Traditional Medicine (ICTM)

(c) National Cancer Institute Thesaurus (NCIt)



(d) Biomedical Resource Ontology (BRO)

(e) Ontology for Parasite Lifecycle (OPL)

Fig. 6. Results for the *Relative-Relationship Paths* analysis: The columns and rows of the transition maps (**bottom area** of Figures 6(a) to 6(e)) represent the transition-probabilities of a first-order Markov chain between relative-relationship levels, where rows are *source relationships* and columns are *target relationships*. A sequence (or transition-probability) is always read *from row to column*. Darker colors represent higher transition-probabilities while lighter colors indicate lesser transition-probabilities. Absolute probability values are dependent on the number of investigated rows and columns, hence relative differences are of greater importance. Across all datasets, aside from *Self*, a very clear trend towards editing the ontology along *Siblings* can be observed. The histograms (**top area** of Figures 6(a) to 6(e)) show the total number of occurrences of each relationship in the corresponding datasets aggregated over all users (again for a first-order Markov chain). Note, that the y-axes for all histograms are scaled differently for each dataset. For the x-axes (and column/rows of the transition maps) we only relationships that occur at least once in the corresponding paths, thus the x-axes could be different from project to project. Given the very high amount of *Self* and *Sibling* transitions we can concur that users, when they contribute to classes on the same depth level follow a *breadth-first* strategy.

Results: When looking at the histograms (see top area of Figures 6(a) to 6(e)), we can observe that the states *Self*, *Sibling* and *Other* are highly represented across all datasets. The transition maps (bottom area of Figures 6(a) to 6(e)) show that after a *BREAK*, across all five datasets, users tend to change classes "somewhere else" in the ontology, evident in the high transition probability from *BREAK* towards *Other*, and are likely not to resume work in the same area of the ontology that they stopped working on. For ICD-11, ICTM and OPL, no matter which relationship type occurs, users tend to edit the same class consecutively (dark colors in the *Self* column). From this *Self* state, which is also the state that occurs the most often in ICD-11, ICTM and OPL, users are very likely either to change the same class again (*Self*) or to change a *Sibling* of the current class.

For NCIt, BRO and OPL we can observe that users, when changing a *Parent* are very likely to change a *Child* of that parent afterwards. Note, that this *Child* does not necessarily have to be the same class that was changed prior to the traversal to *Parent*. In all datasets, except for OPL, very high transition probabilities towards *Other* can be observed for all not so frequently used states. In particular for NCIt we can observe that *Other* is the most frequently observed transition, even before *Self* and *Sibling*.

Interpretation & Practical Implications: By combining the results of this analysis with the results of the *Depth-Level Paths* analysis, we can infer that users exhibit a tendency towards *top-down* editing while contributing to the ontology, when only considering changes that occur on different depth levels. If they concentrate their efforts on the same depth levels, users exhibit a *breadth-first* editing behavior either changing the same class multiple times or traversing along siblings of the current class. We can leverage this information not only to refine the previously suggested pre-fetching of classes but also to enhance possible class recommendations. Similarly, it is possible for ontology-engineering tool developers to minimize the necessary efforts of users to contribute to the ontology by implementing, for example, guided workflows that take the underlying edit strategies of the contributors into account.

As classes in ICD-11 and ICTM have a large number of properties and for ICTM certain properties have to be added in multiple languages, the high transition probabilities towards *Self* (dark colors in the *Self* column) are not surprising. One possible explanation for this observation for ICD-11 could be the special functionality available in iCAT (for ICD-11) that allows users to export parts of the ontology as spreadsheets for local editing and adding property values. Once contributors finished editing the spreadsheet they have to enter the data into the system manually, as no automatic import functionality is present. In the iCAT interface, users are simultaneously presented with the ontology tree for navigating through the classes and the corresponding properties and property values. When users select a property they can easily switch between classes, with the selected property staying selected, thus allowing to quickly enter the same properties for different classes.

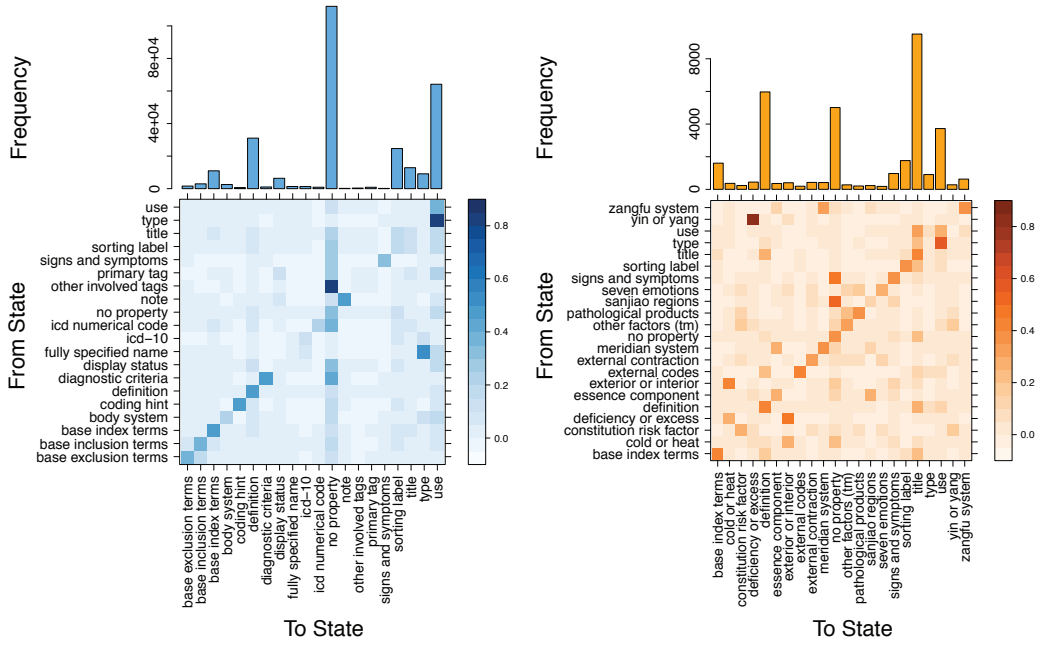
A similar, yet not as dominant as in ICD-11 and ICTM, behavior can be observed for NCIt and BRO and even to some extent in OPL, which all do not use the export functionality. According to our observations, users travel along the underlying hierarchy when contributing to the ontology. Given the observations made for ICD-11 this behavior can be enforced by providing certain functionalities in the user-interface especially when they compliment the workflows of the contributors.

The results of this analysis have also shown that users are likely to pursue a certain strategy or intermediate goal for their edit sessions, for example changing all classes in a specific (narrow) area of the ontology. This is evident in the observation that after returning from a *BREAK*, users have a very high tendency to change the ontology "somewhere else" (see the transition probabilities from *BREAK* towards *Other* in the top-row of Figure 6), rather than picking up the work, where they left off. This discovery is very important for developing class-recommender, as we can use the results of this analysis to suggest closely related classes to the current class a user is working on, however when that user stays inactive for the duration defined for introducing *BREAK*s the recommendation strategy has to be changed.

4.3 Property Paths

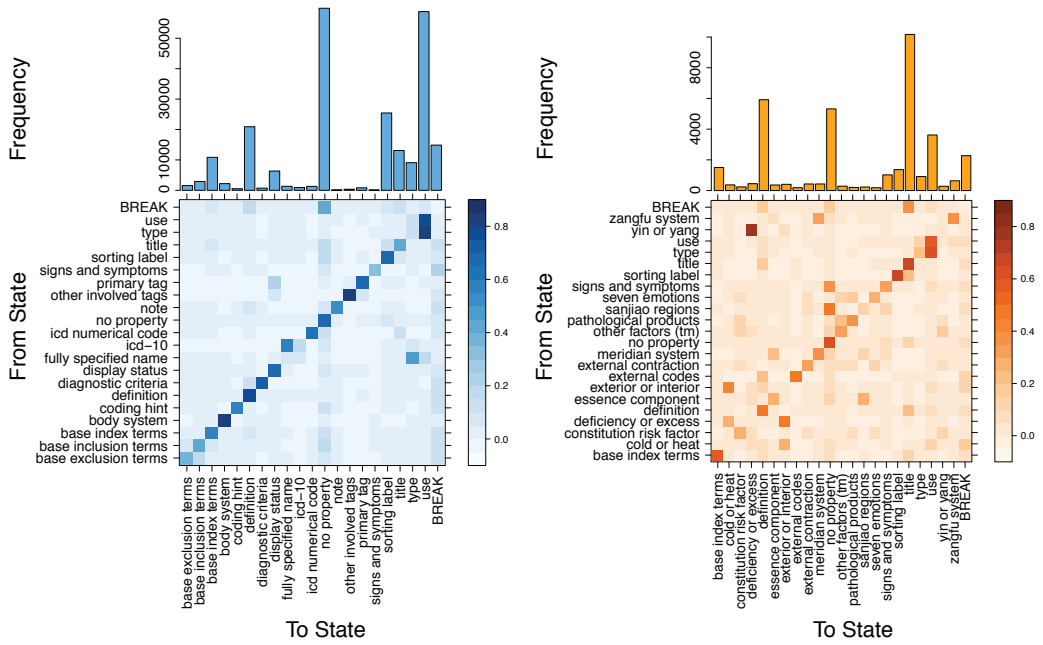
Aside from analyzing different aspects of activity (Section 4.1) and the correlation between contribution patterns and the structure of an ontology (Section 4.2), we can use Markov chains to perform an analysis on the properties used in the ontology. We were not able to perform the *Property Paths* analysis on OPL and BRO as these datasets contain only a very limited number of unique property changes during our observation periods. We also had to discard the results from NCIt, as the ontology-editing environment for NCIt provides a unique change-queuing mechanism that allows for multiple properties to be changed at the same time, making it impossible to extract chronologically ordered sequential property patterns. To compensate for the missing datasets, we will use this section of the paper to show that the Markov chain analysis can provide insights for the same state space when assuming different viewing angles for the observations (or transitions) of ICD-11 and ICTM. In particular, we want to know if it is possible to predict who is going to change which property next and which property is changed next for a given class.

All properties which where rarely edited have been removed from Figure 7 as they do not hold information but their removal increased the readability of the plots dramatically.



(a) International Classification of Diseases (ICD-11) (Class)

(b) International Classification of Traditional Medicine (ICTM) (Class)



(c) International Classification of Diseases (ICD-11) (User)

(d) International Classification of Traditional Medicine (ICTM) (User)

Fig. 7. Results for the *Property Paths* analysis: The columns and rows of the transition maps (**bottom area** of Figures 7(a) to 7(d)) represent the transition-probabilities of a first-order Markov chain between consecutively changed properties, where rows are *source properties* and columns are *target properties*. A sequence (or transition-probability) is always read *from row to column*. Darker colors represent higher transition-probabilities while lighter colors indicate lesser transition-probabilities. Absolute probability values are dependent on the number of investigated rows and columns, hence relative differences are of greater importance. Across all datasets, aside from *Self*, a very clear trend towards consecutively editing the same properties can be observed. The histograms (**top area** of Figures 7(a) to 7(d)) show the total edits of each property in the corresponding datasets aggregated over all users and classes (again for a first-order Markov chain). Note, that the y-axes for all histograms are scaled differently for each dataset. As ICTM and ICD-11 only share a limited amount of properties the x-axes (and column/rows of the transition maps) are different from project to project. In both projects and across all 4 different approaches the *title*, *definition* and *use* properties are frequently used. Due to reasons of readability we were forced to remove properties from the plots, which exhibited only a very limited amount of changes, thus did not provide substantial information for the purpose of this analysis.

Path Extraction & State Descriptions: First, we extracted the properties that were changed in ICD-11 and ICTM, sorted either by user and timestamp or by class and timestamp. Finally, two chronologically ordered property lists were extracted, one ordered per user and one ordered per class (for both datasets). The states used for the *Property Paths* analyses represent the different properties, which can be assigned a value for each class in ICD-11 and ICTM. Whenever a change did not change a property (e.g., because the change action dealt with moving or creating a class) we used the *no property* state. Similar to previous analyses, if the same user has consecutively changed the same property (e.g., *title*) on the same class, we merged these changes into one transition from said property to said property (e.g., one transition from *title* to *title*). Analogously, however without the restriction of the same user, if the same property was changed on the same class, we merged these changes into one *self-loop*.

Results: When looking at the histograms (top area in Figures 7(a) to 7(d)) we can see that even after removing not very frequently used properties, both datasets exhibit few properties which have received a high number of changes, while the remaining majority of properties only received a very limited amount of changes. For both datasets, aside from the *no property* state, *use*, *title* and *definition* appear to be the most frequently used properties. As can be seen in the top area of Figures 7(a) and 7(b), multiple consecutive changes of the same property by different users (or only two changes by one user at all) appear to be fairly common for both datasets. In contrast, when looking at Figures 7(c) and 7(d), which depict the transition probabilities between the sequences of properties changed by each user, we can see an even stronger trend towards consecutively changing the same properties across different classes, especially *definition*, *title* and *use*. Even though there is a stronger emphasis on consecutively changing the same property multiple times, the overall transition probabilities are slightly more diverse for the user-based analysis than for the class-based analysis for both datasets. For ICD-11 Figures 7(a) and 7(c) show that the class-based approach is less focused on *self-loops*, evident in the brighter diagonal, when compared to the user-based approach. This is due to the export functionality available in iCAT combined with the manual process of inserting the same property for different classes by users of ICD-11. In contrast, such functionality is absent in ICTM, thus leading to similar behaviors for the class and user-based approaches for ICTM. The high concentration on *self-loops* in both approaches analyzed for ICTM could also be due to the multilingual nature of the project, meaning that certain properties, such as *title* and *definition*, have to be entered multiple times in multiple languages. Similar results have been presented by Wang et al. [18], who used association rule mining techniques to analyze the change-logs of ICD-11 and ICTM.

Contributors in ICD-11 have a high tendency of performing *no property* changes after they return from a *BREAK* followed by *use*, *title* and *definition*. In ICTM, users resume their work primarily by changing the *title* property, the *definition* property followed by *no property* changes.

Interpretation & Practical Implications: One of the main benefits of this analysis is the identification of commonly and consecutively changed properties for classes and users. This information can be used to suggest work (e.g., prompting a user to check a certain property by combining the *User-Sequence Paths* analysis and the *Property Paths* analysis), or by ontology-engineering tool developers to anticipate the property a user is most likely to change next. The fact that classes appear to exhibit more diverse property-contribution patterns when being changed than users could be a direct result of the multi-lingual nature of ICTM and the already mentioned export functionality present in iCAT. This means that given the most recent property of a class that was edited, we can predict which property is most likely to be changed next. Similarly, we can predict the property a user is going to edit next.

Table 2. A summary of all findings applicable to all investigated biomedical ontologies. All listed findings are discussed in more detail in section 5.

User-Sequence Paths (cf. section 4.1)	**Users work in micro-workflows**	Information about who has edited classes in the past contains predictive information about who is going to change a class next.
	User-roles can be identified	Looking at historic data, we can identify different user roles, i.e., administrators and moderators, gardeners and users that frequently interact with (collaborate/revert) each other.
Structural Paths (cf. section 4.2)	**Users edit behavior is influenced by the class hierarchy**	Contributors, when adding content to the ontology, are influenced by the class hierarchy.
	Users edit the ontology top-down and breadth-first	By and large, users exhibit a minor tendency towards top-down editing behavior when changing hierarchy levels while contributing. However, when staying in the same hierarchy level, contributors rather follow a *breadth-first* edit behavior, moving from one sibling of a class to the next sibling.
	Users edit closely related classes	Contributors have a very high tendency to consecutively change closely related classes, as opposed to randomly and distantly related classes.
Property Paths (cf. section 4.3)	**Users perform property-based workflows**	Contributors, when adding content to the ontology, tend to concentrate their efforts on one single property, which is added and edited for multiple classes.

5 Findings and Discussion

In this section we first report results from our evaluation of higher order Markov chain models in section 5.1 before we summarize our findings in section 5.2. Next, we discuss differences between the investigated projects in section 5.3 and finally, point out potential limitations of this work in section 5.4.

5.1 Evaluation of higher Markov chain orders

Until now, we have only reported results and visualizations using first order Markov chain models due to visualization tractability. However, as already postulated in section 3.3, higher order Markov chain models may be more appropriate for modeling the available sequential paths. This would mean that the next state in a sequence (e.g., the next property a user modifies) can be best predicted by not only looking at the current one, but by looking at a sequence of preceding properties the user has changed. Hence, we now also tackle the problem of identifying the appropriate Markov chain order for our data.

Table 3 summarizes the results of our evaluation task. For each analysis the rows marked with *Evaluation* represent the suggested Markov chain orders that performed best in our prediction task. The rows marked with *Suggested* correspond to the manually selected Markov chain orders, which provide the best balance between the complexity of the Markov chain and the performance in the evaluation task. More details on the results of the evaluation can be found in A.

For all of our findings, preprocessing the change logs by merging multiple consecutive occurrences of certain states into one single action or one *self-loop* biases the detection of higher-order Markov chains. Without this preprocessing, we are more likely to obtain higher-order Markov chains, and thus better results for the conducted prediction tasks. However, this would also mean that the resulting Markov chain would be biased towards *self-loops* (higher transition probabilities in the merged states for each analysis in the transition maps), which do not provide further actionable information and potentially mask other transition probabilities of the transition maps.

Despite this restriction we were able to extract multiple first- and higher-order Markov chains, meaning that for several analyses users' actions depend on a single or a series of preceding actions. As shown in Table 3, we have suggested the use of a first-order Markov chain for the majority of our analyses even though the conducted evaluation prediction task determined a higher-order Markov chain to produce the best average position. Our suggested Markov chain orders for productive use represent a manually selected trade-off between performance and complexity of the Markov chain model (i.e., a manual Occam's razor).

Given the obtained differences in Markov chain orders (see *Evaluation* rows in Table 3) we can see that for OPL and BRO, representing smaller collaborative ontology-engineering datasets, the Markovian principle appears to hold. This means that in BRO and OPL, users's actions depend on one single preceding action. In contrast, the larger datasets, being ICD-11, ICTM and NCIt, exhibit

Markov chains up to an order of three, suggesting that the Markovian principle is violated and the next action to occur in the system can be predicted best by looking at a series of up to three previous actions.

For all datasets and all experiments, we were able to show that a first-order Markov chain performed (at least minimally) better than randomly selecting the next state in our evaluation task. As depicted in Table 3, users, when collaboratively creating and editing an ontology, exhibit strong tendencies towards a memoryless editing behavior (first-order Markov chain) but simultaneously exhibit memory influenced editing behavior (second-order Markov chain or higher) as well. Hence, we can say that the Markovian principle does not hold for all aspects of the sequential pattern analysis of collaborative ontology-engineering change-logs.

By and large, this confirms that looking at first order Markov chain orders already allows us to get thorough insights into the dynamics of sequential user actions in collaboratively-engineered ontologies as we did in section 4. However, this also suggests that looking at a larger history of actions might even further augment our understanding of these dynamics and might also clearly improve our ability of predicting user actions.

5.2 Summary of findings

We will now discuss our main findings (Table 2) and explore their consequences.

Emergence of micro-workflows: By investigating whether sequential user-contribution patterns (see section 4.1) can be identified in five different collaborative ontology-engineering projects, we have shown that users appear to work in micro-workflows, indicating that for all investigated projects, each user contains predictive information (the extracted Markov chain order) about the user, who is going to contribute to a specific class next.

Additionally, however not presented in this paper due to reasons of space, we have also conducted an analysis to determine the change type (e.g., adding a property value, moving a class, replacing a property value etc.) a user is most likely to perform next (as shown in Walk et al. [22] for ICD-11). In this analysis we were able to extract a first-order Markov chain for all datasets presented in this paper, meaning that the last change type that a user performed contains information about the next change type of that user. When combining the information about the user who is most likely to contribute to a class next and the specific change action that this user is most likely to conduct (or the change action that is most likely conducted on a class next), we can create specific tasks for contributors, asking them to perform a certain change on a specific class.

Our results can be used by project managers and ontology-engineering tool developers to identify classes for users and users for classes, helping editors to minimize the necessary efforts for finding and identifying classes to contribute to. Moreover, automatic means of curating and delegating work-tasks to users can be derived by ontology-engineering tool developers, which can help to potentially increase participation as discussed in Kittur and Kraut [23].

The conducted cross-fold evaluation suggests that for all datasets a first- or second-order Markov chain model yields the best results in our prediction task. This means that given the (two) last user(s) who changed a class, we can predict the user who is most likely to change the class next.

User roles can be identified: Across all datasets we were able to identify that a limited number of users have contributed to the majority of all changes. These highly active users are very likely to be *target users* for all other users, meaning that they are very likely to change the same class after another user. Across all five datasets, the roles of these *target users* could be identified by us as moderators or administrators of the corresponding projects performing maintenance tasks, such as gardening (e.g., pruning outdated classes, fixing errors etc.) or manual verification of newly added data.

Furthermore, we were able to show that moderators and administrators divide work among each other, as they are not very likely to change the same classes directly after another administrator or moderator, even though these users exhibit the highest absolute numbers of changes in the corresponding projects. Looking at the transition probabilities of Figure 3 it is possible to identify users or even groups of users who have a high tendency to work on the same classes, thus might be collaborators or reverting/correcting changes of each other.

Users edit the ontology top-down and breadth-first: The *Depth-Level Paths* analysis (see section 4.2) demonstrated that users have a very high tendency of staying in the same depth level when contributing to the ontology. If editors change depth levels while editing the ontology they exhibit a minimal preference to do so in a *top-down* rather than a *bottom-up* manner. Furthermore, the results suggest that users move along the hierarchy as we were able to show that they follow a *top-down* editing strategy for classes that are closer to the root node while this changes to a *bottom-up* editing strategy for classes closer to the deepest depth levels and transitions are more likely to occur along the immediate higher or lower depth level.

Users edit the ontology top-down and breadth-first: To further investigate the distances between changed classes at the same depth levels we investigated the *Relative Relationship Paths* (e.g., child, parent, sibling, cousin etc.) between these changed classes. We found that users, when they edit classes on the same depth level, follow a *breadth-first* manner, focusing on editing all the siblings of a class before switching to a completely different area of the ontology to continue their work after a *BREAK*.

Users edit closely related classes: Additionally to the *breadth-first* manner that users follow when editing classes in the same depth level, we discovered that users have a very high tendency to work on closely related classes (e.g., the sibling or cousin of the currently changed class). The information collected in section 4.2 can be used to predict (or narrow down) the class a user is going to contribute to next, which is a very valuable information that can be used for a variety of improvements and adaptions. For example, user-interface designers could implement pre-fetching algorithms to minimize load-times or emphasize certain areas of the ontology to direct users towards specific classes – especially

after they return from a *BREAK* – while project-administrators could adjust the milestones of the development-strategy to better reflect the way users contribute to the ontology. For contributors in particular, the task of identifying and finding classes that they (i) want and (ii) have the necessary expert knowledge to contribute to is a time consuming task, which can be minimized by implementing class recommender based on the results of the *Structural Paths Analysis* and *User-Sequence Paths Analysis*.

Users perform property-based workflows: The investigation of sequential patterns for property-contributions showed that in ICD-11, users have a very high tendency of consecutively changing the same property across multiple classes.

The conducted evaluation showed that we are able to predict the property a user is most likely to change next by looking at exactly the property that user changed last in both projects.

The results collected in the section 4.3 provide new insights for administrators and ontology-engineering tool developers, as they allow the generation of actual work-tasks (e.g., Please verify the property *title* of the class *XII Diseases of the skin*!). So far, users are always presented first with the section of the interface that allows for changing or adding the *title* and *definition*, which could be one explanation for the high probabilities of users changing these properties when returning from a *BREAK*.

Note, that for this analysis we have used the data from ICD-11 and ICTM, which both share a very similar ontology-engineering tool, thus the results might be biased towards the used ontology-editor.

5.3 Differences between the investigated projects

One obvious difference, which strengthens the observed commonalities of edit strategies between the different projects, is the fact that all projects exhibit a different number of depth levels, which all receive a different amount of attention by the contributors. For example, the levels 3 to 6 exhibit the highest number of changes in our observation period for ICD-11, while for OPL these levels are 6 and 7.

Regarding the relative relationships we can see that consecutively changing the same class is very likely to happen in ICD-11, ICTM, BRO and OPL regardless of the source state (evident in the darker colored *Self* columns in Figures 6(a), 6(b), 6(d) and 6(e)). This *Self*-state is still very prominent, however the transition probabilities towards *Self* for NCIt are not as dominant as they are for the other datasets.

Another observation depicted in the transition maps is the clear focus on transitions from *Sibling* to *Sibling* across all datasets, with the exception of ICTM and OPL. One explanation for ICTM could be the fact that some properties of the ontology are multi-lingual, thus require users to add multiple languages for the same property, which are all stored as a single change. For OPL, transitions, except towards *Self* are in general really scarce, indicating that users focused

on editing and entering multiple property values (or one property value) of one class before continuing to the next class.

When looking at the sequence of changed properties for each class (in contrast to: for each user) we can observe a concentration on *self-loops* in ICTM, which is most likely a direct result of the multi-lingual nature of the properties used in this project. In ICD-11 on the other hand, transitions between changed properties of classes are much more diverse and less focused on transitions between the same properties. This observation indicates that either not all properties have received a substantial amount of values for all the possible properties and/or that users make use of this special export functionality of iCAT, thus *self-loops* are less common as the content is only inserted once into the system.

In the *User-Interface Sections Paths* analysis we have mapped the changed properties to the corresponding sections of the user interface of the used ontology-engineering tools, which essentially represents a more abstract analysis of the *Property Paths* analysis. By investigating the sequences of user interface sections we could confirm that, for ICD-11, users have a very high tendency to consecutively change the same properties for multiple classes, evident in the scarce transitions between different sections and the high concentration on transitions between the same sections. For ICTM this behavior was not as distinctive as it was for ICD-11, which could be due to the missing export functionality and therefore the lack of the previously explained manual import sessions.

In general these observations indicate that the absence or presence of a given functionality of the ontology-engineering tool can produce (and influence) different editing behaviors when developing an ontology.

5.4 Limitations

We were not able to recreate the exact class hierarchy of the ontology for every single change across our observation periods for all datasets. This limitation is partly due to a lack of detail in the change-logs. Thus, we decided to focus our analysis, using all five ontologies *as is* at the latest point in time, which is also what would most likely be used in a *real-world* scenario.

For example, if a class was changed by a user while it was located on depth level 3 and at a later point in time moved to a different location where it now resides at depth level 5, we would assume that this class has always been on depth level 5. Please note that this bias is only present in the *Structural Paths* analyses (Section 4.2). To measure the extent of the potential bias, we counted all changes that were performed on a class before it was moved within in the ontology. Applying this rule to our change dataset, we collected a total of $116,204$ of $439,229$ changes for ICD-11 and $18,958$ of $67,522$ for ICTM. These numbers represent about $1/4$ and $1/3$ of all changes for ICD-11 and ICTM respectively. For BRO 276 of $2,507$ (ca. $1/10$) and for OPL 2 of $1,993$ of all changes were performed on classes, which have been moved afterwards.

Note that an additional requirement for the identification of sequential patterns in collaborative ontology-engineering projects using Markov chains is the availability of rather large change-logs, which we do not discuss in detail in this

paper. In general, the more unique states are present in the change-log the more (exponentially) observations have to be available to detect higher order Markov chains. Without enough observations (changes), the identification of sequential patterns is either very hard and can only be approximated or not possible at all. As can be seen in Table 1, we have selected all of our datasets to satisfy this requirement, as all chosen datasets exhibit a substantial amount of changes.

Furthermore, we have included *artificial session breaks* into our analysis as described by Walk et al. [22] to analyze where or what users start to edit in the ontology and where or what users edit before they take a break. For all user-based analyses we have introduced a *BREAK* state if two consecutive changes of the same user were apart longer than 5 minutes.

6 Related Work

For the analysis and evaluation conducted in this paper, we identified relevant information and publications in the domains of (i) Markov chain models, (ii) collaborative authoring systems and (iii) sequential pattern mining.

6.1 Markov chain models

Previously, Markov chain models have been heavily applied for modeling Web navigation—some sample applications of Markov chains can be found in [24, 25, 26, 27, 28, 29]. Markov chains can be used to uncover the transition likelihoods between a set of states for n transitions in a row, where n represents the order of the Markov chain under investigation. Detailed specifications of the parameters in a Markov chain (e.g., transition probabilities or also the specification of model orders) capture specific assumptions about the real human navigational behavior. One frequently used assumption is that human navigation on the Web is memoryless. This *Markovian assumption* states that the next state depends only on the current state and not on a sequence of preceding ones. The Random Surfer model in Google's PageRank [30] also models this assumption.

Previously, researchers investigated whether human navigation really is memoryless in a series of studies (e.g., [31, 27]). However, these studies mostly showed that the benefit of higher orders is not enough to compensate for the extreme high number of parameters needed and hence, the memoryless model seems to be a quite plausible abstraction (see e.g., [32, 33, 28, 29]). Recently, a study picked up on these investigations and again suggested that the Markovian assumption might be wrong [34]. However, this study did not reveal any statistically significant improvements of higher order models. Singer et al. [21] solved this problem by developing a framework for determining the appropriate order of a Markov chain for given set of input data. Their studies on several navigational datasets also revealed that the memoryless model indeed seems to be a plausible abstraction as it is very difficult for higher order models to show statistically significant improvements due to the high number of parameters needed combined with shortcomings in available data. However, their work showed that on

a cognitive level (by looking at paths over topics instead of pages) clear memory effects can be observed. In Walk et al. [22] we applied and mapped the presented framework onto structured logs of changes and provided an in-depth description of the requirements and steps necessary to use the framework in this setting.

In this paper we present a detailed analysis of sequential patterns by applying and analyzing Markov chains across the change-logs of five collaborative ontology-engineering projects in the biomedical domain. A more detailed explanation of the necessary steps to be able to apply Markov chains onto the change-logs of collaborative ontology-engineering projects is presented in Walk et al. [22].

6.2 Collaborative Authoring Systems

Research on collaborative authoring systems such as Wikipedia has in part focused on developing methods and studying factors that improve article quality or increase user participation. These problems represent important facets of collaborative authoring systems and solutions to tackle these problems are of interest for collaborative ontology-engineering projects.

For example, Cabrera and Cabrera [35] demonstrated the effect of minimizing the costs and efforts necessary for users to contribute on potentially achieving higher contribution rates. Another approach, also presented by Cabrera and Cabrera [35], focuses on providing an environment where interactions and communication between contributors are encouraged and performed frequently over a long period of time to establish a group identity and to promote personal responsibility.

More recent research on collaborative authoring systems, such as Wikipedia, focuses on describing and defining not only the act of collaboration amongst strangers and uncertain situations that contribute to a digital good [36] but also on antagonism and sabotage of said systems [37]. It has also been discovered only recently that Wikipedia editors are slowly but steadily declining [38]. Therefore Halfaker et al. [39] have analyzed what impact reverts have on new editors of Wikipedia. Kittur and Kraut [23] showed that an increase in participation can be achieved by directly delegating specific tasks to contributors. As simple as this approach may appear, the identification of work (and thus specific tasks) is still a tedious and time-consuming process, which can only partly be automated due to its assigned complexity.

With the analysis that we described here, we provide new results that we can use to tackle some of the problems for collaborative authoring systems. These problems are also present in collaborative ontology-engineering projects. For example, we can identify new tasks by combining the results of the *User-Sequence Paths* (Section 4.1) and *Property Paths* (Section 4.3) analyses to suggest classes and the corresponding properties to work on to users.

6.3 Sequential Pattern Mining

In 1995 Agrawal and Srikant [40] have first addressed the problem of sequential pattern mining. They stated that given a collection of chronologically ordered sequences, sequential pattern mining is about discovering all sequential patterns weighted according to the number of sequences that contain these patterns. The presented algorithm represents one of the first *a priori* sequential pattern mining algorithms. This means that a specific pattern cannot occur more frequently (above a threshold) if a sub-pattern of this pattern occurs less often (below that threshold). Other examples of a priori algorithms are [41, 42].

One of the biggest problems assigned to the a priori based sequential pattern mining algorithms was (in the worst case) the exponential number of candidate generation. To tackle this problem Han et al. [43] developed the FP-Growth algorithm.

Many researchers have adapted different algorithms and approaches for different domains to anticipate changing requirements, such as Wang and Han [44] and Hsu et al. [45] who analyzed algorithms for sequential pattern mining in the biomedical domain.

In Walk et al. [22] the authors have presented a novel application of Markov chains to mine and determine sequential patterns from the structured logs of changes of collaborative ontology-engineering projects. Making use of this framework we investigate differences and commonalities across five different collaborative ontology-engineering projects from the biomedical domain.

7 Conclusions & Future Work

In this work, we discover intriguing social and sequential patterns that suggest that large collaborative ontology engineering projects are governed by a few general principles that determine and drive development. Specifically, our results indicate that general sequential patterns can be found in all investigated projects (see Table 3), even though the National Cancer Institute Thesaurus (NCIt), the International Classification of Diseases (ICD-11), the International Classification of Traditional Medicine (ICTM), the Ontology for Parasite Lifecycle (OPL) and the Biomedical Resource Ontology (BRO) (i) represent different projects with different goals, (ii) use variations of the same ontology-editors and tools for the engineering process and (iii) differ in the way the projects are coordinated. The presented analysis not only provides new insights about the engineering and development processes of each single project, but also shows that the analysis of sequential patterns can be used to provide actionable insights for different stakeholders in collaborative ontology-engineering projects.

Furthermore, the information of the next possible state (e.g., a user, a change-type, a property, set of classes) or the combination of multiple of these next states can be used by ontology-engineering tool developers to augment users in collaboratively creating an ontology. For example, by highlighting, prefetching, rearranging or adjusting sections and content of the interface dynamically, according to the user's needs.

In the future, as change tracking and even click tracking data will become available more broadly, we believe that the analysis of this paper and the possible benefits of putting the results into practical use represent an import step towards the development of even better (and simpler) ontology editors, which can dynamically anticipate the editing-style of the users. Project administrators could augment the results of the analysis, for example by allowing for easier delegation of work to the "right" users. This is even more emphasized when considering that the Markov chain analysis is not computationally intensive, making it highly suitable for productive use.

As biomedical ontologies play an increasingly critical role in acquiring, representing, and processing information about human health, we can use quantitative analysis of editing behavior to generate useful insights for building better tools and infrastructures to support these tasks.

Bibliography

[1] K. Weller, Knowledge Representation in the Social Semantic Web, vol. 3 of *Knowledge and Information*, De Gruyter Saur, Berlin, Germany, 2010.

[2] T. Gruber, A translation approach to portable ontology specifications, Knowledge Acquisition 5 (2) (1993) 199–220.

[3] W. Borst, Construction of engineering ontologies for knowledge sharing and reuse .

[4] R. Studer, V. R. Benjamins, D. Fensel, Knowledge engineering: Principles and methods, vol. 25, 161–197, 1998.

[5] N. F. Noy, T. Tudorache, Collaborative Ontology Development on the (Semantic) Web., in: AAAI Spring Symposium: Symbiotic Relationships between Semantic Web and Knowledge Engineering, AAAI, 63–68, URL `http://dblp.uni-trier.de/db/conf/aaaiss/aaaiss2008-7.html#NoyT08`, 2008.

[6] T. Groza, T. Tudorache, M. Dumontier, Commentary: State of the Art and Open Challenges in Community-driven Knowledge Curation, Journal of Biomedical Informatics 46 (1) (2013) 1–4, ISSN 1532-0464, URL `http://dx.doi.org/10.1016/j.jbi.2012.11.007`.

[7] M. Krötzsch, D. Vrandecic, M. Völkel, Semantic MediaWiki, in: Proceedings of the 5th International Semantic Web Conference 2006 (ISWC 2006), Springer, 935–942, 2006.

[8] S. Auer, S. Dietzold, T. Riechert, OntoWiki–A Tool for Social, Semantic Collaboration, in: Proceedings of the 5th International Semantic Web Conference (ISWC 2006), vol. LNCS 4273, Springer, Athens, GA, 2006.

[9] C. Ghidini, B. Kump, S. Lindstaedt, N. Mahbub, V. Pammer, M. Rospocher, L. Serafini, MoKi: The Enterprise Modelling Wiki, in: L. Aroyo, P. Traverso, F. Ciravegna, P. Cimiano, T. Heath, E. Hyvönen, R. Mizoguchi, E. Oren, M. Sabou, E. P. B. Simperl (Eds.), Proceedings of the 6th European Semantic Web Conference on The Semantic Web: Research and Applications 2009, Springer, Berlin, Heidelberg, 831–835, 2009.

[10] T. Schandl, A. Blumauer, Poolparty: SKOS thesaurus management utilizing linked data, The Semantic Web: Research and Applications 6089 (2010) 421–425.

[11] T. Tudorache, C. Nyulas, N. F. Noy, M. A. Musen, WebProtégé: A Distributed Ontology Editor and Knowledge Acquisition Tool for the Web, Semantic Web Journal 4 (1/2013) (2013) 89–99.

[12] T. Tudorache, S. M. Falconer, C. I. Nyulas, N. F. Noy, M. A. Musen, Will Semantic Web technologies work for the development of ICD-11?, in: Proceedings of the 9th International Semantic Web Conference (ISWC 2010), ISWC (In-Use), Springer, Shanghai, China, 2010.

[13] J. Pöschko, M. Strohmaier, T. Tudorache, M. A. Musen, Pragmatic analysis of crowd-based knowledge production systems with iCAT Analytics: Visualizing changes to the ICD-11 ontology, in: Proceedings of the AAAI

Spring Symposium 2012: Wisdom of the Crowd, accepted for publication, 2012.

[14] S. Walk, J. Pöschko, M. Strohmaier, K. Andrews, T. Tudorache, C. Nyulas, M. A. Musen, N. F. Noy, PragmatiX: An Interactive Tool for Visualizing the Creation Process Behind Collaboratively Engineered Ontologies, International Journal on Semantic Web and Information Systems .

[15] S. M. Falconer, T. Tudorache, N. F. Noy, An analysis of collaborative patterns in large-scale ontology development projects., in: M. A. Musen, . Corcho (Eds.), K-CAP, ACM, ISBN 978-1-4503-0396-5, 25–32, URL http://dblp.uni-trier.de/db/conf/kcap/kcap2011.html#FalconerTN11, 2011.

[16] C. Pesquita, F. M. Couto, Predicting the Extension of Biomedical Ontologies, PLoS Comput Biol 8 (9) (2012) e1002630, URL http://dx.doi.org/10.1371%2Fjournal.pcbi.1002630.

[17] M. Strohmaier, S. Walk, J. Pöschko, D. Lamprecht, T. Tudorache, C. Nyulas, M. A. Musen, N. F. Noy, How Ontologies are Made: Studying the Hidden Social Dynamics Behind Collaborative Ontology Engineering Projects, Web Semantics: Science, Services and Agents on the World Wide Web 20 (0), ISSN 1570-8268, URL http://www.websemanticsjournal.org/index.php/ps/article/view/333.

[18] H. Wang, T. Tudorache, D. Dou, N. F. Noy, M. A. Musen, Analysis of User Editing Patterns in Ontology Development Projects, in: On the Move to Meaningful Internet Systems: OTM 2013 Conferences, Springer, 470–487, 2013.

[19] N. Sioutos, S. de Coronado, M. W. Haber, F. W. Hartel, W.-L. Shaiu, L. W. Wright, NCI Thesaurus: A semantic model integrating cancer-related clinical and molecular information, Journal of Biomedical Informatics 40 (1) (2007) 30–43.

[20] J. D. Tenenbaum, P. L. Whetzel, K. Anderson, C. D. Borromeo, I. D. Dinov, D. Gabriel, B. A. Kirschner, B. Mirel, T. D. Morris, N. F. Noy, C. Nyulas, D. Rubenson, P. R. Saxman, H. Singh, N. Whelan, Z. Wright, B. D. Athey, M. J. Becich, G. S. Ginsburg, M. A. Musen, K. A. Smith, A. F. Tarantal, D. L. Rubin, P. Lyster, The Biomedical Resource Ontology (BRO) to enable resource discovery in clinical and translational research, Journal of Biomedical Informatics 44 (1) (2011) 137–145.

[21] P. Singer, D. Helic, B. Taraghi, M. Strohmaier, Memory and Structure in Human Navigation Patterns, arXiv preprint arXiv:1402.0790 .

[22] S. Walk, P. Singer, M. Strohmaier, N. F. Noy, M. A. Musen, Mining Sequential Patterns from Collaborative Ontology-Engineering Change-Logs, International Journal of Human Computer Studies .

[23] A. Kittur, R. E. Kraut, Harnessing the wisdom of crowds in wikipedia: quality through coordination, in: Proceedings of the 2008 ACM conference on Computer supported cooperative work, CSCW '08, ACM, New York, NY, USA, 37–46, 2008.

[24] J. Borges, M. Levene, Evaluating Variable-Length Markov Chain Models for Analysis of User Web Navigation Sessions, IEEE Trans. on Knowl. and

Data Eng. 19 (4) (2007) 441–452, ISSN 1041-4347, URL http://dx.doi.org/10.1109/TKDE.2007.1012.

[25] M. Deshpande, G. Karypis, Selective Markov models for predicting Web page accesses, ACM Trans. Internet Technol. 4 (2) (2004) 163–184, ISSN 15335399, URL http://doi.acm.org/10.1145/990301.990304.

[26] R. Lempel, S. Moran, The stochastic approach for link-structure analysis (SALSA) and the TKC effect, Comput. Netw. 33 (1-6) (2000) 387–401, ISSN 1389-1286, URL http://dx.doi.org/10.1016/S1389-1286(00)00034-7.

[27] P. L. T. Pirolli, J. E. Pitkow, Distributions of surfers' paths through the World Wide Web: Empirical characterizations, World Wide Web 2 (1-2) (1999) 29–45, ISSN 1386145X, URL http://dx.doi.org/10.1023/A:1019288403823.

[28] R. Sen, M. Hansen, Predicting a Web user's next access based on log data, Journal of Computational Graphics and Statistics 12 (1) (2003) 143–155, URL http://citeseer.ist.psu.edu/sen03predicting.html.

[29] I. Zukerman, D. W. Albrecht, A. E. Nicholson, Predicting users' requests on the WWW, Proceedings of the Seventh International Conference on User Modeling, Springer-Verlag New York, Inc., Secaucus, NJ, USA, ISBN 3-211-83151-7, 275–284, URL http://dl.acm.org/citation.cfm?id=317328.317370, 1999.

[30] S. Brin, L. Page, The anatomy of a large-scale hypertextual Web search engine, in: Proceedings of the seventh international conference on World Wide Web 7, WWW7, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, 107–117, 1998.

[31] J. Borges, M. Levene, Data Mining of User Navigation Patterns, in: Revised Papers from the International Workshop on Web Usage Analysis and User Profiling, WEBKDD '99, Springer-Verlag, London, UK, UK, ISBN 3-540-67818-2, 92–111, URL http://dl.acm.org/citation.cfm?id=648036.744399, 2000.

[32] I. Cadez, D. Heckerman, C. Meek, P. Smyth, S. White, Model-Based Clustering and Visualization of Navigation Patterns on a Web Site, Data Min. Knowl. Discov. 7 (4) (2003) 399–424, ISSN 13845810, URL http://dx.doi.org/10.1023/A:1024992613384.

[33] R. R. Sarukkai, Link prediction and path analysis using Markov chains, Proceedings of the 9th international World Wide Web conference on Computer networks: the international journal of computer and telecommunications netowrking, North-Holland Publishing Co., Amsterdam, The Netherlands, The Netherlands, 377–386, URL http://dl.acm.org/citation.cfm?id=347319.346322, 2000.

[34] F. Chierichetti, R. Kumar, P. Raghavan, T. Sarlos, Are web users really Markovian?, in: Proceedings of the 21st international conference on World Wide Web, WWW '12, ACM, New York, NY, USA, ISBN 978-1-4503-1229-5, 609–618, URL http://doi.acm.org/10.1145/2187836.2187919, 2012.

[35] A. Cabrera, E. F. Cabrera, Knowledge-Sharing Dilemmas, Organization Studies 23 (5) (2002) 687–710.

[36] B. Keegan, D. Gergle, N. S. Contractor, Hot off the wiki: dynamics, practices, and structures in Wikipedia's coverage of the Tohoku catastrophes., in: F. Ortega, A. Forte (Eds.), Int. Sym. Wikis, ACM, 105–113, 2011.

[37] N. Shachaf, Beyond vandalism: Wikipedia trolls., Journal of Information Science; Jun2010, Vol. 36 Issue 3, p357-370, 14p, 2 Charts .

[38] B. Suh, G. Convertino, E. H. Chi, P. Pirolli, The singularity is not near: slowing growth of Wikipedia, in: WikiSym '09: Proceedings of the 5th International Symposium on Wikis and Open Collaboration, ACM, New York, NY, USA, 1–10, 2009.

[39] A. Halfaker, A. Kittur, J. Riedl, Don't bite the newbies: how reverts affect the quantity and quality of Wikipedia work., in: F. Ortega, A. Forte (Eds.), Int. Sym. Wikis, ACM, 163–172, 2011.

[40] R. Agrawal, R. Srikant, Mining Sequential Patterns, in: Proceedings of the Eleventh International Conference on Data Engineering, ICDE '95, IEEE Computer Society, Washington, DC, USA, ISBN 0-8186-6910-1, 3–14, URL http://dl.acm.org/citation.cfm?id=645480.655281, 1995.

[41] R. T. Ng, L. V. S. Lakshmanan, J. Han, A. Pang, Exploratory Mining and Pruning Optimizations of Constrained Associations Rules, in: Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, SIGMOD '98, ACM, New York, NY, USA, ISBN 0-89791-995-5, 13–24, URL http://doi.acm.org/10.1145/276304.276307, 1998.

[42] S. Sarawagi, S. Thomas, R. Agrawal, Integrating Association Rule Mining with Relational Database Systems: Alternatives and Implications, in: Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, SIGMOD '98, ACM, New York, NY, USA, ISBN 0-89791-995-5, 343–354, URL http://doi.acm.org/10.1145/276304.276335, 1998.

[43] J. Han, J. Pei, Y. Yin, Mining Frequent Patterns Without Candidate Generation, in: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, SIGMOD '00, ACM, New York, NY, USA, ISBN 1-58113-217-4, 1–12, URL http://doi.acm.org/10.1145/342009.335372, 2000.

[44] J. Wang, J. Han, BIDE: Efficient Mining of Frequent Closed Sequences, in: Proceedings of the 20th International Conference on Data Engineering, ICDE '04, IEEE Computer Society, Washington, DC, USA, ISBN 0-7695-2065-0, 79–, URL http://dl.acm.org/citation.cfm?id=977401.978142, 2004.

[45] C.-M. Hsu, C.-Y. Chen, B.-J. Liu, C.-C. Huang, M.-H. Laio, C.-C. Lin, T.-L. Wu, Identification of hot regions in protein-protein interactions by sequential pattern mining, BMC bioinformatics 8 (Suppl 5) (2007) S8.

A Evaluation Details

To determine the predictive power of our determined Markov chain models we conduct a cross-fold validation prediction task. The main idea behind this approach is to calculate the transition-probabilities and parameters on a training

set and validate the model on an independent test set. As described in Singer et al. [21], Walk et al. [22], we also apply Laplace smoothing in this case in order to be able to predict states that are only present in the test set and not in the training set. For reducing variance we perform a stratified cross-fold validation. In this case we refer to the term stratified as we try to keep the number of visited states in each fold equal. Note that the number of folds is determined for each evaluation individually due to their stratified nature. Not every dataset can be split into the same amount of pieces while retaining the stratified property of the corresponding states. Therefore we will explicitly state the number of performed folds that we were able to perform for each analysis and all datasets in the corresponding evaluation subsections.

The validations are based on the task of predicting the next step in a path of the test set. This also enables us to get detailed insights into the prediction-possibilities of distinct Markov chain order models. Simply, one could predict the next state by taking the one with the highest probability in the transition matrix.

For calculating the prediction accuracy we measure the average rank of the actual state of the test path in the sorted probabilities from the transition matrix. Hence, we look up the rank of the *target state* in the sorted list of transition probabilities of the *start state* (or series of preceding states for higher order models). Next, we average over the rank of all observations in the test set. Finally, we average over all folds and suggest the model with the lowest average rank.

Table 3. The results for all datasets and all analyses conducted in section 4. *Evaluation* indicates the best-fitting calculated order of a Markov chain in our prediction task while *Suggested* indicates the manually selected best-fitting order of a Markov chain simultaneously representing the best trade-off between complexity of the Markov chain (and thus calculations) and the average position in our evaluation task. Analyses marked with a * have been performed but are not described in detail in this manuscript but are briefly discussed in section 5.

			ICD-11	ICTM	NCIt	BRO	OPL
User-Sequence Paths (cf. section 4.1)	User-Sequence Paths (**Section 4.1**)	Evaluation	2	2	2	1	1
		Suggested	1	1	1	1	1
	Change-Type Paths (**Section 5**)*	Evaluation	3	3	3	1	2
		Suggested	1	1	1	1	2
Structural Paths (cf. section 4.2)	Edit-Strategy Paths (**Section 5**)*	Evaluation	4	4	5	2	2
		Suggested	3	3	2	2	2
	Depth-Level Paths (**Section 4.2**)	Evaluation	2	2	2	1	1
		Suggested	1	1	1	1	1
	Relative Relationship Paths (**Section 4.2**)	Evaluation	3	2	3	1	1
		Suggested	2	2	2	1	0
Property Paths (cf. section 4.3)	Property Paths (User) (**Section 4.3**)	Evaluation	2	2	-	-	-
		Suggested	1	1	-	-	-
	Property Paths (class) **(Section 4.3**)	Evaluation	1	1	-	-	-
		Suggested	1	1	-	-	-
	UI Sections Paths (User) (**Section 5**)*	Evaluation	3	3	-	-	-
		Suggested	1	1	-	-	-
	UI Sections Paths (class) (**Section 5**)*	Evaluation	3	1	-	-	-
		Suggested	1	1	-	-	-

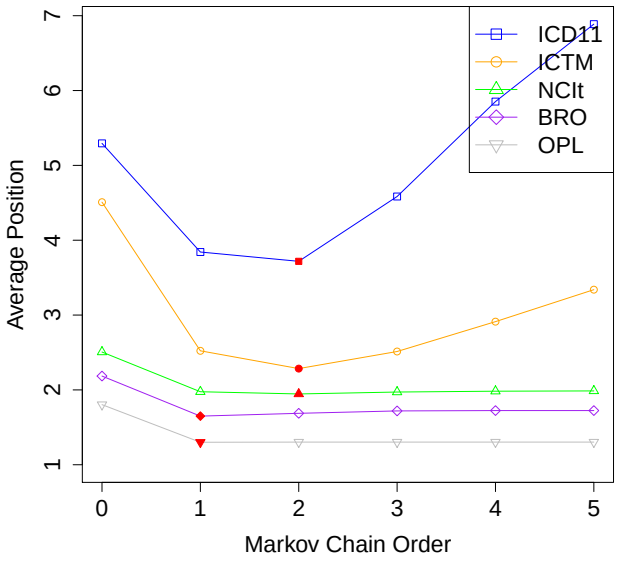


Fig. 8. Results for the *User-Sequence Paths Evaluation*: This plot depicts the results of the stratified cross-fold evaluation for all five datasets for the *User-Sequence Paths* analysis. The filled elements represent the corresponding Markov-Chain models for each dataset, which achieved the best (lowest) average position score in the prediction tasks. The position score is calculated by determining the position of the next most likely state to occur in a test path given n previous states, where n represents the investigated Markov chain order. Probabilities to select the next most likely state are created using the training set to calculate the transition maps for all datasets and Markov chain orders. For all datasets either a first- or second-order Markov chain performed best. Given the minimal differences between the second- and first-order Markov chains, we suggest the usage of a first-order Markov model for productive use.

User-Sequence Paths Evaluation (cf. section 4.1): To determine the order of a Markov chain with the highest predictive power a 4-fold stratified cross-fold validation was performed. According to Figure 8 a first- and second-order Markov chain works best for all datasets. Due to very similar results across the different Markov chain orders and the increased complexity of higher ordered models, we would suggest the usage of a first-order Markov chain model for all datasets. Overall, the results of the conducted prediction task suggests that the user who modifies a class next appears to depend on the previous one. Given the relatively low average positions in Figure 8 we are able to predict the next user who edits a class reasonable well for all of our datasets.

Depth-Level Paths Evaluation (cf. section 4.2): The evaluation was carried out as a two-fold stratified cross-fold prediction task (see Figure 9; sec-

tion A for a detailed explanation). The goal of this evaluation is to determine which order of the investigated Markov chains performs best for predicting the depth level of the class a user is going to change next.

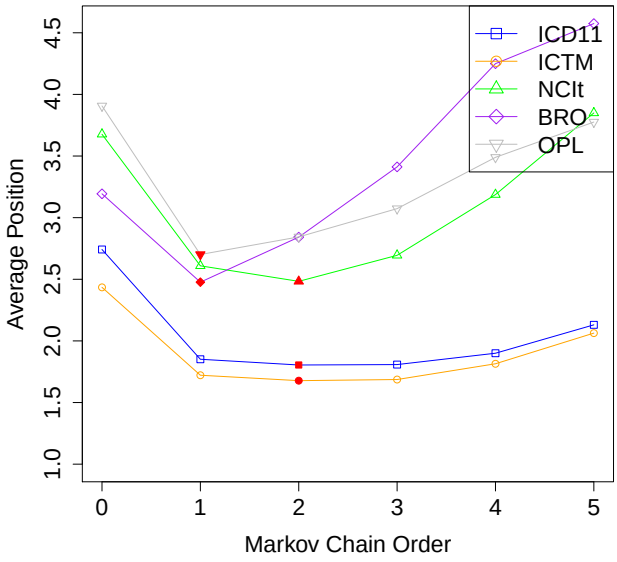


Fig. 9. Results for the *Depth-Level Paths Evaluation*: This plot depicts the results of the stratified cross-fold evaluation for all five datasets for the *Depth-Level Paths* analysis. The filled elements represent the corresponding Markov-Chain models for each dataset, which achieved the best (lowest) average position score in the prediction tasks. The position score is calculated by determining the position of the next most likely state to occur in a test path given n previous states, where n represents the investigated Markov chain order. Probabilities to select the next most likely state are created using the training set to calculate the transition maps for all datasets and Markov chain orders. For all projects a first- or second-order Markov chain performed best. As the differences between the higher-order Markov chains and the first-order Markov chains are minimal we suggest the usage of a first-order Markov model for predictive tasks in all datasets.

According to the cross-fold validation a first- and second-order Markov chain model performs best for our prediction task. As the differences in average positions between the first- and second-order models are very small, we recommend the usage of a first-order Markov chain model in our five investigated projects due to the drastic increase in complexity of higher-order models. This means that the next depth can best be predicted by using information about the current one.

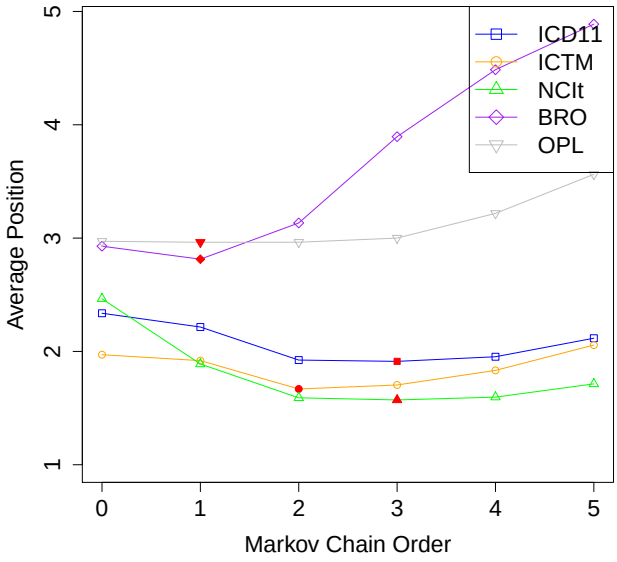


Fig. 10. Results for the *Relative-Relationship Paths Evaluation*: This plot depicts the results of the stratified cross-fold evaluation for all five datasets for the *Relative-Relationship Paths* analysis. The filled elements represent the corresponding Markov-Chain models for each dataset, which achieved the best (lowest) average position score in the prediction tasks. The position score is calculated by determining the position of the next most likely state to occur in a test path given n previous states, where n represents the investigated Markov chain order. Probabilities to select the next most likely state are created using the training set to calculate the transition maps for all datasets and Markov chain orders. For OPL and BRO a first-order Markov chain performed best while a second-order model provided the best results for ICTM. For NCIt and ICD-11 a third-order Markov chain was determined to provide the best results in our prediction task. As the differences between the higher-order Markov chains and the second-order Markov chains are minimal we suggest the usage of a second-order Markov model for ICD-11, ICTM and NCIt for the given prediction task. For OPL and BRO the differences between a zero- and first-order Markov chains are very small, meaning that the difference between randomly selecting the next state and using the transition-probabilities of a first-order Markov chain to determine the next state perform virtually the same.

Relative-Relationship Paths Evaluation (cf. section 4.2): To determine the Markov chain models with the highest predictive power a stratified cross-fold validation with two folds (see Figure 10) was conducted. For BRO and OPL a first-order Markov chain performed better in our prediction task than any other model, meaning that the extracted transition probabilities contain minimally more information than randomly selecting the next state. For ICTM, the

cross-fold validation strengthens suggests a second-order Markov chain. Similarly to NCIt and ICD-11, where a third-order Markov chain yielded minimally better results than a second order Markov chain. That means that we would prefer to predict the next relative relationship between two consecutively changed classes by using information about the current relationship or a series of two preceding relationships. Note that for this analysis, randomly selecting the next state already produces reasonable average positions. Hence, improving on these results is difficult, resulting in a higher weight for improvements when manually determining the best balance between a Markov chains complexity and the average position score of our evaluation task.

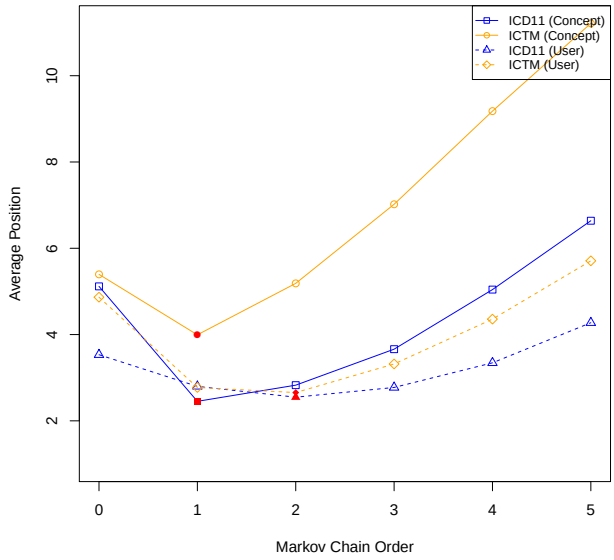


Fig. 11. Results for the *Property Paths Evaluation*: This plot depicts the results of the stratified cross-fold evaluation for all five datasets for the *Property Paths* analysis. The filled elements represent the corresponding Markov-Chain models for each dataset, which achieved the best (lowest) average position score in the prediction tasks. The position score is calculated by determining the position of the next most likely state to occur in a test path given n previous states, where n represents the investigated Markov chain order. Probabilities to select the next most likely state are created using the training set to calculate the transition maps for all datasets and Markov chain orders. For the user-based approach a second-order Markov chain performed best for both datasets. In contrast, a first-order Markov chain yielded the best average positions for both datasets for the class-based approach. Given the very increase in average position of a second-order Markov chain versus a first-order Markov chain we suggest the usage of a first-order Markov chain for productive use.

Property Paths Evaluation (cf. section 4.3): We have conducted a 4-fold stratified cross-fold evaluation for all four approaches (see Figure 11; section A for a detailed explanation) to evaluate the predictive power the different Markov chain models. The class-based approaches performed best using a second-order Markov chain for the prediction task while the first-order Markov chain is equally good which is why we would suggest the less-complex model (first-order) in this case. For the user-based approach a first-order Markov chain performed best. This means that we would prefer to only use the currently changed property in a path as information for predicting the next one (for both the user and class case).