



# A Repository for Formal Contexts

Tom Hanika<sup>1</sup>  and Robert Jäschke<sup>2</sup> 

<sup>1</sup> University of Hildesheim

`tom.hanika@uni-hildesheim.de`

<sup>2</sup> Berlin School for Library and Information Science

Humboldt-Universität zu Berlin

`robert.jaeschke@hu-berlin.de`

**Abstract.** Data is always at the center of the theoretical development and investigation of the applicability of formal concept analysis. It is therefore not surprising that a large number of data sets are repeatedly used in scholarly articles and software tools, acting as de facto standard data sets. However, the distribution of the data sets poses a problem for the sustainable development of the research field. There is a lack of a central location that provides and describes FCA data sets and links them to already known analysis results. This article analyses the current state of the dissemination of FCA data sets, presents the requirements for a central FCA repository, and highlights the challenges for this.

## 1 Introduction

Reproducing and comparing results are mandatory for the scientific method. This is in particular true for data-driven and data-centered research fields, such as formal concept analysis (FCA). Research data repositories are therefore an integral part of a functioning research ecosystem.

A large number of such repositories have been established in the field of machine learning, such as the *The UCI Machine Learning Repository* [18], the platform OpenML [30], or *Hugging Face*.<sup>3</sup> There is a distinction between general and specialized repositories. The latter can be specialized, for example, in data types, application areas, or learning methods. Without such repositories, freely circulating data sets are used in a research community. It is often not possible to trace where the data originates from, how it may have been derived from another data set, whether it is complete and correct, etc. What's more, data can disappear from the community and no longer be found, for example, if researchers retire or websites vanish.

This problem and its effects increasingly affect the research field of FCA, which has also been recognized previously [2, 4, 22]. Various researchers have approached this problem in part by providing websites<sup>4</sup> for the FCA community

---

<sup>3</sup> <https://huggingface.co/>

<sup>4</sup> for example, <https://upriss.github.io/fca/examples.html>

that link different sources of information, or by including test data sets<sup>5</sup> in their FCA software [14, 6].

A problem analysis and initial approaches to solving this issue have already been described independently by S. Andrews [2] and C. Orphanides and G. Georgiou [22]. More than ten years later, however, none of the proposed solutions have been manifested themselves in real existing repositories. This is where this work comes in by a) presenting an updated analysis of the requirements for an FCA repository, b) describing an approach for an initial solution, and c) presenting a rudimentary repository,<sup>6</sup> that we have already initiated, and its future development.

With our solution we follow the KISS<sup>7</sup> principle, favoring a simple functioning repository (with few features) over a complicated and therefore difficult to implement repository. Already in its present form, the repository allows for a) testing algorithms (and implementations) on correctness, b) benchmarking algorithms, c) comparing different FCA procedures, d) helping beginners to explore or learn FCA. It also contains the “classic” data sets from Ganter and Wille’s FCA book [12]. Furthermore, our approach supports low-threshold access similarly to other popular data science libraries, for example, *Seaborn*<sup>8</sup> via its `load_dataset()` method. In our repository located at <https://fcarepository.org/>, a) each context is a file in a git repository, and b) the metadata for each context is described in a file that is machine-readable and human-editable.

With this modeling we obtain version control, a workflow for collaboration and contributions (forks, pull requests), a continuous integration pipeline for the automatic generation of derivatives (e.g., human-readable documentation, statistics, lattice diagrams), simple programmatic access using HTTP, etc. Therefore, the repository could easily be integrated into FCA workflows, tools, and libraries and can enable and simplify the (re)use of FCA data. For sustainability, we envision to create snapshots of the git repository on a regular basis and to publish them on suitable platforms, such as Zenodo.<sup>9</sup>

In order to make the FCA repository sustainable and as scientifically reliable as possible, many challenges still need to be overcome. Specifically: a) a curation policy and metadata schema has to be developed, b) more formal contexts and metadata have to be collected, and c) authors of FCA tools and libraries need to implement a reliable interface to the repository. We hope that our initial analysis and solution helps to raise awareness for the repository problem and starts the discussion that gathers feedback from the FCA community.

This paper is organized as follows: In Section 2 we give an overview on related work and then discuss existing related approaches in more depth in Section 3.

---

<sup>5</sup> <https://github.com/tomhanika/conexp-clj/tree/dev/testing-data>, <https://github.com/neuroimaging/fcaR/tree/master/data>

<sup>6</sup> <https://fcarepository.org/>

<sup>7</sup> *keep it simple, stupid*

<sup>8</sup> <https://github.com/mwaskom/seaborn-data>

<sup>9</sup> <https://zenodo.org/>

In the main part in Section 4 we analyze requirements for an FCA repository and provide suggestions for implementation. Finally, we discuss next steps and the organization of the repository in Section 5 and conclude the paper with a discussion in Section 6.

## 2 Related Work

There exists a wide variety of repositories that host research data – institutional, domain-specific, and generic, such as Zenodo [9], which was launched by the OpenAIRE<sup>10</sup> partner CERN. Most of them are clearly intended as archives for research data without providing any insights into the data apart from basic metadata (e.g., license, authorship, file format). Registries, such as, *re3data*<sup>11</sup> record more than 3200 different repositories [26].

A prominent example in computer science is the UCI Machine Learning Repository [18]. Founded in 1987, it is one of the oldest repositories hosting data sets for the empirical analysis of machine learning algorithms. With a focus on tabular data of ‘instances’ described by ‘variables’ it provides basic information about each variable (e.g., role, type, unit, description). In the realm of machine learning, OpenML [30] considerably extends this idea by allowing everyone to share data sets, tasks, implementations, and results. Its goal is to enable “networked science” that uses “online tools to share, structure and analyse scientific data on a global scale” [30].

KONECT, the Koblenz Network Collection [19], is a repository dedicated to network science (i.e., graph-represented) data sets. Its web site facilitates the exploration of such data sets by providing more than thirty descriptive network statistics (e.g., average degree, mean distance) and data visualizations (e.g., spectral distributions, degree assortativity). In addition, a handbook and a toolbox for the statistics framework GNU/R supports researchers in using the data.

Finally, DraCor [11] is an example from the humanities, specifically literature. It is a website and REST-based API that provides multi-faceted access to drama corpora, coined “programmable corpora” by its founders. Apart from the full-texts of the dramas, DraCor provides metadata, and structured information (e.g., speaker text) together with a web-based interface for exploration of the character co-occurrence networks.

Various trends can be identified in the recent development of research data repositories. On the one hand, the creation and further development of existing generalist repositories. Secondly, the creation of domain-specific repositories, for example, for agricultural science [28] or mathematics [8], or application-specific, for instance, collaborative research centers [21]. The latter two are particularly favored in Germany by an initiative for a national research data infrastructure.

There are reasons for the emergence and further development of repositories that are specifically tailored to certain areas of research: by focusing on certain types of data and file formats, they can provide dedicated services, for example,

<sup>10</sup> <https://www.openaire.eu/>

<sup>11</sup> <https://re3data.org/>

additional metadata, faceted search, an integration into the tools and workflows of the research community, or draw on the expertise from other domain experts.

A web-based repository for formal contexts was first described in 2009 by S. Andrews [2]. The idea was that the repository hosts donated and randomly-generated data sets, “along with, where possible, [their] original data file, information about the original data (probably from the original data source), a link back to the original data source, the conversion log, and FCA information, such as context density and number of concepts” [2]. However, few details were provided and – to the best of our knowledge – the repository either was never realized or is no longer available online. Four years later C. Orphanides and G. Georgio proposed *FCAWarehouse* [22], a website no longer in service.

In 2016, Hanika and Borchmann [4] reiterated the need for an archive of data and computational results related to FCA when they uncovered implausible results in the Stegosaurus phenomenon. As far as the authors are aware, there is no other domain-specific FCA repository apart from the one presented here.

### 3 Existing Related Approaches

As described in the previous section, there are no other FCA repositories. However, there are a number of tools and other sources of information that perform some of the tasks of an FCA repository. Moreover, the authors are keen to integrate the proposed repository into existing FCA tools. We therefore provide a brief overview for them. In addition, we can justify some of the design decisions of the FCA repository based on their characteristics.

#### 3.1 Tools

There is an abundance of software libraries and tools to conduct formal concept analysis. In the following we provide a brief overview on the most recent tools. Therefore, we omitted offline-tools that were not updated for more than two years. Moreover we discarded for our analysis pure concept mining tools, such as *pcbo* or *inclose*. We acknowledge that the overview is not exhaustive and might miss important tools.

**xflr6 / concepts** is a basic but useful Python3 library for FCA. It allows for computing and drawing concept lattices.<sup>12</sup> Binary comma separated value (CSV) as well as the Burmeister format are supported for loading and storing formal contexts. **FCapy** is also a Python3 FCA library with a focus on interacting with machine learning algorithms. **fcaR** is a popular library within the GNU/R ecosystem. In particular it allows for computing implications and conceptual scaling [6] **ConExp** – “The Concept Explorer” is one of the classic FCA tools. Although its last release was in 2013, it is still very popular due to its intuitive graphical user interface. **FcaKit** is a software library written in the Swift programming language, which is predominantly used in the iOS / MacOS

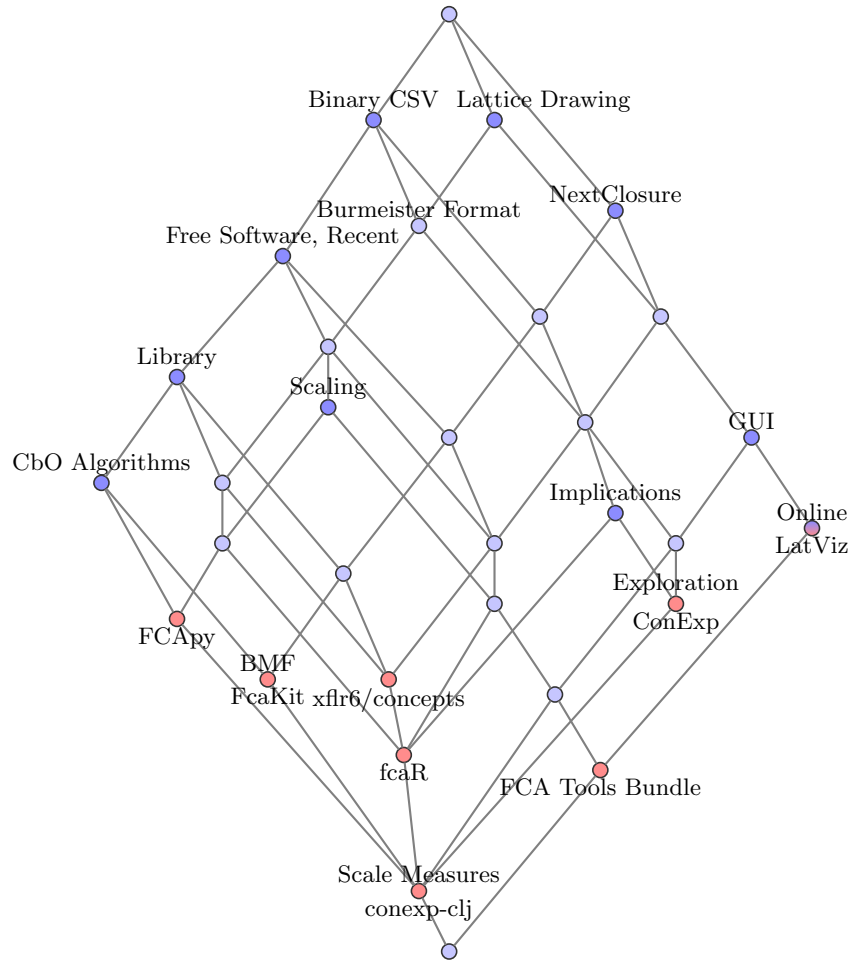
<sup>12</sup> <https://github.com/xflr6/concepts>

	Implications	Lattice Drawing	Free Software	ChO Algorithms	Burmeister Format	Scaling	NextClosure	Library	Recent	Exploration	BMF	Scale Measures	Binary CSV	Online	GUI
xflr6 / concepts		×	×		×		×	×	×				×		
fcaR	×	×	×		×	×	×	×	×				×		
FCApy		×	×	×	×	×		×	×				×		
FCA Tools Bundle		×	×		×	×	×		×				×	×	×
FcaKit			×	×			×	×	×		×		×		
conexp-clj	×	×	×	×	×	×	×	×	×	×	×	×	×		×
LatViz		×					×								×
ConExp	×	×			×		×			×			×		×

**Fig. 1.** Comparison of recent FCA tools. The classic Conexp tool was included for comparison. Also non-recent online platforms were added.

ecosystem. Many concept mining and factorization algorithms are implemented within this library. **conexp-clj** is one of the most versatile software libraries for FCA. It is implemented in the functional programming language Clojure and allows interaction with Java code, among other things. **FCA Tools Bundle** is an online platform [7] and has a large collection (i.e., 166 files) of formal contexts. The context are presented with minor meta data (e.g., number of objects, etc) but only minimal provenience information is available, as many data sets have no description beyond one or a couple of words. Some data sets can be exported as CSV files and others using the Burmeister format. The focus of this platform is on the analysis of triadic and polyadic contexts. **LatViz** is also an online platform [1] which allows editing formal context and computing their concept lattice. Data sets can only be exchanged via an undocumented JSON format. However, a converter from binary CSV is provided as an offline tool. All these tools and their attributes are depicted in Figure 1. The corresponding lattice is shown in Figure 2. For a more exhaustive and extensive comparison of FCA tools we refer the reader to Saab et al. [27]. Their analysis includes many tools that we have discarded for the reasons mentioned above.

T. Tilley already addressed the problem of tool support in 2004 [29] and U. Priss addressed the problem of data interoperability for FCA in 2008 [24, 25]. Unfortunately, none of the tools U. Priss considered in her work made it into our comparison, as there have been no releases for many years. Irrespective of this, the considerations made are still valid. In particular, the finding that the Burmeister format and the representation as a binary CSV have the widest support among the FCA tools.



**Fig. 2.** Concept Lattice for the formal context in Figure 1.

### 3.2 Data Collections

U. Priss has some ‘classic’ contexts on her web page<sup>13</sup> and some FCA tools have contexts for unit tests (e.g., `conexp-clj`<sup>14</sup> or `concepts`<sup>15</sup>) but these are neither comprehensive nor easy to find, they have no machine-readable metadata, they are not integrated into FCA tools or libraries, and they are sometimes difficult to cite.

<sup>13</sup> <https://upriss.github.io/fca/examples.html>

<sup>14</sup> <https://github.com/tomhanika/conexp-clj/tree/dev/testing-data>

<sup>15</sup> <https://github.com/xflr6/concepts/tree/master/examples>

## 4 Analysis and Proposition

In this section, we analyze the requirements and the corresponding simple solutions for the planned FCA repository in detail. The goal of supporting the FCA community with the repository should guide us in all decisions. It is clear to us that the attempt to create a comprehensive FCA platform is doomed to failure and will result in software that resembles a jack-of-all-trades. Thus, the overall guiding principle will be KISS.

### 4.1 Parts

The main parts of the planned FCA repository are briefly listed below and their necessity is explained in short.

*Contexts:* The central entity is the formal context. All parts should be depicted explicitly, that is, objects, attributes, and incidence relation.

*Simple Statistics, Metadata and Usage:* For every context simple statistics, for example, number of objects, density, etc. shall be provided. Moreover, metadata including provenience, contributor, editor etc. is necessary. Also, the fact if a formal context is artificial or derived from real data should be noted. In addition, an overview of where a context has already been used/analyzed, for instance, in which scholarly articles, would be helpful.

*Relations:* A central element of FCA is control over scaling. Since many formal contexts are derived from non-binary data, access to the scales used is essential. Furthermore, sub-contexts are often used for analyses. This information is also important for the analyst. These and other relationships between formal contexts are to be mapped by the FCA Repo.

*Collections:* We envision that standard collections of formal contexts can be useful. For example, when evaluating a new algorithm one may employ a *standard benchmark set of contexts*. These and similar tasks require the compilation and labeling of named collections of formal contexts.

*Concepts, Lattices, Diagrams:* Although not essential, it would be very helpful to store the formal concepts belonging to a formal context, the concept lattice, or even a lattice diagram. This would increase the scientific reproducibility of results and the ecological sustainability of analyses.

*Implication Bases:* Similarly to the last point, it would be very helpful to store implication bases.

## 4.2 Implementation Considerations

Implementing the envisioned FCA repository requires storing and curating formal contexts, possibly additional data, and descriptive metadata. In this section, we discuss aspects that need to be taken into account and make specific suggestions for implementing the repository. As an overall pre-condition, we restrict ourselves to a *file-based* repository, as files are the typical format in which formal contexts are stored and used.

**Files** A file in a repository is essentially characterized by its *name*, its *location*, and its *content*.

*Name.* The name of a file acts as an identifier within a file system and must be unique at least within one directory. In addition, the file name acts as a visible identifier to everyone who is using the file. We consider two approaches for naming a file: choosing a meaningful name or simply using an arbitrary identifier (e.g., a number or UUID [20]) as name. On the one hand, a meaningful name has some benefits for humans, for example, the name can give an indication to a file’s content. On the other hand, choosing a good name is a hard problem [3], as the following Example 1 shows.

*Example 1.* Let us consider the formal context from Figure 1.1 of the FCA book [12]. The caption states that the context is from an educational film “Living Beings and Water”, so a meaningful file name could be based on the title of that film. When creating a file name from that title, immediately certain questions arise: How many words to include? How should words be separated (by space, underscore, or not at all)? Where and how to use lower/upper case characters (all lower/upper case, title capitalization, or CamelCase)? And, given that the book is the English translation of the 1996 German original [13] and the film’s original (German) title is “Lebewesen und Wasser”, we should consider which language to use: the original language or English as a common and default language.

Furthermore, the file name could also include metadata to provide more information to users, for example, an ISO language code [16] to signal the language of the object and attribute names of the context and to distinguish it from translated variants of the same context. Typically, the file name also includes a file extension which indicates the file format and thus the structure of the content. Despite the challenges, we propose to use a meaningful file name based on the content of the formal context. Our rationale is, that the repository is intended for formal contexts (i.e., files) to be downloaded and used by researchers. And in that scenario the file name should bear some meaning to the user, as they have to store and find the file on their computer’s file system. The exact format of the file name is up to discussion, but for now we suggest to keep it short, all lowercase, in English, words separated by underscore, and appending the ISO language code (to allow for distinguishing translations of contexts). We are aware that naming things comes with some power, since together with their repository URL the file names can easily become unique identifiers for the corresponding



contexts. Later on, we propose a curation policy that shall ensure conscientious naming.

*Location.* The location of a file within the repository needs to be specified. This is partly a matter of the overall structure of the repository, that is, how it is organized into directories and subdirectories. We propose to have one directory, named **contexts**, in the top-level directory of the repository that contains the files for all contexts. Additional files, that, for example, contain lattices or implication bases can be put into suitably named additional directories. Together with the base URL of the repository, the **contexts** directory and the file name specify a URL which acts as a unique identifier (and locator) for each context.

*Content.* The content of the file can be represented in various file formats [24, 25]. A discussion of their benefits and drawbacks is beyond the scope of this work, see U. Priss [25] for an overview. Since FCAStone [25] can convert between some of the more common formats, we propose to use the format introduced in ConImp by Peter Burmeister [5] as default. As a plain text format, it is easy to understand for humans but also easy to parse for machines. As we have seen in Section 3.1, it is also well supported by still maintained tools and libraries.

Files in that format are typically identified by the extension “cxt” (cf. [5, footnote 20]). Since, within the specification of the format, the names and order of objects and attributes can be freely chosen, we propose to stay as close to the original source as possible. Furthermore, we propose to use UTF-8 as text encoding for names of objects and attributes. Although at the time of conception some older tools may have only supported ASCII characters, we believe the repository should be geared towards current and future tools and use cases. Additional file formats can be provided for contexts. Since a conversion could easily be automated, we can also imagine a conversion service for the repository. Nevertheless, the Burmeister (“cxt”) format should be the gold standard.

**Metadata** Metadata shall be provided for all contexts, as it provides context and important information for humans, can simplify processing of the data, and is crucial for applications built on top of the repository (for example, an exploratory web page). These aspects are particularly important with regard to the scholarly use of the FCA repository. We discuss the *what*, *how*, and *where*.

*What.* Contexts typically have a *title* which is often used to refer to them (e.g., “Living Beings”). At least the title but typically also the names of objects and attributes are in a certain *language*. Since the repository should host well-known, published contexts, the *source* where the context was published or used should be provided. What is considered to be the source is an open question – should it be the first (published) use in formal concept analysis or the first publication of the data at all?<sup>16</sup> A *description* should provide further information, for example, the meaning of the attributes or how the context was constructed. Except

<sup>16</sup> Or something more complex, for example, comprising the citation chain from the first publication of the data to the first published use in FCA.

*language*, we consider all of these properties to be mandatory. If the *language* is not English, it must be specified. Other metadata is conceivable, for example, to model relationships between contexts (e.g., derived or translated contexts).

*How.* To store the metadata, a suitable data representation together with a serialization into a file is required. On the one hand, the amount and complexity of the metadata is tractable; on the other hand, the representation should not impose too many restrictions but support later extensions. Thus, a good trade-off between expressivity and simplicity is required.

Having a look at the proposed metadata fields for each context, *title*, *description*, *language* could be represented in natural language (preferably English). The *language* itself could be represented via an ISO language code [16]. The *source* could either focus on human-readability and, thus, be a string describing the source, for example, using a citation in a typical citation style. Otherwise, it could also be represented with explicit metadata fields that contain the full bibliographic information using, for example, the BibTeX data model [23]. Relationships between contexts could be represented by directly referring to other contexts using their (file) name(s).

The fields for each context could be represented as key-value pairs and the metadata for all contexts can be represented as a list of key-value pairs, where the keys are the names of contexts (i.e., their file names) and the values are each a list of the key-value pairs of the metadata for each context. In principle, there exist many suitable data representations, for example, RDF, XML, JSON, or YAML. Since we are aiming at a simple human-editable solution, JSON and YAML seem to be good candidates. Among those two, YAML has the additional benefit of using indentation instead of brackets which might be easier to handle for humans not trained in reading bracketed expressions. Using YAML, the context from Example 1 could be represented as follows:

```
livingbeings_en.cxt:
  title: Living Beings and Water
  source: "Ganter, B., & Wille, R. (1999). Formal Concept
  ↪ analysis. Springer, p. 18"
  language: English
  description: conditions different living beings need
```

*Where.* Like contexts, the metadata should also be file-based. Initially, we consider one file describing all contexts to be sufficient. The file shall reside in the highest level directory of the repository, next to the `contexts` directory. Another option would be to have one file with metadata for each context. Apart from doubling the number of files this would make the automatic retrieval of the information for several contexts more difficult, since the name of the contexts need to be known in advance. Furthermore, the metadata file shall also function as an index for the contexts. We are aware that the Burmeister format supports one line for a comment. However, it is unclear whether and how the current tools handle this comment and whether this line can be of any length. We therefore refrain from explicitly using this comment line.

**Curation Policy** As we have seen, there are different options on how to implement the repository and many, often subtle, decisions have to be made. To ensure consistency and ease of use, we consider it necessary to settle some decisions beforehand in the form of guidelines for implementers and contributors. We also think that such a *curation policy* will simplify and encourage contributions and – if well-considered – avoid some pitfalls (e.g., biases or compatibility problems). The policy shall cover all aspects presented in Section 4.2 and, in principle, we consider all decisions proposed there subject to discussion, for example, our suggestions for file names. Important questions are whether the policy should be regarded as ‘rules’ or as ‘guidelines’ and how it should be enforced. In Section 5.2 we propose an approach on how to settle these questions and how to implement such a policy.

**Other Aspects** We briefly address some aspects we have not discussed so far:

- The repository should support keeping a history of changes (or version information) to be able to trace changes in the data and to enable referring to a specific version (which is important for replicable research). Using the git version control system automatically solves this problem. In addition, git’s pull and fork mechanisms can contribute to record provenance and attribution.
- Another issue are legacy aspects, for example, (limited) support for UTF-8 encoding in older tools, or differences in the choice of newline character(s) among operating systems. We are open for debate how to deal with such issues. However, this is probably not a practical problem either, as it can be solved dynamically by the git system.
- We see a need for an accompanying support structure for the repository, for example, scripts to check the consistency and completeness of data and metadata, or to convert them from other formats.
- To enable tools and libraries to access the contexts remotely, a remote API needs to be provided. Using a repository based on git typically provides this automatically. Specifically, GitHub provides access using HTTPS and a REST-like API [10]. However, one might consider using a different transport protocol or providing a dedicated API (like DraCor [11] does).

## 5 Organization and Next Steps

Our activity to create the FCA repository started with a post on the fca-list mailing list<sup>17</sup> in February 2024 [17]. Subsequently, we set up a git repository as part of the fcatoools organization on GitHub<sup>18</sup> and published its web page on the domain [fcarepository.org](https://fcarepository.org).<sup>19</sup> Afterwards we uploaded twelve formal contexts together with their metadata. We describe the first steps we have taken

<sup>17</sup> [fca-list@cs.uni-kassel.de](mailto:fca-list@cs.uni-kassel.de)

<sup>18</sup> <https://github.com/fcatoools/contexts>

<sup>19</sup> <https://fcarepository.org/>

to integrate the repository into the FCA ecosystem in Section 5.3, but first we state challenges we have observed or do anticipate in Section 5.1. We envision to establish a working group, as set out in Section 5.2. Finally, we want to limit the scope of the repository in Section 5.4.

## 5.1 Challenges

In establishing and maintaining the envisioned repository we foresee several challenges that must be tackled. Among the most crucial ones are *acceptance and usage* by the FCA community. We hope to set a good foundation with our approach of early involvement of the community (e.g., our mailing list post [17] and this paper), our suggestions for implementation (e.g., prefer simple and robust approaches), and the establishment of a working group to steer the future of the repository (cf. the next section). Another, but related challenge is the *sustainability* of the repository. This also affects technical issues, for example, the choice of hosting service. We tackle this by providing a dedicated domain for the repository which simplifies migration to other hosting services. We aim to avoid a potential *bias* in the selection and description of the formal contexts by establishing procedures which involve the community in the development of a curation policy. Clearly, this list of challenges is not final (for further examples in the context of OpenML see their considerations [30]) but we consider those to be the most important ones.

## 5.2 Working Group

We think that tackling the aforementioned challenges requires a multi-stakeholder effort. Therefore, we propose to establish a *working group* which sustainably steers the extension, curation, and dissemination of the repository. The working group should be assembled by the FCA community to have their trust and representation. For practical purposes, this could happen at the 2024 CONCEPTS conference or at a later community meeting. Concrete next steps the group should pursue include

- the community-driven establishment of a curation policy (cf. Section 4.2),
- networking with other initiatives and collaboration (or federation) with related approaches (cf. Section 2),
- development of accompanying resources, for example, workflows to check the integrity of contexts and metadata, and
- establishing research data management measures, for example, the publication of regular snapshots of the repository in persistent research data repositories like Zenodo.

## 5.3 Integration into the FCA Ecosystem

Our goal is that every relevant FCA software tool provides access to the repository. Similar to, for example, machine learning libraries<sup>20</sup> we propose that FCA

<sup>20</sup> For example, scikit-learn has some basic data sets included which can be accessed with just one line of Python code: `iris = datasets.load_iris()`. Simi-

software tools allow their users to load contexts from the repository with just one line of code. We have exemplarily implemented this in a fork<sup>21</sup> of the Python library `concepts`,<sup>22</sup> in which it is possible to load the “Living Beings” context mentioned in Example 1 as follows:

```
import concepts

context = concepts.load_dataset('livingbeings_en')
```

Upon approval of a pull request<sup>23</sup> this functionality will become available to all users of this Python library. Another pull request<sup>24</sup> will integrate similar functionality into `conexp-clj`. To continue this effort, the working group shall reach out to developers of other (recent) FCA software tools and libraries and support them in integrating access to the repository.

#### 5.4 Limiting the Scope

We are fully aware that proper research data management comprises more than just setting up a git repository. However, as a small community that can not afford professional research data management [15] the proposed approach can cope with limited resources. We consider our proposal to be an important and concrete step towards actually having a repository of formal contexts for FCA. We think it is better to have an imperfect (in some respects) repository now than a perfect repository later (or never). Therefore, on purpose, we limit the scope of the repository as follows:

- a) It is centered around formal contexts and there is no intention for a comprehensive or even complete modeling of all FCA data structures.
- b) Its focus should be on the needs of the FCA community but not on the integration with other data modeling approaches, for example, linked data.
- c) It should comprise contexts that are well-known or especially important for the FCA community. By well-known we mean that a context has been used in at least one published work or within a tutorial or in other educational capacities. We refrain from including purely generated contexts. Furthermore, no arbitrary data that (also) can be interpreted as contexts should be included.

## 6 Discussion

In this paper we have outlined the first steps for (the only) real existing FCA repository. We have looked at approaches for repositories in other domains and

---

larly, Seaborn’s `load_dataset()` method loads data sets from the git repository <https://github.com/mwaskom/seaborn-data>.

<sup>21</sup> [https://github.com/rjoberon/concepts/tree/example\\_datasets](https://github.com/rjoberon/concepts/tree/example_datasets)

<sup>22</sup> <https://github.com/xflr6/concepts>

<sup>23</sup> <https://github.com/xflr6/concepts/pull/24>

<sup>24</sup> <https://github.com/tomhanika/conexp-clj/pull/141>

drawn comparisons with the tools and services in the FCA ecosystem. We then carried out an in-depth analysis of the necessary parts and their implementation of the proposed FCA repository, which led to a number of preliminary but not firm design decisions. We expect the current state of the FCA repository to be initially resilient due to its anchoring in the Github service and its representation using the non-centralised versions control system git. In addition, choosing a domain for the repository that is independent of Github is a good choice for possibly changing the underlying hosting service later.

Nevertheless, there are a number of challenges to be met and many (technical solutions) to be developed. These can only be overcome with and through the FCA community. That is why – with this work – we want to invite them to take part in this endeavor. The most crucial next step is for the community to find a reliable group of members who will take care of the repository and its further development. We hope that this will be successful at one of the next scholarly meetings.

**Acknowledgments.** We thank Melanie Seltmann and Dorothea Strecker for their valuable information and feedback on research data repositories.

## References

- [1] Mehwish Alam, Thi Nhu Nguyen Le, and Amedeo Napoli. “LatViz: A new practical tool for performing interactive exploration over concept lattices”. In: *Proceedings of 13th CLA*. Ed. by Marianne Huchard and Sergei O. Kuznetsov. Vol. 1624. CEUR Workshop Proceedings. CEUR-WS.org, 2016, pp. 9–20. URL: <https://ceur-ws.org/Vol-1624/paper1.pdf>.
- [2] Simon Andrews. “Data Conversion and Interoperability for FCA”. In: *Proceedings of the 4th Conceptual Structures Tool Interoperability Workshop at ICCS*. University of Kassel. 2009, pp. 33–43. eprint: <https://core.ac.uk/download/pdf/99874.pdf>.
- [3] Tom Benner. *Naming things : the hardest problem in software engineering*. English. Second edition. [place of publication not identified]: Independently published, 2023. ISBN: 9798366113397.
- [4] Daniel Borchmann and Tom Hanika. “Some Experimental Results on Randomly Generating Formal Contexts”. In: *Proceedings of 13th CLA*. Ed. by Marianne Huchard and Sergei O. Kuznetsov. Vol. 1624. CEUR Workshop Proceedings. CEUR-WS.org, 2016, pp. 57–69. URL: <https://ceur-ws.org/Vol-1624/paper5.pdf>.
- [5] Peter Burmeister. *Formal concept analysis with ConImp: Introduction to the basic features*. Tech. rep. Darmstadt, Germany: TU Darmstadt, 2003. URL: <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=a22862aeedefa49f548de533127591a3399dd7e>.
- [6] Pablo Cordero et al. “fcaR, Formal Concept Analysis with R.” In: *R Journal* 14.1 (2022).

- [7] Diana Cristea, Christian Sacarea, and Diana-Florina Sotropa. “FCA Tools Bundle”. In: *Suppl. Proceedings of ICFCA*. Ed. by Diana Cristea et al. Vol. 2378. CEUR Workshop Proceedings. CEUR-WS.org, 2019, pp. 50–54. URL: <https://ceur-ws.org/Vol-2378/shortAT4.pdf>.
- [8] Renita Danabalan et al. “MaRDI: Building Research Data Infrastructures for Mathematics and the Mathematical Sciences”. In: *Proceedings of the Conference on Research Data Infrastructure*. Vol. 1. 2023.
- [9] European Organization For Nuclear Research and OpenAIRE. *Zenodo*. en. 2013. DOI: 10.25495/7GXX-RD71.
- [10] Roy T. Fielding. “Architectural Styles and the Design of Network-based Software Architectures”. PhD thesis. UC, Irvine, 2000. URL: <https://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>.
- [11] Frank Fischer et al. “Programmable Corpora: Introducing DraCor, an Infrastructure for the Research on European Drama”. In: *Proceedings of DH2019: "Complexities", Utrecht, July 9–12, 2019*. Utrecht University, 2019. DOI: 10.5281/zenodo.4284002.
- [12] Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations*. Berlin/Heidelberg: Springer, 1999. DOI: 10.1007/978-3-642-59830-2.
- [13] Bernhard Ganter and Rudolf Wille. *Formale Begriffsanalyse: mathematische Grundlagen*. Berlin/Heidelberg: Springer, 1996. DOI: 10.1007/978-3-642-61450-7.
- [14] Tom Hanika and Johannes Hirth. “Conexp-Clj - A Research Tool for FCA”. In: *Suppl. Proceedings of ICFCA*. Ed. by Diana Cristea et al. Vol. 2378. CEUR Workshop Proceedings. CEUR-WS.org, 2019, pp. 70–75. URL: <https://ceur-ws.org/Vol-2378/shortAT8.pdf>.
- [15] A. Hofelich Mohr et al. *Making Research Data Publicly Accessible: Estimates of Institutional & Researcher Expenses*. Tech. rep. Association of Research Libraries, 2024. DOI: 10.29242/report.radsexpense2024.
- [16] *Code for individual languages and language groups*. Standard. Geneva, CH: International Organization for Standardization, Nov. 2023.
- [17] Robert Jäschke. *Request for Comments: a repository for formal contexts*. Message to the fca-list mailinglist. Feb. 2024. URL: <https://lists.cs.uni-kassel.de/hyperkitty/list/fca-list@cs.uni-kassel.de/message/IUIXCLRL0REWFAP0GJQFDMKBMB2HSXEF/>.
- [18] Markelle Kelly, Rachel Longjohn, and Kolby Nottingham. *The UCI Machine Learning Repository*. 2023. URL: <https://archive.ics.uci.edu>.
- [19] Jérôme Kunegis. “KONECT – The Koblenz Network Collection”. In: *Proceedings of the 22nd WWW*. WWW ’13 Companion. New York, NY, USA: ACM, 2013, pp. 1343–1350. DOI: 10.1145/2487788.2488173.
- [20] Paul J. Leach, Rich Salz, and Michael H. Mealling. *A Universally Unique Identifier (UUID) URN Namespace*. RFC 4122. July 2005. DOI: 10.17487/RFC4122.

- [21] Deepti Mittal et al. “Data management strategy for a collaborative research center”. In: *GigaScience* 12 (July 2023), giad049. ISSN: 2047-217X. DOI: [10.1093/gigascience/giad049](https://doi.org/10.1093/gigascience/giad049).
- [22] Constantinos Orphanides and George Georgiou. “FCAWarehouse, a Prototype Online Data Repository for FCA.” In: *CUBIST Workshop*. 2013, pp. 54–61.
- [23] Oren Patashnik. *BIBTEXing*. Feb. 1988. URL: <https://tug.org/texmf-docs/bibtex/btxdoc.pdf>.
- [24] Uta Priss. “FCA Software Interoperability”. In: *Proceedings of the 6th CLA*. Vol. 433. CEUR Workshop Proceedings. CEUR-WS.org, 2008, pp. 133–144. URL: <https://ceur-ws.org/Vol-433/paper11.pdf>.
- [25] Uta Priss. “FcaStone – FCA file format conversion and interoperability software”. In: *Proceedings of the 3rd CLA*. Vol. 352. CEUR Workshop Proceedings. CEUR-WS.org, 2008, pp. 33–43. URL: <https://ceur-ws.org/Vol-352/paper5.pdf>.
- [26] *re3data.org – Registry of Research Data Repositories*. last accessed: 2024-04-02. DOI: [10.17616/R3D](https://doi.org/10.17616/R3D).
- [27] Nassif Saab, Marianne Huchard, and Pierre Martin. “Evaluating Formal Concept Analysis Software for Anomaly Detection and Correction”. In: *Proceedings of the 16th CLA*. Ed. by Pablo Cordero and Ondrej Kridlo. Vol. 3308. CEUR Workshop Proceedings. CEUR-WS.org, 2022, pp. 213–218. URL: <https://ceur-ws.org/Vol-3308/Paper18.pdf>.
- [28] Xenia Specka et al. “FAIRagro: Ein Konsortium in der Nationalen Forschungsdateninfrastruktur (NFDI) Für Forschungsdaten in der Agrosystemforschung: Herausforderungen und Lösungsansätze für den Aufbau einer FAIRen Forschungsdateninfrastruktur”. In: *Informatik Spektrum* 46.1 (2023), pp. 24–35.
- [29] Thomas Tilley. “Tool Support for FCA”. In: *Proceedings of the 2nd ICFCA*. Ed. by Peter Eklund. Vol. 2961. Lecture Notes in Computer Science. Berlin/Heidelberg: Springer, 2004, pp. 104–111. DOI: [10.1007/978-3-540-24651-0\\_11](https://doi.org/10.1007/978-3-540-24651-0_11).
- [30] Joaquin Vanschoren et al. “OpenML: networked science in machine learning”. In: *ACM SIGKDD Explorations Newsletter* 15.2 (2014), pp. 49–60. DOI: <https://doi.org/10.1145/2641190.2641198>.