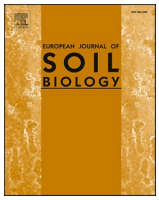
European Journal of Soil Biology 117 (2023) 103497

Contents lists available at ScienceDirect

European Journal of Soil Biology

journal homepage: www.elsevier.com/locate/ejsobi



Original article

A practical approach to constructing a knowledge graph for soil ecological research

Nicolas Le Guillarme *, Wilfried Thuiller

Univ. Grenoble Alpes, Univ. Savoie Mont Blanc, CNRS, LECA, F-38000, Grenoble, France

ARTICLE INFO

Handling editor: Anton Potapov

Keywords:
Data integration
Knowledge graph
Ontology
Reasoning
Soil ecology

ABSTRACT

With the rapid accumulation of biodiversity data, data integration has emerged as a hot topic in soil ecology. Data integration has indeed the potential to advance our knowledge of global patterns in soil biodiversity by facilitating large-scale meta-analytical studies of soil ecosystems. However, ecologists are still poorly equipped when it comes to integrating disparate datasets. In recent years, knowledge graphs have emerged as a powerful tool for integrating large amounts of distributed heterogeneous data while making these data more easily interpretable by humans and computers. This paper presents a practical approach to constructing a biodiversity knowledge graph from heterogeneous and distributed (semi-)structured data sources. To illustrate our approach, we integrate several datasets on the trophic ecology of soil organisms into a trophic knowledge graph and show how both explicit and implicit information can be retrieved from the graph to support multi-trophic studies.

1. Introduction

In recent years, a number of initiatives aiming at collecting new soil biodiversity data or assembling existing datasets have emerged, resulting in a rapid accumulation of data in soil ecology [1]. Because of the enormous phylogenetic, taxonomic and functional diversity of soil organisms, datasets are often collected by individual scientists or small project teams from different communities or disciplines to answer precise research questions. These datasets are typically small, with a limited spatial/temporal/taxonomic coverage, and are formatted according to the project needs, with little or no concern for data standardization [2]. This causes datasets to be heterogeneous in semantics (differences in terminologies, meaning or interpretation of data in different disciplines or research contexts), schema (differences in data structures and formats) and syntax (differences in models or languages). In addition, datasets are widely distributed: they reside on diverse locations, e.g. files or databases on the local network or published on the web, and are accessible using different interfaces, e.g., files downloads, database queries or web APIs.

Integrating these 'long-tail data' dispersed across different datasets could help address research questions at larger scales [3]. Data integration is of growing interest in the ecological domain, with much effort directed towards the creation of standard terminologies for describing, sharing and facilitating the aggregation of biodiversity data, e.g. organismal trait data [4–7], into large open databases. Recent initiatives in trait-based ecology have targeted specific taxonomic groups, e.g. ants [8], spiders [9], soil invertebrates [10,11], fungi [12,13], plants [14]. Although these databases have made aggregated data more readily accessible to scientists, they are not yet interoperable. The difficulty of integrating data distributed across heterogeneous sources remains. As a result, integrative analyses of soil communities that span several taxonomic groups and integrate multitrophic interactions are scarce — see Ref. [15] for an example — although essential to improve our understanding of the links between soil biodiversity and ecosystem functioning [16].

Here, we address the problem of semantic data integration in the biodiversity science domain. Data integration is defined in Ref. [17] as the process of combining data residing at different sources, and providing the user with a unified view of these data. (Fig. 1). As a result, the user has the ability to seamlessly manipulate data from multiple sources, regardless of the original format or location of the data. Semantic data integration aims at combining heterogeneous data in a way that preserves the original 'meaning' of the data in their particular semantic context. In practice, this often consists in establishing semantic correspondences (also called *mappings*) between the vocabularies of the different data sources and a common reference *ontology*. The result of this process is called a knowledge graph.

A knowledge graph (KG) is a graph-structured knowledge base that

* Corresponding author.
E-mail address: nicolas.leguillarme@univ-grenoble-alpes.fr (N. Le Guillarme).

https://doi.org/10.1016/j.ejsobi.2023.103497
Received 4 February 2023; Received in revised form 24 April 2023; Accepted 3 May 2023
Available online 16 May 2023

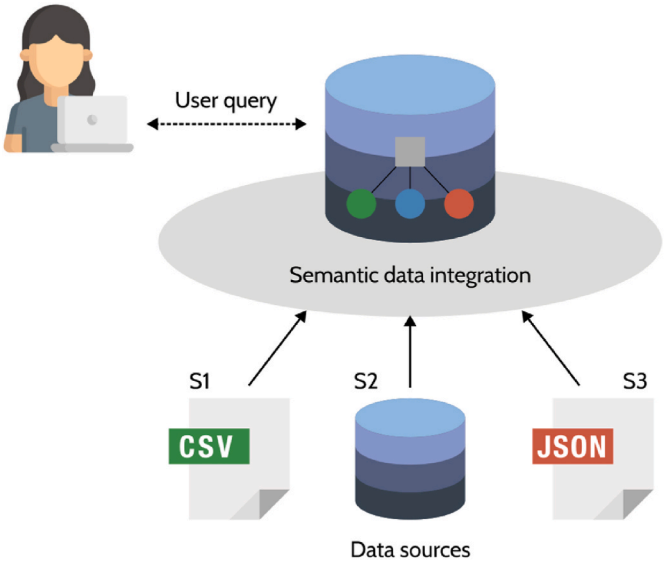


Fig. 1. Semantic data integration provides the user with a uniform access to a set of autonomous and possibly heterogeneous data sources in a particular application domain.

stores factual information in the form of relationships between real-world entities (like people, places, 'things') [18]. Under the Resource Description Framework (RDF), the standard data model of the Semantic Web, a KG is a set of (subject, predicate, object) triples. A RDF triple is a factual statement about an entity (the subject), connected to another entity or a data value (the object) by a relationship (the predicate). A set of RDF triples forms a labeled directed graph, called a RDF graph (Fig. 2). But not every RDF graph is a KG. The triples in a KG can be separated into two distinct, yet connected, layers (Fig. 3). The schema layer is the conceptual model of the KG and is described by an ontology or a collection of ontologies. An ontology is a formal shared conceptualization of a domain of interest [19]. It defines a common agreed upon terminology in terms of concepts (also called *classes*, i.e. the types of things that exist in the domain) and the relationships holding among them. An ontology is specified using a logic-based ontology language — most often the Web Ontology Language (OWL), built upon RDF — that allows both humans and computers to understand the semantics ('meaning') of the data. The data layer holds the concrete, factual data. These data are *instances* of the general concepts (classes) defined in the ontology. For example, if we were to define a class 'Article author' in a hypothetical ontology to describe the concept of 'people who write scientific papers', then 'Nicolas Le Guillarme' and 'Wilfried Thuiller' would be two instances of this class. In the context of semantic data integration, the data layer of a KG is populated with instance data from multiple sources. The ontology is used to link these disparate datasets at the schema-level, acting as a mediator for reconciling the structural and semantic heterogeneities between data sources.

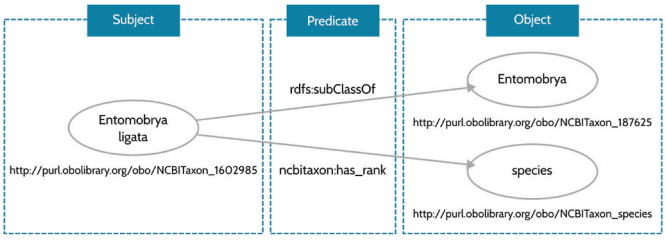


Fig. 2. The RDF data format represents factual statements about entities (here, the taxon Entomobrya ligata) as triples that consist of a subject, predicate and object. RDF triples form a labeled directed graph, which is why RDF databases are also called RDF graphs.

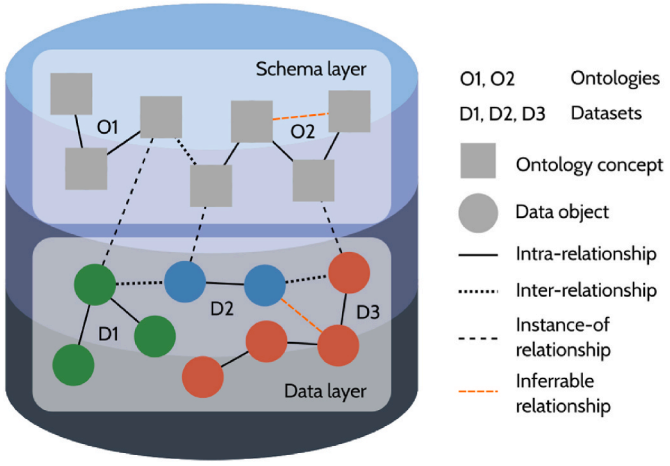


Fig. 3. A knowledge graph is a graph database that embeds both the data and its semantics in two interconnected layers. The schema layer is an ontology or a collection of ontologies that integrate datasets at the schema-level and allow logical inference of implicit knowledge using specialized softwares called reasoners. The data layer is a collection of data from various sources.

KGs have a number of advantages over other types of databases, such as relational ones. Their graph structure allows for efficient querying, intuitive visualization, and analysis using graph algorithms or relational machine learning [18]. Using an ontology as a schema layer, KGs embed a formal semantics with the data which can be used by computers to interpret and reason about the data, thus potentially allowing to infer new facts (e.g. the inferrable relationships in Fig. 2). KGs make it easy to integrate new types of data by altering the ontology or adding a new ontology to the schema layer. When following the Linked Open Data principles [20], domain-specific KGs can be easily interconnected into larger (possibly cross-domain) KGs.

KGs have recently become prevalent as a framework for semantic data integration in many different domains of science and industry [21]. It was R. Page, in his seminal 2016 paper, who first suggested the use of KGs in the biodiversity field [22]. Since then, only a few examples of biodiversity KGs have been published. Ozymandias [23] is a KG for the Australian fauna that integrates data from several sources, including the Atlas of Living Australia, the Australian Faunal Directory, the Biodiversity Heritage Library and the Biodiversity Literature Repository. OpenBiodiv [24] integrates information extracted from the biodiversity literature into a graph database using the OpenBiodiv-O ontology and an RDF version of the Global Biodiversity Information Facility (GBIF) taxonomic backbone. TAXREF-LD [25] is a KG representation of the French national taxonomical register for fauna, flora and fungus that interlinks information about taxonomy, species interactions, development stages, biogeography, conservation statuses, etc. In a recent talk at TDWG 2021, Michel et al. [26] called for more biodiversity data producers to start publishing KGs. However, for now, building a KG from multiple data sources is a complex and time-consuming task that demands high Semantic Web expertise, and we are not aware of an existing tool specifically designed to help ecologists transform their data sets into interoperable KGs — with the notable exception of the iKNOW project [27], which is very similar in spirit to our work, but whose current status is unknown to us.

In this paper, we present inteGraph, a framework and toolbox that facilitates the process of building a KG from heterogeneous and distributed (semi-)structured data sources in the biodiversity domain. With inteGraph, users can create automatic and reproducible semantic data integration pipelines simply through the provision of configuration files. This declarative approach requires no (or little) code from the user and minimizes the amount of manual effort and Semantic Web expertise required to turn datasets into interoperable KGs.

To illustrate our approach, we will show how inteGraph can be used to integrate data on the trophic ecology of soil organisms from multiple sources into a KG that can support multitrophic studies. Multitrophic studies, spanning multiple trophic levels and/or taxonomic groups, are essential to identify general patterns in community ecology [28], understand how diversity is related to ecosystem stability and ecosystem functioning [29], and provide the necessary guidance with biodiversity loss and environmental problems [30]. Multitrophic approaches should acknowledge the complexity of ecosystems while remaining practical. Large trait databases have the potential to address this trade-off between feasibility and completeness. By supporting the assignment of species (or higher taxonomic ranks) to trophic and/or functional groups, they reduce the dimensionality of ecological communities without biasing studies toward a single trophic level or taxonomic group [15,31]. Yet, some challenges still remain. Although we have trait databases available for some groups of soil organisms, our trait knowledge is limited for most of them. In addition, existing databases tend to function as data silos whose lack of interoperability can discourage researchers to include more trophic levels and/or taxonomic groups in their studies.

The ability to create a KG integrating trophic information from a number of trait databases covering different soil taxonomic groups across several trophic levels could greatly facilitate multitrophic studies in soil ecology research. Such a trophic KG would provide a unified access to multigroup, multitrophic, and multisource information. The integration of several trophic datasets (e.g. a first one containing information on carabid diets and a second one focusing on the feeding habits of springtails) into a KG allows the use of a single query to retrieve all organisms with a particular diet, regardless of the format, location or taxonomic coverage of the original data source. KGs also offer the ability to reason about the integrated data to derive additional knowledge. Reasoning about trophic interactions and dietary data opens the way for automatic classification of soil organisms into trophic groups, which can facilitate the reconstruction of consistent soil food webs from multisource data. This will be illustrated with examples in the Results section.

2. Material and methods

2.1. Overview of the approach

Fig. 4 is a high-level representation of our approach to constructing a KG from heterogeneous and distributed (semi-)structured data sources. At the heart of our framework is inteGraph,[1] an open-source toolkit for ontology-based data integration in the biodiversity domain that allows generating data integration pipelines dynamically from configuration files and scheduling and monitoring the execution of these pipelines.

2.2. Data sources

Data sources can be (and often are) distributed on several machines, on a local network and/or on the web. Data must be accessible in a (semi-)structured form, for instance as tabular (e.g. tables in relational databases or in CSV files) or hierarchical data (e.g. data in XML or JSON format). At present, inteGraph does not include information extraction components that would allow the integration of unstructured textual data from the literature.

In our running example, we will use inteGraph to build a trophic KG from the following three data sources:

- The Fun^{Fun} database [13] collates fungal functional trait data, including information about trophic guilds, from a variety of sources, for thousands of species across the fungal tree of life. Data are provided in a tabular format (CSV file) and can be downloaded from Zenodo (https://zenodo.org/record/1216257).
- BETSI [11] is an open database gathering data on morphological traits and ecological preferences for 7 taxonomic groups of soil invertebrates (Aranae, Carabidae, Chilopoda, Collembola, Diplopoda, Isopoda and Diplotesticulata) from about 2000 literature references. BETSI is accessible on demand via a web portal (https://portail.betsi.cnrs.fr) that provides the user with an interface to write queries and download subsets of the database in a tabular format (CSV file). In the following, we will integrate a dataset containing Carabidae diet data.
- The Global Biotic Interactions (GloBI) provides open access to species interaction data (e.g. predator-prey, pollinator-plant, pathogen-host, parasite-host) aggregated from existing open datasets [32]. As of April 2023, GloBI contains over 17 M interaction records obtained from 342 datasets, covering 823,033 taxa. GloBI provides several ways to access its data, including a web portal (https://www.globalbioticinteractions.org/), a downloadable snapshot of the entire database in a tabular format (CSV file), and a web API. In our example, we will use the web API to download data about the trophic interactions of Collembola.

Information from these three data sources will populate the data layer of our trophic KG once it has been transformed into a common representation.

2.3. Target ontology

InteGraph adopts a top-down approach to KG creation. In this type of approach, the schema layer of the KGis populated with a predefined ontology. The semantic data integration process then consists of populating the data layer of the KG with data extracted from the different data sources, and creating semantic links between the schema of the sources and the (global) schema of the KG using mapping rules.

To reconcile schematic and semantic heterogeneities between our trophic data sources, the schema layer of our example KG will be populated with two ontologies: the NCBITaxon ontology and the Soil Food Web Ontology (SFWO) [7]. NCBITaxon is a formal translation of the NCBI Taxonomy database into an ontology, in which each taxon is treated as a class whose instances would be individual organisms, e.g. 'Nicolas Le Guillarme' instance_of NCBITaxon_9606 (*Homo sapiens*). To our knowledge, the NCBI Taxonomy database is the only taxonomic nomenclature available as an OWL ontology. SFWO is an ontology for representing knowledge on the trophic ecology of soil organisms across taxonomic groups and trophic levels. SFWO captures the semantics of trophic concepts such as trophic interactions, feeding processes, diets or trophic groups. SFWO also includes machine-interpretable definitions for most of these concepts, that allow for inference of implicit knowledge using automated reasoning, e.g. deducing a consumer's diet(s) from the trophic interaction(s) in which it participates.

2.4. Triplestore

A triplestore is a database management system, i.e., a software used for storing and querying a database, specifically designed to support the storage and the efficient querying of RDF triples. A triplestore is needed to store both the schema and data layers of a KG. Information stored in the triplestore can be retrieved using SPARQL queries. A multitude of triplestore implementations are available (see Ref. [33] for a survey), which offer different capabilities and performance in terms of data storage and indexing, query processing, reasoning, etc.

InteGraph assumes the existence of a running triplestore instance. It is not tied to a specific implementation and provides connectors to several triplestore solutions. The user is expected to provide connection information as part of the pipeline configurations. As a top-down

[1] Available at https://github.com/nleguillarme/inteGraph.

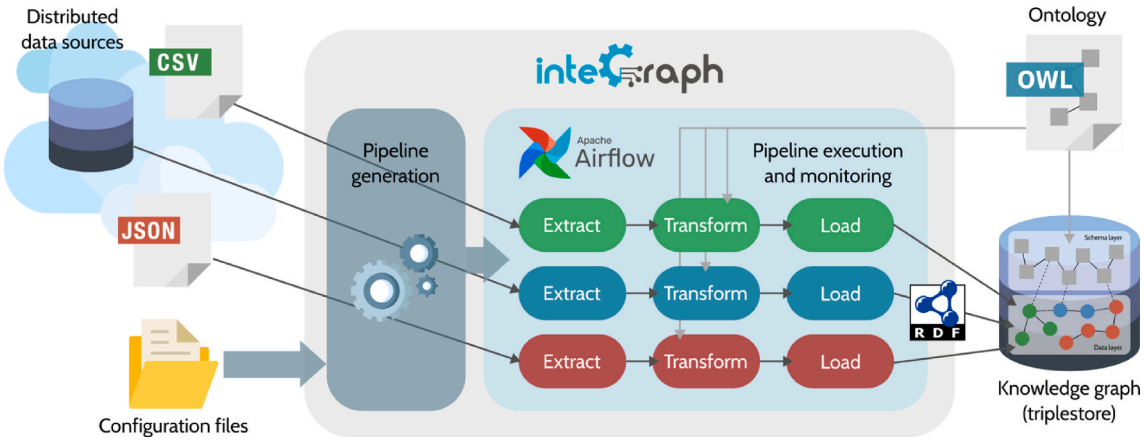


Fig. 4. A high-level representation of the proposed declarative approach for constructing a knowledge graph from distributed (semi-)structured data sources.

approach, inteGraph assumes that the target ontology has been loaded in the triplestore before the data integration process starts.

To store our example knowledge graph, we will use GraphDB Free,[2] a RDF triplestore solution that can manage billions of explicit statements on a desktop hardware, while providing optimized query evaluation and OWL reasoning.

2.5. Configuration files

InteGraph implements a declarative approach to building KGs, which means that it provides control over the creation and execution of semantic data integration pipelines using configuration files. InteGraph requires the user to provide two types of configuration files: a single graph configuration file (Fig. 5a), and a set of source configuration files, one for each data source (Fig. 5b).

The graph configuration file contains global information, including the name of the KG (that acts as a prefix to create an identifier for each graph generated from a data source), the name of the directory containing the source configuration files, the triplestore connection information, and the declaration of the target ontologies.

The source configuration file is where the user specifies the information needed by inteGraph to instantiate the Extract and Transform components of the data integration pipeline for a given source. This includes:

- the internal identifier of the data source;
- data access information, which determines the type of data source – file-like or HTTP – and the appropriate data extraction component to be added to the pipeline;
- information about the format of the input data, e.g. tab or comma-separated values;
- the path to an (optional) data cleansing script;
- for each entity (e.g. taxon, trait) in the input data, the name of the columns containing information about the entity (label and/or identifier), and a sequence of semantic annotation components whose role is to map the entity to its equivalent in the target ontology;
- the path to the spreadsheet containing the schema mapping rules.

2.6. Anatomy of inteGraph pipelines

InteGraph pipelines are structured according to the Extract-Transform-Load (ETL) paradigm. An ETL pipeline collects data from an input source (extract), cleans and maps the data from a source schema — the schema of the original data source — to a target schema (transform), and saves the transformed data into a triplestore (load). In a typical ETL process, a copy of the extracted data is stored in a data staging area and all transformations are applied to the staged data. In inteGraph, an ETL pipeline is dynamically created at runtime for each data source from the configuration files provided by the user. This ETL pipeline extracts and stages the raw data from the data source, transforms the staged data into a RDF graph, and loads the RDF graph into the data layer of the triplestore.

2.7. Data extraction

This first step of the ETL data integration process involves collecting data from the data source. InteGraph implements a number of components to connect to different types of data sources. At the moment, inteGraph supports the following types of data sources:

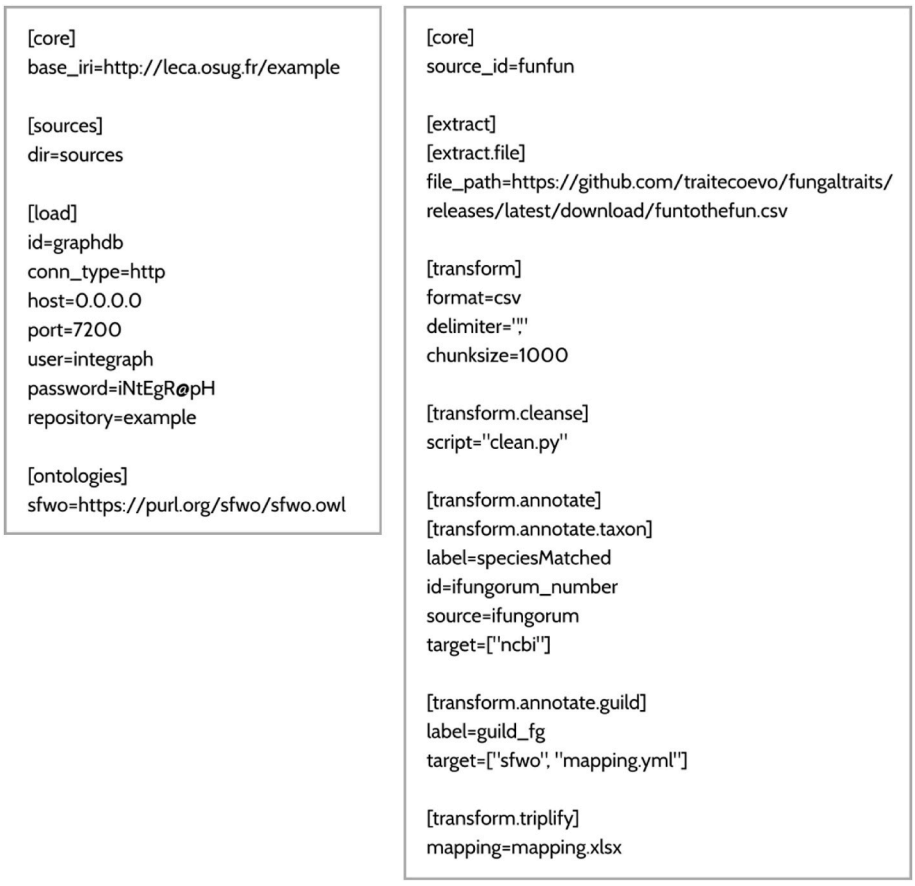
- File-like data sources: inteGraph can download files from remote or local file-like sources by specifying the local path or the URL of the source in the configuration. Archive files, including compressed archives, are supported, and unpacked before staging.
- HTTP data sources: inteGraph can extract data from remote databases exposed through a web-based API by sending HTTP GET requests to the API endpoint. In that case, the user is expected to provide the URL of the endpoint and the query string. Paginated results are supported using the limit and offset parameters.

These two components alone are sufficient to access most ecological datasets. We plan to add more connectors in the future, including connectors to SQL databases, RDF databases, etc. The extracted data are staged on the local file system.

Fig. 6a shows the data extracted from our three example data sources. The three datasets use different data structures and terminologies to organize and describe taxonomic and trophic information.

- The Fun^{Fun} database uses the Index Fungorum taxonomic nomenclature. Each line of the data table contains a single trait information for a given taxon. The name of the trait is given in the trait_name column and its value(s) is given in the value column. The terminology used to describe the guild of each taxon is inherited from the FunGuild database.
- BETSI does not encode taxonomic information using identifiers from a reference taxonomy. Taxa are designated only by their scientific name. Similar to Fun^{Fun}, each line of the data table contains a single trait information for a given taxon. The diet terminology is taken from the T-SITA thesaurus [4].

[2] https://graphdb.ontotext.com/documentation/10.2/about-graphdb.html.



(a) Graph configuration *(b) Source configuration for the Fun^{Fun} database*

Fig. 5. InteGraph implements a declarative approach to KG construction, giving the user control over the creation of data integration pipelines through simple configuration files.

- Each line in GloBI's data table contains information about a single interaction. The interaction is directed, so each line contains information about the source and target taxa (names and identifiers in an external reference taxonomy, e.g. ITIS, NCBI, GBIF …) and the interaction name. GloBi maintains a mapping between different taxonomic nomenclature internally, but each taxon in the data table is linked to a single identifier. The target of the trophic interaction can also be a non-taxonomic entity, e.g. rotten wood.

2.8. Data transformation

The second step of the ETL data integration process involves transforming the staged data into a RDF graph, i.e. a set of RDF triples. In inteGraph, data transformation consists of two successive operations: data cleansing and schema mapping.

Under the term data cleansing, we include all the dataset-specific data processing operations that aim at formatting the extracted data so that they are ready for further processing by the schema mapping component. This includes operations such as removing or filling missing values, removing duplicates, dropping irrelevant data, splitting strings (i.e. splitting a string representing a set of values, e.g. 'bacterivore-detritivore', into a set of strings, each string representing a single value, e.g. 'bacterivore', 'detritivore'), joining two or more data tables, etc. Fig. 6b shows an example of applying cleansing operations to the Fun^{Fun} dataset so that each line of the data table contains a single guild value. As possible cleansing operations are very diverse and highly dependent on the structure of the input data, they cannot be specified in the source configuration file. Instead, the user should provide a Python or R script that implements the data cleansing operations as a separate file. This script should respect some input/output constraints so that it can be ingested by inteGraph at runtime and incorporated into the ETL pipeline.

After data cleansing is complete and cleansed data are staged, the pipeline moves on to schema mapping which involves converting data from the schema of the original data source to the schema of the knowledge graph, i.e. the target ontology. Schema mapping in inteGraph consists of two successive tasks: semantic annotation and RDF graph generation.

Semantic annotation is the process of linking the input data with the concepts in the target ontology that best capture the semantics of the data (Fig. 6c). InteGraph provides several components for semantic annotation of biodiversity data. The first component maps taxonomic entities (identified by their name and/or identifier in a source taxonomic nomenclature) to a target taxonomy — in our running example, the NCBITaxon ontology. Taxonomic mapping uses GNparser [34] to parse scientific names and nomer [35] to match taxon names and identifiers to their equivalent in the target taxonomy. The second annotation component allows any entity (e.g. trait name, trait value, interaction type) to be linked to concepts in a target ontology using exact string matching. For instance, to link the term 'Plant pathogen' found in the Fun^{Fun} database to the corresponding class in the Soil Food Web Ontology, the component will retrieve all the classes whose label (or the label of one of its synonyms) matches exactly the lookup term. If a single eligible class is found, the term in the input data is annotated with this concept identifier, here SFWO:0000159 which is the identifier of the class 'plant pathogen' in SFWO. A third annotation component allows the user to provide a YAML file containing a dictionary with term: concept pairs. Semantic annotation components can be chained together

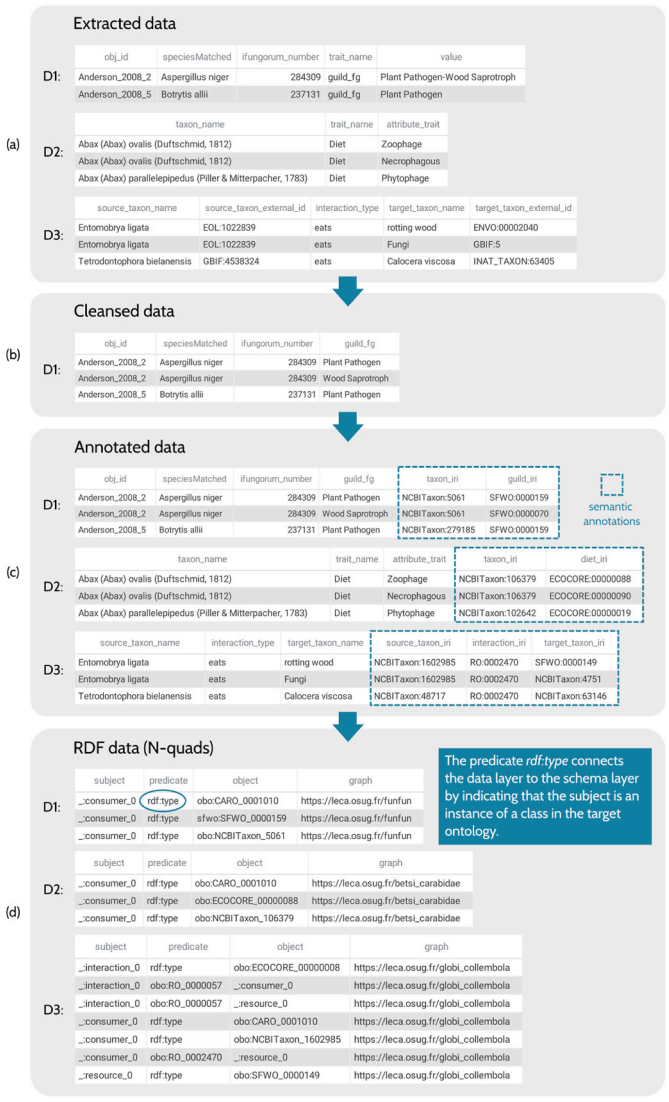


Fig. 6. An illustration of the application of data transformation to our example datasets (D1: Fun[Fun], D2: Carabidae diet data from BETSI, D3: Collembola trophic interaction data from GloBI). Fig. 6d shows the set of RDF triples (in N-quads format) generated from the first row of each data table.

to handle mismatched terms (see the source configuration file example in Fig. 5b). Fig. 6c shows the result of applying semantic annotation on our example datasets.

Once the relevant data have been linked to the corresponding concepts in the target ontology, the final step of schema mapping is the conversion of the annotated dataset into an RDF graph (Fig. 6d). InteGraph uses RDF Mapping Language (RML) [36] rules to transform tabular data into RDF triples. RML is a declarative language for expressing rules that map data in heterogeneous structures to the RDF data model. These rules describe the desired graph structure, that is how the data and schema layers of the graph should be connected to each other. The schema mapping rules should be provided by the user as part of the data source configuration. However, writing RML mapping documents is beyond the reach of most non-expert users. To face this issue, inteGraph enables to specify mapping rules in spreadsheets that are automatically translated into RML documents using Mapeathor [37] (Fig. 7). This provides a more user-friendly manner to declare mapping rules in a language-independent way. Finally, inteGraph applies Morph-KGC [38], a modern RML processing engine with a focus on speed and scalability, to execute the RML mapping rules and generate the RDF graph. Morph-KGC uses the RML rules to determine how the annotated data should be transformed into RDF triples. The RML rules are applied to each row in the annotated data to generate the RDF representation of the information in the row. In case of missing data (e.g. a taxonomic entity that could not be mapped to the target taxonomic), the RDF triples that use the missing data are not materialized. RML rules processing results in a RDF graph which is the sum of the sets of RDF triples generated for each row. Fig. 6d shows an extract of the RDF graphs obtained by applying RML rules to our example datasets. RDF graphs are staged in N-quads format, a serialization format for RDF data that associates each triple with an optional context value at the fourth position. This context value takes the form of a graph label, indicating which RDF graph the triple belongs to. This graph label is used to keep track of the data provenance (original data source) after the different RDF graphs are merged into a single KG in the triplestore.

2.9. Data loading

The third and final step of the ETL data integration process is to save the RDF graph generated during the data transformation stage in an external RDF database, i.e. a triplestore. The triplestore must be set up beforehand, either on the same machine running inteGraph or on a dedicated server. Different triplestore implementations may use different techniques to ingest RDF data. InteGraph provides connectors to the following triplestore solutions: RDFox, GraphDB, and Virtuoso. InteGraph also supports loading RDF data to a triplestore using SPARQL Update operations. In our example, the trophic KG is stored on an instance of GraphDB Free. The GraphDB connector provided by inteGraph simply loads RDF data to the triplestore using an HTTP POST request.

At the moment, inteGraph supports full load only. This means that the transformed data are loaded in full at each run of the ETL pipeline. Therefore, the KG is reconstructed from scratch every time the data integration pipelines are executed. A useful alternative would be incremental data load, i.e. updating the KG at regular intervals by loading only the data that has changed (new or updated data) since the last execution. This requires additional tools to compare the data from the data source with the existing data present in the KG. Incremental data load has a number of advantages over full load, including faster processing and preservation of data history.

2.10. Pipeline creation, scheduling and monitoring

InteGraph uses Apache Airflow[3] to schedule and monitor the execution of the data integration pipelines. Airflow provides a flexible programmatic (i.e. code-based) approach to easily build scheduled data processing pipelines as directed acyclic graphs (DAGs) of tasks. DAGs are a natural representation for ETL pipelines as each step in the ETL process is executed after the previous one has been completed (there is no circular dependency between ETL tasks). Tasks in Airflow should be atomic – they either succeed and produce some proper result or fail in a manner that does not affect the state of the system – and idempotent, i.e. rerunning a task without changing the inputs should not change the overall output.

At runtime, inteGraph parses the graph and source configuration files and creates one Airflow pipeline per data source, decomposing the full ETL pipeline into a DAG of atomic and idempotent tasks. A schedule interval can be assigned to each pipeline, which determines when and how often the pipeline is run. Alternatively, the user can manually trigger the execution of a pipeline in Airflow's graphical user interface. This interface also allows to visualize the pipelines generated by inteGraph and monitor their execution (Fig. 8). Airflow can handle failures in ETL operations by retrying them a couple of times. If the error persists, the user can easily explore the logs of the failing task, identify the cause

[3] https://airflow.apache.org/.

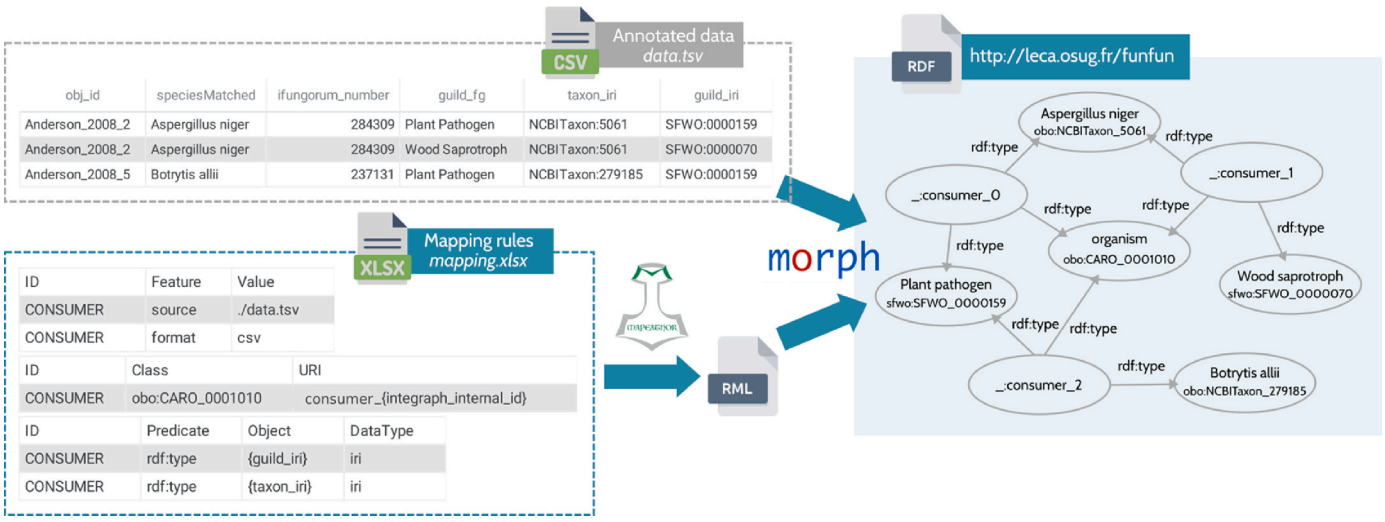


Fig. 7. InteGraph allows the user to specify schema mapping rules in a spreadsheet, which are automatically converted into RML rules using Mapeathor. The RML rules are executed using Morph-KGC to generate the RDF graph.

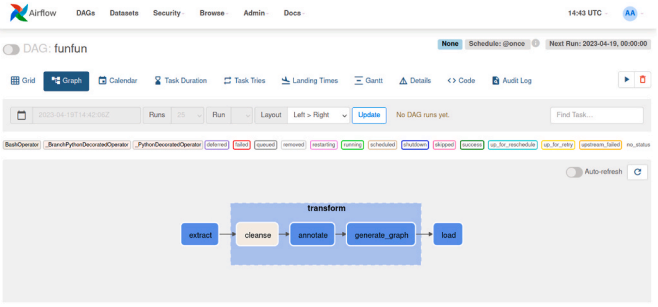


Fig. 8. A high-level view of the ETL pipeline for the Fun[Fun] database in the Airflow user interface.

of the failure, and rerun the failing task (together with any subsequent tasks that depend on that task). Airflow also has the ability to run multiple tasks in parallel. Therefore, pipelines can be executed efficiently, taking advantage of any parallelism inherent in the tasks dependency structure. For example, inteGraph can split the input data into chunks that are transformed in parallel and merged before loading, reducing pipeline execution time. In addition, ETL pipelines can run in parallel as each data source is independent of the others.

3. Results

3.1. Knowledge retrieval

At the end of the semantic data integration process, the target ontology and the transformed data are both saved in a single triplestore. The triplestore is responsible for storing the KG and executing SPARQL queries to retrieve information from it. SPARQL is a query language for retrieving and manipulating data stored in RDF format. SPARQL is based on matching graph patterns against the RDF graph. The basic graph pattern is the triple pattern, which is like a RDF triple where any part of the triple can be replaced by a variable. A graph pattern is a combination of such triple patterns. When executing a SPARQL query against a KG in a triplestore, the triplestore searches for the set(s) of triples that exactly match the graph patterns defined in the query, regardless of the provenance (i.e. the original source) of the triples, unless explicitly requested. This means that the set of RDF triples returned in response to a SPARQL query may contain facts originating from different data sources. The KG provides the user with a unified view of the original data sources through querying, and enables combining multisource information as part of a query response. Fig. 9 shows a SPARQL query searching the KG for phytophagous species. The same query returns both springtail and carabid species, whose dietary information originates from the GloBi and BETSI databases respectively. This simple example shows how a single query against the KG can retrieve information from multiple sources simultaneously, thus greatly facilitating integrative studies across taxonomic groups and/or trophic levels.

3.2. Making implicit knowledge explicit

The semantic data integration process builds a KG by linking heterogeneous datasets at the schema level using an ontology. During the process, the data layer of the KG is populated with the factual information stated in the different datasets and transformed into knowledge through semantic annotation and transformation into a RDF graph. Based on these explicit facts, additional knowledge that is not explicitly present in the data can be derived using reasoning. This ability is a direct

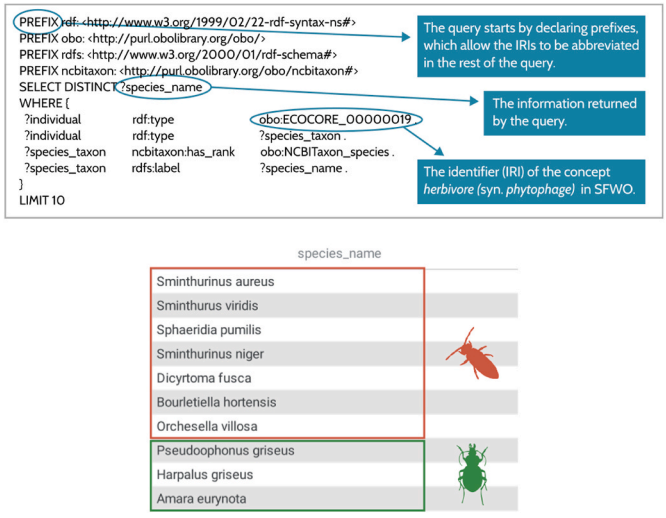


Fig. 9. Example of a SPARQL query returning the species names of phytophagous taxa. ?x denotes a variable called x. The LIMIT keyword is used to limit the number of results to the first 10 entries. The query returns information about both phytophagous springtails (from the GloBi database) and phytophagous carabid beetles (from the BETSI database).

consequence of OWL (the standard language for specifying ontologies) being based on a subset of first-order logic. Therefore, automated reasoners can be employed to evaluate the logical implications of the knowledge encoded in the ontology on the explicitly stated data.

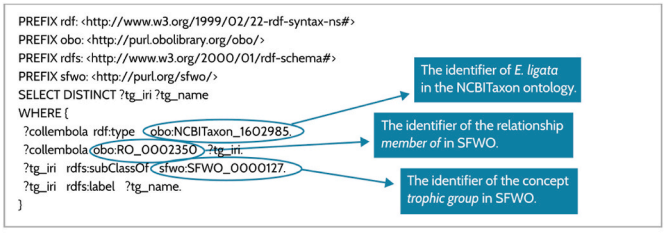
There are two principle strategies for logical inference: forward chaining and backward chaining [39]. Forward chaining, also known as materialization, derives all the facts that can be logically deduced from the existing facts and a set of logical rules, and stores these inferred facts in the triplestore for later querying. Precomputing all inferred facts enables efficient query answering, but it can also be very expensive both in time (the materialization process needs to consider all possible inferences) and memory (the process can derive a large number of facts). In addition, materialization must be redone each time the data is updated.

Backward chaining (query rewriting) starts from a query and applies the logical rules only as far as they are needed to answer the query. With backward chaining, reasoning is done at runtime and no time- and space-consuming precomputation is needed. Furthermore, no recomputation has to be done when the data is updated. However, a major drawback of backward chaining is that reasoning must be done for each new query, which can be computationally expensive and slow.

In our running example, the target ontologies (the NCBITaxon ontology and the Soil Food Web Ontology) are loaded in a triplestore supporting reasoning based on forward chaining. The Soil Food Web Ontology provides a set of logical rules that map consumer-resource interactions to diets (e.g. an animal feeding on detritus is a detritivore), as well as rules for classifying soil-associated consumers into hybrid taxonomic-trophic groups (e.g. detritivorous springtails are members of the group Collembola. detritivores). These rules make it possible to automate the process of assigning taxa to trophic groups using logical inference, thus reducing the burden of manual trophic group assignment.

Fig. 10 illustrates how information about trophic group membership is made explicit in our trophic KG using materialization. After transformed data are loaded in the triplestore at the end of the data integration pipeline, inference rules are applied repeatedly to the asserted (explicit) statements until no further inferred (implicit) statements are produced. Given (1) the explicit information about *Entomobrya ligata* feeding on rotting wood (see first line of data table D3 in Fig. 6a), (2) the hierarchy of taxonomic concepts provided by the NCBITaxon ontology, and (3) the logical rules provided by the Soil Food Web Ontology, the triplestore reasoner is able to materialize the following logical implications:

- *E. ligata* is a species of springtails (Collembola);
- *E. ligata* is a detritivore, as it feeds on rotten wood, which is a type of detritus;



tg_iri	tg_name
http://purl.org/sfwo/SFWO_0000302	Collembola.all
http://purl.org/sfwo/SFWO_0000304	Collembola.detritivores

Fig. 10. Example of a SPARQL query returning the trophic groups to which Entomobrya ligata belongs. A trophic group is defined in the Soil Food Web Ontology as 'a collection of organisms that feed on the same food sources and have the same consumers' [7,31]. SFWO provides a logical formalization of the hierarchical classification of soil consumers proposed in Ref. [40].

- *E. ligata* belongs to the group of detritivorous springtails (Collembola.detritivores) as a logical consequence of the two previous assertions.

3.3. Performance

InteGraph relies on a number of external tools with a strong focus on scalability (e.g. GNparser for scientific name parsing, Morph-KGC for RML rules execution). This, combined with Airflow's ability to run independent tasks in parallel, makes inteGraph itself quite efficient at handling large datasets in a reasonable time. In our experiments, we were able to convert tabular data with over 440 K rows into an RDF graph in about 6 min on a laptop with twelve 2.60 GHz Intel Core i7 CPUs and 16 GB of RAM. Currently, the main bottlenecks are taxonomic mapping, which in some cases may require many calls to web APIs, and logical inference, the performance of which depends on the types of reasoning and the optimisations implemented by the triplestore (inteGraph provides no reasoning component, the inference is left entirely to the triplestore).

With the ability to chain semantic annotation components, including user-provided dictionary-like mapping files, inteGraph is able to convert most of the input data into RDF, with however some entries being dropped because they cannot be linked to concepts in the target ontology. Most of the time, this happens because the taxonomic entities could not be mapped to the target taxonomy. This can be due to the source and target taxonomies being incompatible, the taxon name being ambiguous or deprecated, etc. For example, in the FunFun database, 41 of the 508 unique taxa could not be mapped to the NCBITaxon ontology, resulting in 15% of the input data being dropped during the data integration process. In the Carabidae dataset extracted from BETSI, this proportion is only 0.3% (with 18 of the 5491 unique taxa for which inteGraph could not find a correspondence in the NCBITaxon ontology).

4. Discussion

Multitrophic studies require harmonizing and integrating datasets across a large variety of taxonomic groups and trophic levels. Despite considerable efforts to make more biodiversity data freely available in a (semi-)structured format, the multiple dimensions of data heterogeneity (semantic, structural, syntactic) constitute a major obstacle to the interoperability of data sources [3]. Here, we introduced a practical approach to data integration that aims at making heterogeneous and distributed biodiversity data sources interoperable as part of a single KG. KGs provide a unified representation of disparate data sources and allow for retrieving data across these sources using a single query. By using ontologies as global schemas, they add semantics to the integrated data, making it easier for humans and computers to interpret the data and for reasoners to infer additional facts. As seen from the example discussed in the Results section, the ability to reason about the data in our trophic KG opens avenues for automatic classification of soil organisms, which can facilitate the reconstruction of consistent soil food webs from multi-source data. In addition, KGs provide support for a number of applications [41], including both in-KG, e.g. link prediction, error detection, and out-of-KG applications, e.g. relation extraction from text, recommender systems, etc.

Despite their many advantages, KG construction is currently out-of-reach for most biodiversity data providers and consumers as they require in-depth expertise in Semantic Web technologies. InteGraph is an attempt to make semantic data integration and KG construction more accessible to the biodiversity science community. Requiring little or no code and minimal knowledge of the Semantic Web, inteGraph facilitates the processes of converting a biodiversity dataset into a KG and of integrating multiple datasets into a single KG. Given a set of distributed data sources and a target ontology, inteGraph allows the user to control the creation and execution of reproducible ontology-based data integration pipelines through a set of simple configuration files. This

declarative approach relieves the user of the implementation burden. Instead, the user can focus on the desired structure of the target KG and on the schema mapping rules needed to transform the input data into RDF graphs. InteGraph relies on high-performance third-party tools (gnparser, nomer, Morph-KGC, Airflow), which guarantees a certain ability to scale to large datasets. The viability of our approach has been tested by creating a KG of soil trophic ecology from multiple open trait databases, using the Soil Food Web Ontology and the NCBITaxon ontology as the KG schema. Currently, inteGraph is at the proof-of-concept stage. It still needs some development to make it more robust, scalable and user-friendly. We also plan to add more advanced features in the future, especially regarding data provenance tracking and continuous KG updating.

Although it represents a significant advance in the field of ontology-based biodiversity data integration, inteGraph suffers limitations related to current practices in biodiversity data management. First, inteGraph requires the data sources to provide data in a (semi-)structured format through a programmatic interface, e.g. a URL to download the data file or a web API that handles HTTP requests. Still lots of data about soil biodiversity are not accessible this way, e.g. data from the BETSI database must be downloaded manually. We are confident that this situation will become less frequent in the future. Second, as a top-down approach to KG creation, inteGraph requires a predefined ontology to act as the mediating schema to link heterogeneous data sources. Creating an ontology to model knowledge in a domain of interest is a complex process that requires a significant investment of time and effort. Ontology engineering asks for a group of experts to produce a consensual conceptualization of the domain. For instance, in the domain of soil trophic ecology, this means trying to harmonize the use of diet terms that may have different meanings from one taxonomic group to another. However, we believe that the result is worth the effort, as a properly designed ontology can benefit the whole community by facilitating knowledge sharing, dataset standardization and, ultimately, data integration.

Finally, although it aims to make semantic data integration more accessible to a non-expert audience, inteGraph still requires a minimum of knowledge of Semantic Web technologies (RDF data, ontologies, SPARQL queries …). Just as the environmental community has begun to embrace new artificial intelligence tools from recent developments in deep machine learning, we encourage the community to take an interest in Semantic Web tools for better biodiversity knowledge management.

Continuing our efforts to develop more and more biodiversity ontologies [7,25,42,43] would allow us to envision increasing semantification of the ecology domain in the near future. Combined with tools such as inteGraph, which facilitate the conversion of biodiversity datasets into graph knowledge bases, these semantic resources could support the creation by different communities of numerous domain-specific KGs, which could eventually be interconnected to form a single biodiversity KG covering the entire tree of life and the full diversity of global ecosystems.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Wilfried Thuiller reports financial support was provided by European Union. Wilfried Thuiller reports financial support was provided by French National Research Agency. Wilfried Thuiller reports financial

Box 1
Glossary

Class: in an ontology, a description of a concept in the domain of interest.

A class is a set of individuals that share common characteristics, and the class definition gives the properties that these individuals must fulfill to be members of the class.

For instance, ‘bacterivore’ is the class of all individual organisms that feed on bacteria.

Extract-Transform-Load: a three-phase data integration process that combines data from multiple sources into a single central repository.

Instance (individual): a real-world realization of a concept defined in an ontology. In ontological terms, an individual is an instance of a class in the ontology.

Knowledge graph: a knowledge base that uses a graph-structured data model to integrate data.

N-quads: a line-based, plain text format for storing and transmitting RDF data. A N-quads statement is a RDF triple extended with an optional context value that takes the form of a graph label, indicating which graph the triple belongs to.

Ontology: the formal and consensual description of a domain of interest as a set of interrelated concepts.

Reasoner: a computer program that uses an ontology to infer logical consequences from a set of asserted facts.

Resource Description Framework (RDF): the standard data model of the Semantic Web. RDF represents any piece of information as subject-predicate-object triples.

RDF Mapping Language: a language for expressing mapping rules from heterogeneous data structures to the RDF data model.

Semantic data integration: the process of combining data from different sources into a single, unified view using ontologies.

Semantic Web: a set of standard technologies - including the Resource Description Framework (RDF), Web Ontology Language (OWL), and SPARQL - that help make computers better able to interpret data and information published on the web.

SPARQL: the standard query language for retrieving and manipulating data stored in RDF format.

Triplestore: a database engine optimized for the storage and retrieval of RDF data.

Web API: an interface consisting of one or more endpoints publicly exposed on the web, that allow a user to programmatically access some specific features or the data of an application, e.g. a database.

Web Ontology Language (OWL): a family of knowledge representation languages and the World Wide Web Consortium’s (W3C) standard for authoring ontologies, built on RDF and characterized by formal semantics based on description logics (decidable fragments of first-order logic).

support was provided by MIAI Grenoble Alpes. Wilfried Thuiller reports financial support was provided by Grenoble Alpes Métropole. Wilfried Thuiller reports financial support was provided by Département de l'Isère.

Data availability

Code available at https://github.com/nleguillarme/inteGraph.

Acknowledgements

We acknowledge support from the European Union's Horizon Europe under grant agreement N°101060429 (NaturaConnect), the French Agence Nationale de la Recherche through the EcoNet (ANR-18-CE02-0010), GlobNet (ANR-16-CE02-0009) and FishPredict (ANR-21-AAFI-0001) projects and the MIAI@Grenoble Alpes (ANR-19-P3IA-0003) institute.

References

[1] H.J. White, L. León-Sánchez, V.J. Burton, E.K. Cameron, T. Caruso, L. Cunha, P. Caplat, Methods and approaches to advance soil macroecology, Global Ecol. Biogeogr. 29 (10) (2020) 1674–1690.
[2] T. Poisot, A. Bruneau, A. Gonzalez, D. Gravel, P. Peres-Neto, Ecological data should not be so hard to find and reuse, Trends Ecol. Evol. 34 (6) (2019) 494–496.
[3] K. Vanderbilt, C. Gries, Integrating long-tail data: how far are we? Ecol. Inf. 64 (C) (2021).
[4] B. Pey, M.A. Laporte, J. Nahmani, A. Auclerc, Y. Capowiez, G. Caro, M. Hedde, A thesaurus for soil invertebrate trait-based approaches, PLoS One 9 (10) (2014), e108985.
[5] E. Garnier, U. Stahl, M.A. Laporte, J. Kattge, I. Mougenot, I. Kühn, S. Klotz, Towards a thesaurus of plant characteristics: an ecological contribution, J. Ecol. 105 (2) (2017) 298–309.
[6] F.D. Schneider, D. Fichtmueller, M.M. Gossner, A. Güntsch, M. Jochum, B. König-Ries, N.K. Simons, Towards an ecological trait-data standard, Methods Ecol. Evol. 10 (12) (2019) 2006–2019.
[7] N. Le Guillarme, M. Hedde, A. Potapov, M.P. Berg, M.J.I. Briones, K. Hohberg, W. Thuiller, The Soil Food Web Ontology: aligning trophic groups, processes, and resources to harmonise and automatise soil food web reconstructions, bioRxiv (2023), https://doi.org/10.1101/2023.02.03.526812.
[8] C.L. Parr, R.R. Dunn, N.J. Sanders, M.D. Weiser, M. Photakis, T.R. Bishop, H. Gibb, GlobalAnts: a new database on the geography of ant traits (Hymenoptera: formicidae), Insect Conserv. Divers. 10 (1) (2017) 5–20.
[9] S. Pekár, J.O. Wolff, Ľ. Černecká, K. Birkhofer, S. Mammola, E.C. Lowe, P. Cardoso, The World Spider Trait Database: a Centralized Global Open Repository for Curated Data on Spider Traits, Database, 2021, 2021.
[10] A. Potapov, D. Sandmann, S. Scheu, Ecotaxonomy: linking traits, taxa, individuals and samples in a flexible virtual research environment for ecological studies, Biodivers. Inf. Sci. Stand. 3 (2019), e37166.
[11] S. Joimel, J. Nahmani, M. Hedde, A. Auclerc, B. Léa, J. Bonfanti, P. Benjamin, A large database on functional traits for soil ecologists: BETSI, in: Global Symposium on Soil Bio*diversity*, 2021, April, pp. 523–528.
[12] N.A. Soudzilovskaia, S. Vaessen, M. Barcelo, J. He, S. Rahimlou, K. Abarenkov, L. Tedersoo, FungalRoot: global online database of plant mycorrhizal associations, New Phytol. 227 (3) (2020) 955–966.
[13] A.E. Zanne, K. Abarenkov, M.E. Afkhami, C.A. Aguilar-Trigueros, S. Bates, J. M. Bhatnagar, K.K. Treseder, Fungal functional ecology: bringing a trait-based approach to plant-associated fungi, Biol. Rev. 95 (2) (2020) 409–433.
[14] J. Kattge, G. Bönisch, S. Díaz, S. Lavorel, I.C. Prentice, P. Leadley, M. Cuntz, TRY plant trait database–enhanced coverage and open access, Global Change Biol. 26 (1) (2020) 119–188.
[15] I. Calderón-Sanou, L. Zinger, M. Hedde, C. Martinez-Almoyna, A. Saillard, J. Renaud, W. Thuiller, Energy and physiological tolerance explain multi-trophic soil diversity in temperate mountains, Divers. Distrib. 28 (12) (2022) 2549–2564.
[16] N. Eisenhauer, S.F. Bender, I. Calderón-Sanou, F.T. de Vries, J.J. Lembrechts, W. Thuiller, A. Potapov, Frontiers in soil ecology—insights from the world biodiversity forum 2022, J. Sustain. Agric. Environ. 1 (4) (2022) 245–261.
[17] M. Lenzerini, Data integration: a theoretical perspective, in: Proceedings of the Twenty-First ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, 2002, pp. 233–246.
[18] M. Nickel, K. Murphy, V. Tresp, E. Gabrilovich, A review of relational machine learning for knowledge graphs, Proc. IEEE 104 (1) (2015) 11–33.
[19] J.S. Madin, S. Bowers, M.P. Schildhauer, M.B. Jones, Advancing ecological research with ontologies, Trends Ecol. Evol. 23 (3) (2008) 159–168.
[20] M. Mountantonakis, Y. Tzitzikas, Large-scale semantic integration of linked data: a survey, ACM Comput. Surv. 52 (5) (2019) 1–40.
[21] V. Ryen, A. Soylu, D. Roman, Building semantic knowledge graphs from (semi-) structured data: a review, Future Internet 14 (5) (2022) 129.
[22] R.D.M. Page, Towards a biodiversity knowledge graph, Res. Ideas Outcomes 2 (2016).
[23] R.D.M. Page, Ozymandias: a biodiversity knowledge graph, PeerJ 7 (2019), e6739.
[24] L. Penev, M. Dimitrova, V. Senderov, G. Zhelezov, T. Georgiev, P. Stoev, K. Simov, OpenBiodiv: a knowledge graph for literature-extracted linked open data in biodiversity science, Publications 7 (2) (2019) 38.
[25] F. Michel, C. Faron, S. Tercerie, O. Gargominy, TAXREF-LD: Knowledge Graph of the French Taxonomic Registry, 2017-2022, https://doi.org/10.5281/zenodo.5848916.
[26] F. Michel, A. Ettorre, C. Faron, J. Kaplan, O. Gargominy, Biodiversity knowledge graphs: time to move up a gear, Biodivers. Inf. Sci. Stand. 5 (2021), e73699.
[27] S. Babalou, E. Kleinsteuber, B. El Haouni, F. Zander, D.S. Costa, J. Kattge, B. König-Ries, iKNOW-A knowledge graph management platform for the biodiversity domain, in: International Semantic Web Conference (ISWC) 2022: Posters, Demos, and Industry Tracks, 2022.
[28] P. Gaüzere, L. O'Connor, C. Botella, G. Poggiato, T. Münkemüller, L.J. Pollock, U. Brose, L. Maiorano, M.H. Harfoot, W. Thuiller, The diversity of interactions complements functional and phylogenetic facets of biodiversity, Curr. Biol. 32 (9) (2022) 2093–2100.
[29] R.M. Thompson, U. Brose, J.A. Dunne, R.O. Hall Jr., S. Hladyz, R.L. Kitching, J. M. Tylianakis, Food webs: reconciling the structure and function of biodiversity, Trends Ecol. Evol. 27 (12) (2012) 689–697.
[30] S. Seibold, M.W. Cadotte, J.S. MacIvor, S. Thorn, J. Müller, The necessity of multitrophic approaches in community ecology, Trends Ecol. Evol. 33 (10) (2018) 754–764.
[31] M. Hedde, O. Blight, M.J. Briones, J. Bonfanti, A. Brauman, M. Brondani, Y. Capowiez, A common framework for developing robust soil fauna classifications, Geoderma 426 (2022), 116073.
[32] J.H. Poelen, J.D. Simons, C.J. Mungall, Global biotic interactions: an open infrastructure to share and analyze species-interaction datasets, Ecol. Inf. 24 (2014) 148–159.
[33] W. Ali, M. Saleem, B. Yao, A. Hogan, A.C.N. Ngomo, Storage, Indexing, Query Processing, and Benchmarking in Centralized and Distributed RDF Engines: A Survey, 2020 *arXiv preprint arXiv:2009.10331*.
[34] D.Y. Mozzherin, A.A. Myltsev, D.J. Patterson, "gnparser": a powerful parser for scientific names based on Parsing Expression Grammar, BMC Bioinf. 18 (1) (2017) 1–14.
[35] J.A. Salim, J. Poelen, Globalbioticinteractions/Nomer: 0.4.8 (0.4.8), Zenodo, 2022, https://doi.org/10.5281/zenodo.7458675.
[36] A. Dimou, M. Vander Sande, P. Colpaert, R. Verborgh, E. Mannens, R. Van de Walle, RML: a generic language for integrated RDF mappings of heterogeneous data, in: Proceedings of the Workshop on Linked Data on the Web Co-located with the 23rd International World Wide Web Conference (WWW 2014), 2014. Seoul, Korea. (Accessed 8 April 2014).
[37] A. Iglesias-Molina, L. Pozo-Gilo, D. Dona, E. Ruckhaus, D. Chaves-Fraga, O. Corcho, Mapeathor: simplifying the specification of declarative rules for knowledge graph construction, in: ISWC (Demos/Industry), 2020, January.
[38] J. Arenas-Guerrero, D. Chaves-Fraga, J. Toledo, M.S. Pérez, O. Corcho, Morph-KGC: scalable knowledge graph materialization with mapping partitions, Semantic Web (2022) 1–20. Preprint.
[39] G. Antoniou, S. Batsakis, R. Mutharaju, J.Z. Pan, G. Qi, I. Tachmazidis, Z. Zhou, A survey of large-scale reasoning on the web of data, Knowl. Eng. Rev. 33 (2018).
[40] A.M. Potapov, F. Beaulieu, K. Birkhofer, S.L. Bluhm, M.I. Degtyarev, M. Devetter, S. Scheu, Feeding habits and multifunctional classification of soil-associated consumers from protists to vertebrates, Biol. Rev. 97 (3) (2022) 1057–1117.
[41] Q. Wang, Z. Mao, B. Wang, L. Guo, Knowledge graph embedding: a survey of approaches and applications, IEEE Trans. Knowl. Data Eng. 29 (12) (2017) 2724–2743.
[42] R.L. Walls, J. Deck, R. Guralnick, S. Baskauf, R. Beaman, S. Blum, J. Wooley, Semantics in support of biodiversity knowledge discovery: an introduction to the biological collections ontology and related ontologies, PLoS One 9 (3) (2014), e89606.
[43] N. Abdelmageed, A. Algergawy, S. Samuel, B. König-Ries, BiodivOnto: towards a core ontology for biodiversity, in: The Semantic Web: ESWC 2021 Satellite Events: Virtual Event, vol. 18, Springer International Publishing, 2021, pp. 3–8. June 6–10, 2021, Revised Selected Papers.