Chapter 7
The Use of Data Integration and Knowledge Graphs in Modern Molecular Plant Breeding

Bjoern Oest Hansen, Jan Taubert, and Thomas Thiel

Abstract Feeding nearly ten billion people by 2050 requires a year-on-year yield increase of major field crops of about 1–2%, while arable land will likely decrease due to urbanization and climate change. On the other hand, developing a new crop variety traditionally can take up to 10–12 years. To speed up molecular breeding new ways of harnessing breeding information, including state-of-the-art statistical methods and predicting candidate genes as targets for breeding from massive amounts of data are required. As most of the necessary data is still buried in thousands of public and proprietary databases, siloed in legacy systems or can only be found in spreadsheets, novel approaches in data integration to overcome these challenges are needed. Here we describe our approach of using workflow-driven data integration and knowledge graphs in an industrial application at one of the world's leading plant breeding companies.

We adopt state-of-the-art statistical approaches for plant breeding and apply them on public and in-house generated and expert-curated data from different data domains that date back to more than a decade. For this we use a customized instance of the open-source Galaxy computational platform and analyze breeding data in a workflow-driven approach. We also shed some light on the challenges of in-house deployment of open-source tools in an industrial application, as well as ensuring software quality and coding standards for own developments.

We apply knowledge graphs in knowledge discovery use-cases to show some benefits of handling ontology-enriched in-house data as a structured graph. Here it is possible to extract information related to connections, communities in the data, infer new edges, or look for complex patterns across the graph and to perform tasks that would have been highly complex and time consuming on a silo-based data information system.

Nevertheless, the challenge of ever-increasing data in breeding information remains and necessitates the combination of different approaches to continuously drive value from data.

B. O. Hansen · J. Taubert (✉) · T. Thiel
KWS Saat SE & Co. KGaA, Einbeck, Germany
e-mail: Jan.Taubert@kws.com

M. Chen, R. Hofestädt (eds.), *Integrative Bioinformatics*,
https://doi.org/10.1007/978-981-16-6795-4_7

Keywords Data integration · Knowledge graph · Workflow · Galaxy · Ontologies · Open-source · Plant breeding · Industry

7.1 Introduction

Ensuring sustainable food supply for an increasing world population of nearly ten billion people by 2050 (see United Nations, Department of Economic and Social Affairs, https://www.un.org/development/desa/en/news/population/world-population-prospects-2019.html) requires significant progress in plant breeding and farming practices across the whole world. Climate change and the scarcity of arable land are set to impact food production in the foreseeable future. As part of the EU Green Deal, the Farm to Fork strategy sets out ambitious goals for agriculture in the coming years. These goals (see European Commission, Farm to Fork Strategy Action Plan 2020, https://ec.europa.eu/food/sites/food/files/safety/docs/f2f_action-plan_2020_strategy-info_en.pdf) include a reduction in the use of chemical plant protection by 50%, a reduction in the use of fertilizers by 20%, and the use of organic farming practices on at least 30% of farmland in the EU by 2030. On the other hand, developing a new crop variety traditionally can take up to 10–12 years.

To speed up plant breeding with the use of molecular technologies, new ways of harnessing breeding information, including state-of-the-art statistical methods and predicting candidate genes as targets for breeding from massive amounts of data are required. As most of the necessary data is still buried in thousands of public and proprietary databases, siloed in legacy systems or can only be found in spreadsheets, novel approaches in data integration to overcome these challenges are needed. Here we describe our approach of using workflow-driven data integration and knowledge graphs in an industrial application at one of the world's leading plant breeding companies KWS (see Box 7.1).

We adopt state-of-the-art statistical approaches for plant breeding and apply them to public and in-house generated and expert-curated data from different data domains that date back to more than a decade. For this we use a customized instance of the open-source Galaxy (Blankenberg et al. 2010) computational platform and analyze breeding data in a workflow-driven approach. The use of open-source software in the industry requires paying attention to the associated license terms and how such software is integrated into an industry application context. Furthermore, to ensure a high quality of own developed functionality a staged code quality and release process has been implemented with the goal to ensure high productivity in routine data analysis applications.

The challenges of integrating data across different types, from different years and different domains (e.g., genotypic, and phenotypic data) can then be addressed using workflows in Galaxy. Proprietary tools providing data from several in-house data

silos, together with applying public analysis tools are demonstrated in a genome-wide association study (GWAS) use case. Furthermore, the results of the GWAS study can be embedded in a wider context using data from a knowledge graph database.

Graphs are among the most flexible formats for a data structure. In a graph, information is described as a network of nodes and links between them, rather than tables with rows and columns. Both the nodes and edges can also have attributes assigned to them. Graph-based systems are easier to expand, as their schemas are not as strict as classical relational databases. In knowledge discovery research, this is a huge advantage. The term Knowledge graph was coined by Google in 2012, even though the topic itself has been around for longer. Though there is no formal definition of a knowledge graph, it is often described as a semantically enriched graph, supported by ontologies for standardizing the semantics. This allows for machine-readable meaning to be integrated with the data. By handling data as a structured graph, other benefits appear, it is possible to extract information related to connections, communities in the data, infer new edges, or look for complex patterns across the graph. It also becomes possible to perform tasks that would have been highly complex and time consuming on a silo-based data information system.

This combination of highly automated workflow-driven processing of genotypic and phenotypic data in plant breeding applications combined with a flexible exploration of the surrounding context of such results using knowledge graphs is supporting the decision-making of breeders at KWS. With better decision-making, plant breeding can improve the genetic potential of all crops to tackle the challenges of climate change, reduction of inputs, zero(low) chem ag and organic farming practices with the goal to provide the best seeds to our customers, the farmers.

Box 7.1 About KWS

About KWS

KWS is one of the world's leading plant breeding companies. With the tradition of family ownership, KWS has operated independently for more than 160 years. It focuses on plant breeding and the production and sale of seed for corn, sugar beet, cereals, potato, rapeseed, sunflowers, and vegetables. KWS breeding programs aim to offer every farmer—whether they use conventional or organic farming methods—targeted varieties and solutions to fit their operational needs, while also optimally tailored to the climatic conditions and specific geological conditions of their respective regions. This is the basis for efficient and productive agriculture. KWS uses leading-edge plant breeding methods. 5700 employees represent KWS in more than 70 countries.

Source: https://www.kws.com

7.2 Methods and Implementation

7.2.1 Deploying the Galaxy System in an Industry Application Context

The open-source Galaxy system developed by the Galaxy Project (Blankenberg et al. 2010, https://galaxyproject.org/) is a web-based platform for accessible, reproducible, and transparent computational research. The software is licensed (see https://galaxyproject.org/admin/license/) under the Academic Free License version 3.0 and images and documentation are licensed under the Creative Commons Attribution 3.0 (CC BY 3.0) License, which in principle allows deploying the Galaxy system in an industry application context (see Sect. 7.2.2). The Galaxy Project is supported in part by NSF, NHGRI, The Huck Institutes of the Life Sciences, The Institute for CyberScience at Penn State, and Johns Hopkins University. According to the Galaxy Project website (accessed January 2021) the Galaxy system is characterized by:

- **Accessible:** programming experience is not required to easily upload data, run complex tools and workflows, and visualize results.
- **Reproducible:** Galaxy captures information so that you do not have to; any user can repeat and understand a complete computational analysis, from tool parameters to the dependency tree.
- **Transparent:** Users share and publish their histories, workflows, and visualizations via the web.
- **Community centered:** Our inclusive and diverse users (developers, educators, researchers, clinicians, etc.) are empowered to share their findings.

The Galaxy system (Blankenberg et al. 2010) is publicly available at https://usegalaxy.org. As an important free and publicly accessible resource, it cannot offer encrypted data transfer and data storage and scalability. For most applications in an industry context, data integrity, data security and know-how protection are major concerns. Therefore, the preferred way would be to run your own Galaxy instance either on-premises or in your private cloud environment. This provides additional possibilities of closely integrating the Galaxy system with other in-house data resources, compute environments and storage systems.

Depending on the importance of the Galaxy system for data analysis needs at the company and resulting requirements to provide the system as a service to users, a more sophisticated setup than a single Galaxy system instance can be chosen. From our experience deploying the Galaxy system in an industry application context at a major plant breeding company, we recommend a setup that involves three Galaxy instances (Table 7.1): A test instance serves mainly for early testing by in-house users as well as the establishment and fine-tuning of Galaxy workflows. The productive Galaxy instance hosts tools and workflows suitable for routine operation with respect to fault tolerance and performance optimization. A third

Table 7.1 Recommended setup with three Galaxy instances

ID	Description	Users	Features	Updates
dev	Galaxy development system to follow main Galaxy branch closely and test/develop new Galaxy platform features	Galaxy in-house infrastructure team and some script developers	All required Galaxy tools installed, following the latest tool versions, possibly local new tool development	Very frequently, all Galaxy releases
test	Galaxy test system for Galaxy tools, not for testing Galaxy platform features	Above and certain Galaxy test users	All required Galaxy tools installed via Galaxy tool-shed, usually latest tool versions	Frequently, might skip minor releases
prod	Galaxy production system for routine high-performance data analysis workflows	All Galaxy users	Stable versions of required Galaxy tools for productive workflows installed via tool-shed, availability monitoring	Only major releases, maintenance window for updates

Table 7.2 Categories of development related to the Galaxy system

Category	Examples	Contributors	Distribution
Galaxy platform	Add new authentication mechanisms to Galaxy (e.g., OKTA), add more interfaces to compute cluster (e.g., IBM LSF)	Galaxy in-house infrastructure team and community developers	Submission to main Galaxy branch after community review
Public tools	Fixes to publicly available Galaxy tools (public tool-shed)	Script developers (internal and external)	In accordance with public tool owner
Proprietary tools	Specific tools for routine data analysis workflows (e.g., genomic selection)	Internal script developers (e.g., Biostatisticians)	Non-public, company confidential

Galaxy instance (Galaxy dev) serves for testing new Galaxy releases and in-house tool development.

As the Galaxy platform supports the management and installation of Galaxy tools via the Galaxy tool-shed, a local tool-shed instance is used to provide proprietary tools to the Galaxy instances. The Galaxy public tool-sheds are integrated to use and update publicly available tools for Galaxy. This allows for the clear separation of development in three major categories, see Table 7.2.

Another advantage of using a local tool-shed is the integration possibility into continuous code integration and deployment pipelines (CI/CD). Modern source code management platforms, like GitLab (see https://about.gitlab.com/) facilitate the setup of CI/CD pipelines which upon code submission to a tool repository manage the assembly, testing and deployment of changes to proprietary Galaxy tools to in-house Galaxy instances via the Galaxy tool-shed automatically. This automation ensures a high quality of the tools by performing unit testing and integration testing, as well as convenience for the script developers, which do not have to manually deploy tools to the Galaxy instances anymore. Provision of multiple Galaxy instances with different levels of productivity allows fine-tuning of CI/CD pipelines with respect to code quality and release speed for in-house Galaxy tool developments. Overall, this approach results in quality improvements, time savings and faster availability of features for the users of routine data analysis workflows (Fig. 7.1).

To be able to scale routine data analysis to multiple compute nodes beside the main Galaxy instance, a high-performance compute cluster is used. Galaxy schedules the jobs and submits these into different queues of the cluster. The cluster queue is determined by which tool should be run. For new tools, this mapping will be updated once a tool is in production usage. Factors like the number of CPUs or memory used on average by the tool will determine which queue it will be assigned to. For low memory consuming and quick running tools, e.g., data upload, a queue with a high priority is used so that the user will get tool run results almost immediately. For long running and high memory consuming tools, a lower priority queue is chosen, so that the impact of these long analyses on the overall Galaxy performance is mitigated. However, the users are made aware that such analysis might not finish when expected depending on the average cluster load. To achieve transparency, the status of the compute cluster is reported to the users on the Galaxy starting page (Fig. 7.2).

As an open-source software, the development of Galaxy largely depends on an active Galaxy community. To follow the latest developments timely, we also actively participate in the development of the Galaxy platform by submitting work items ("issues") for the public Galaxy repository, which are then being integrated into future Galaxy versions. Additionally, we contribute with own code submissions to the general Galaxy platform (Afgan et al. 2018).

7.2.2 Implications of Open-Source Licenses on the Use of Open-Source Software in the Industry

The use of free and open-source software (FOSS) in the industry is steadily increasing, driven not just by in most cases the absence of a license fee, but also by the highly innovative character of some FOSS products and packages, especially when it comes to addressing scientific challenges. However, besides these advantages

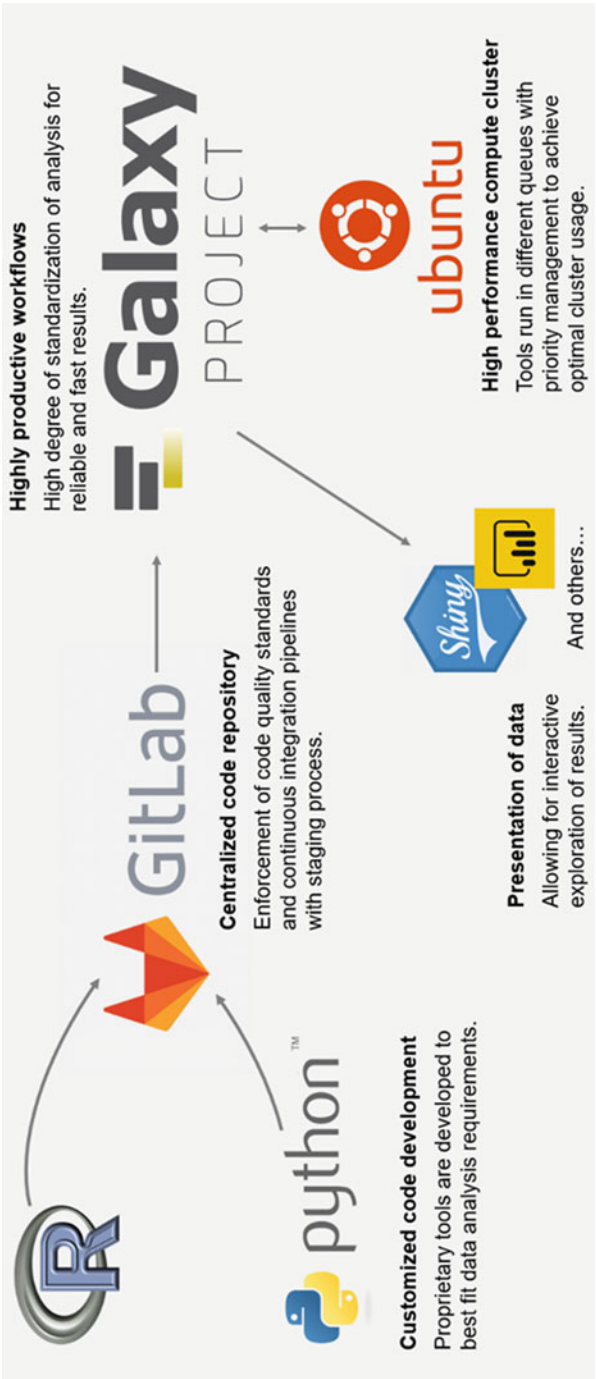

Fig. 7.1 Overview of components of an example on-premises Galaxy deployment

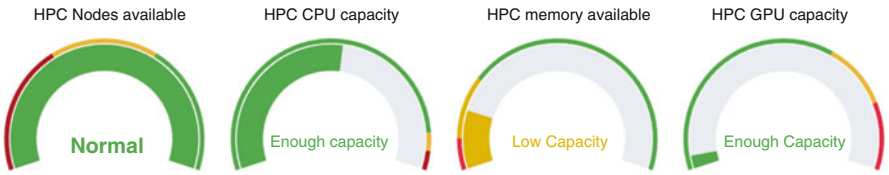

Fig. 7.2 Grafana-based reporting of HPC resources on local Galaxy starting page

there might come some pitfalls with associate open-source licenses, which need to be looked at closely. The Open-Source Initiative (OSI, https://opensource.org/) lists at least 104 OSI approved licenses, which might exist in different versions, qualities or have even been already retired. An open-source license ensures not just access to the source code, but in most cases also the free redistribution of the program, rules concerning derived work, proper acknowledgment of the code authors, some limitation of liability and the further distribution of the license itself. Here different license types include different rights and duties.

As a rule of thumb, just using unmodified FOSS programs in most cases can be seen as uncritical in industry, except for programs with licenses like GNU Affero General Public License (AGPL) or licenses explicitly restricting commercial use. Attention needs to be paid when modifying or further distributing FOSS programs with certain licenses, especially those with "Copyleft" clauses. As the GNU project supported by the Free Software Foundation states (https://www.gnu.org/licenses/copyleft.en.html): "Copyleft is a general method for making a program (or other work) free (in the sense of freedom, not "zero price"), and requiring all modified and extended versions of the program to be free as well." Such requirement can in some cases develop a viral character on additions made to FOSS programs and in the case of some licenses even on patents held by a company.

As know-how and intellectual property protection is a major concern in most industry, such cases need to be dealt with great care and attention. One prominent example is the violation of the GNU GPL license of the Linux kernel as part of the FRITZ!Box router operation system by AVM (https://avm.de) in 2011, which led to a lawsuit and finally to the distribution of AVM modifications under the same license conditions (see https://fsfe.org/activities/avm-gpl-violation/avm-gpl-violation.en.html). In this case, competitors of AVM could have benefitted from insights gained from the released source code.

Table 7.3 gives without any warranty of correctness or liability some examples of open-source licenses together with a "traffic light" indication of the perceived criticality of their usage in industry. In any case, it is advisable to get an expert opinion on the legal implications of each license in combination with the intended use. In general, to avoid possible future complications with open-source licenses in industry applications try avoiding strong Copyleft licenses (e.g., AGPL, GPL). If at all necessary, use unmodified libraries and executables called via "exec" or "fork" in the case of GPL, which are not distributed or bundled together with an industry application. In some cases, it is also possible that FOSS is available under different

Table 7.3 Examples of open-source licenses

Identifier	Name	Link	Usage
AFL 3.0	Academic Free License	https://opensource.org/licenses/AFL-3.0	Ok
AGPL v3	GNU Affero General Public License	https://opensource.org/licenses/AGPL-3.0	Critical
Apache 2.0	Apache License by Apache Software Foundation	https://opensource.org/licenses/Apache-2.0	Ok
BSD	Berkeley Software Distribution (3-clause)	https://opensource.org/licenses/BSD-3-Clause	Ok
CC0	Zero/public domain	https://creativecommons.org/share-your-work/public-domain/cc0/	Ok
CC-BY-NC	Creative Commons (CC)	https://creativecommons.org/licenses/by-nc/3.0/de/	Check
EPL	Eclipse Public License	https://opensource.org/licenses/EPL-2.0	Check
FreeBSD	FreeBSD License (BSD 2-clause)	https://opensource.org/licenses/BSD-2-Clause	Ok
GPLv2	GNU General Public License	https://opensource.org/licenses/GPL-2.0	Check
GPLv3	GNU General Public License	https://opensource.org/licenses/GPL-3.0	Check
IPL	IBM Public License	https://opensource.org/licenses/IPL-1.0	Check
ISC	Internet Software Consortium	https://opensource.org/licenses/ISC	Ok
LGPL v2	GNU Lesser General Public License	https://opensource.org/licenses/LGPL-2.0	Check
LGPL v3	GNU Lesser General Public License	https://opensource.org/licenses/LGPL-3.0	Check
MIT	MIT License by Massachusetts Institute of Technology	https://opensource.org/licenses/MIT	Ok
MPL	Mozilla Public License	https://opensource.org/licenses/MPL-2.0	Check
Ruby	Ruby License	https://www.ruby-lang.org/en/about/license.txt	Ok

licenses, here choose the less restrictive one, e.g., LGPL vs. GPL. For compliance reasons, it is also advisable to document the use of open-source licenses in industrial software applications. License finder tools exist (e.g., https://github.com/pivotal/LicenseFinder), which can be integrated into a continuous software build process to identify the licenses of the used software libraries.

Nevertheless, one of the original intentions of open-source licenses was that contributions by other parties help to improve the software overall for every user of the software. This principle should still be upheld even when FOSS is used in industry applications. Also, with industry application development it is possible to contribute to say more general features of a particular FOSS program, which do not constitute a competitive advantage and could be released to the general public. However, it is advisable to check the effort required and the acceptance of industry contributions to a particular FOSS program before contributing source code back to the original project. There are many examples of large companies like IBM, Google, Facebook, and many others making extensive contributions to FOSS packages. Advantages of contributing directly to FOSS packages include industry requirements becoming part of main FOSS releases and thus updates of FOSS packages require fewer modifications when deployed for industry applications.

7.2.3 Ensuring Software Quality and Code Standards for In-House Galaxy Tool Development

For highly productive data analysis workflows within the Galaxy system, it is important to ensure a high level of software quality and code standards for in-house developed functionality. Such functionality might not always be developed by professional software engineers, but also by biostatisticians, bioinformaticians, researchers, or even breeders themselves. This diversity of potential sources of customs tools to be integrated into the Galaxy system made it necessary to define a common set of rules or guidelines to which software quality adheres to:

- Increased process security (e.g., correct interpretation of analysis results).
- Similar end-user experience across several tools.
- Easy code transition between different developers (similar code structure, documentation, examples, and tests).
- Easier and faster to extend or refactor.
- Lower technical debt.

Basically, there are three code quality levels as part of these guidelines proposed (Table 7.4), which increase in requirements needed to be fulfilled by the respective software tool. Only tools with code quality level 1 (in some cases) and code quality level 2 (usually) should be considered for integration into the Galaxy system.

Table 7.4 Proposal of three levels of code quality

Level	Developers	Usage	Requirements
0	Only yourself	Prototypic, experimental, maybe throw-away code for one-time use	No recommendations
1	More than one developer	Tool used on a regular basis (more than once a week) Used by a low number of other users	Clear code structure and use of version control system Documentation of functions and potentially associated files Contains minimal working examples Passes Galaxy integration tests Preliminary performance evaluation
2	More than one developer and user support / software stewardship	An integral part of productive workflows for many users At least one other application relies on the correct operation	Standardized code structure which adheres to code style and/or templates Use of version control with CI/CD pipelines for Galaxy integration Documentation of the tool in a standard format (including external packages) Unit tests and integration tests with high coverage as part of CI/CD pipelines Realistic performance data for a variety of use-cases and test data

Only tools with code quality level 2 should be deployed to Galaxy productive instances (see Sect. 7.2.1). Tools with code quality level 1 can be deployed to Galaxy test instances for a limited number of users.

7.2.4 Ontologies for Structuring and Representing of Biological Knowledge

Ontologies are a framework for representing knowledge across a domain, in a format that is shareable and reusable. The goal is to provide standardization and structure, however standardization of terms in a domain is not enough for a successful ontology, adaptation is as important. One popular language for defining ontologies is

the Web Ontology Language (OWL, McGuinness and Van Harmelen 2004), which is built upon the resource description framework (RDF, Lassila and Swick 1998).

In the life science area, the "The Open Biological and Biomedical Ontologies (OBO) Foundry" (Smith et al. 2007) is a group of people working together to develop and maintain ontologies related to the field. They define principles for ontology development. More than 150 ontologies follow their guidelines.

In agribusiness, some of the important ones are the Agronomy Ontology (Jonquet et al. 2018), Plant Ontology (Bruskiewich et al. 2002, www.plantontology.org), Gene Ontology (Ashburner et al. 2000, www.geneontology.org), Crop Ontology (Shrestha et al. 2012, www.cropontology.org), Environment Ontology (Buttigieg et al. 2013, www.environmentontology.org) and Plant Trait Ontology (Arnaud et al. 2012, www.planteome.org). It is important to understand the structure of the ontology when working with it, for example the Gene Ontology, which is developed to describe the function of a gene product and contains three distinct graphs, one for functional domain: Cellular Component (where in the cell is the gene product active), Molecular Function (what is the specific function of the gene product), Biological Process (in what process is the gene product active). All of them are Directed Acyclic Graphs (DAG), which means that the edges in the graph have a direction, but there are no cycles: the direction is always one way. Standardized schemas are recommended whenever possible. Schema.org and in this case bioschemas.org would be a good place to start.

The relationships of ontological terms also encode knowledge, and they contain rules on how to traverse the relationships, for example on a hieratical structure it is possible to apply the "true path" rule, meaning that if something is annotated with a child term, all the parent terms are also implicit assigned. This could for example be if you have taken a sample from "vascular leaf," then you have indirectly also taken a sample from "leaf." This can be utilized when integrating data on multiple levels, for example, one dataset is measured in *vascular_leaf* "PO_0009025" and the other with *leaf* "PO_0025034", one can easily identify that *vascular_leaf* is a subterm of *leaf* and you can generalize to the nearest common ancestor. The same would be if we were to integrate *non_vascular_leaf* and *vascular_leaf*, the nearest common also be leaf, an example of this structure can be seen in Fig 7.3.

7.2.5 Using Knowledge Graphs for Linking Information Together

Recently another method for data integration has gained popularity, the knowledge graph. Have you ever asked a question on Google? Or used Alexa, SIRI, or Cortana? Then you most likely have been taking advantage of a knowledge graph, maybe without even knowing. The concept has existed since the 1980s but got traction when Google introduced their Knowledge Graph in a blog post in 2012. They described it as *"... we've been working on an intelligent model — in geek-speak, a 'graph' — that understands real-world entities and their relationships to one another: things, not strings."*.

A graph or a network as it is often called when referring to the practical use like social networks, or information networks, is a representation of data as entities with connections between them. In mathematical terms, an entity is a node or vertex, and a connection is an edge. A collection of nodes or vertices V together with a collection of edges E form a graph G = (V, E).

Graphs can be directed or undirected, for example, a network representing the co-occurrence of proteins in a cell is undirected, whereas a social network like Twitter is directed since the following is not reciprocal. The edges can also be unweighted or weighted, meaning that each edge has a weight assigned based on its importance.

Graphs are often visualized by drawing a point or circle for every vertex and drawing a line between two vertices if they are connected by an edge. If the graph is directed, the direction is indicated by drawing an arrow. Likewise, the weight of the edge is often represented by the thickness of the line between the vertices. Graphs allow the mathematical field of graph theory to be used when analyzing them. This could, for example, be looking at the number of connections for a node also known as the degree, or finding the shortest path between two nodes. Googles build their business around their Page Rank algorithm (Page et al. 1999), which identifies important websites among a network of websites linking to each other, which could also be seen as identifying the importance of a node based on its connections.

Emerging in the area of semantic web knowledge graphs are now seen widespread usage across many fields. There is no formal definition of a knowledge graph, though attempts have been made, one is by Ehrlinger and Wöß (2016), which define *"A knowledge graph acquires and integrates information into an ontology and applies a reasoner to derive new knowledge."*; others are less strict and consider everything that is semantically connected in a graph to be a knowledge graph (Fig. 7.3).

A knowledge graph is a representation of a knowledge domain and its logic, using a graph. It can be seen as a network of nodes of information and edges connecting them instead of tables with rows and columns. By that, people and machines can benefit from a dynamically growing semantic network of facts about things and can use it for data integration, knowledge discovery, and in-depth analyses.

It allows companies and research institutes to utilize knowledge more efficiently. In the industry, the enterprise knowledge graph is nothing more than a graph containing a precise model of business processes, with which relevant questions, facts, and events can be analyzed more quickly. Adding more information to a knowledge graph increases its value. A lot of the work originated based on the semantic web idea (Berners-Lee et al. 2001) of creating computer readable connections between data on the internet. A human being can easily distinguish how a hyperlink relates on page with another, and what the reason for the link is, a computer cannot as easily do this. To deal with this a set of specifications that are widely used also within knowledge graphs were developed, including the Resource Description Framework (RDF) Core Model, the RDF Schema language (RDF schema), the Web Ontology Language (OWL) and last the SPAQRL query language to query data in RDF format.

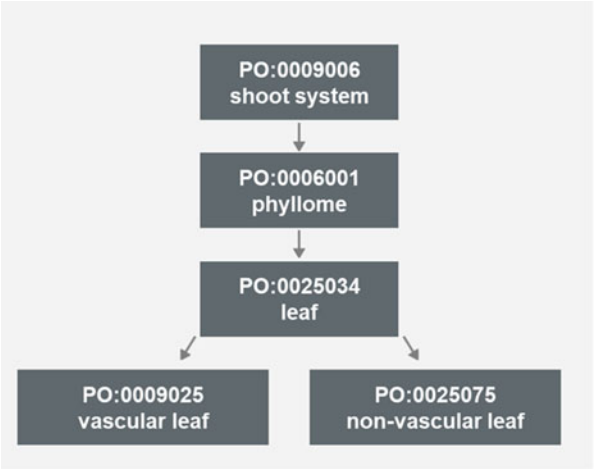

Fig. 7.3 Example of the direct children and ancestors of the ontological term "Leaf"

7.2.6 Representing Data in a Structured Format

Relational databases have been the de-facto industry standard for storing data since the 1960s. They store structured data in tables with defined columns and rows containing this data. RDBMS requires users to adhere to a schema of the data and structure their data and applications according to this.

Graphs are among the most flexible formats for data structure. In a graph, information is described as a network of nodes and links between them, rather than tables with rows and columns. Both the nodes and edges can also have attributes assigned to them. Graph-based systems are easier to expand, as they often are schemeless. It is still recommended to adhere to a schema, but it gives the flexibility of extending the schema when new data or connections arrive. There is usually not an optimal way of best modeling your data, it all depends on your question. Therefore, one should be prepared to evolve the data schema as the data and experience evolve.

Data can be modeled as graphs in multiple ways. One approach is to use the RDF standard. RDF stands for Resource Description Framework and it is a W3C standard for data exchange in the Web and is built using the existing web standards of XML and URI. It is used for describing data using relationships between objects. RDF connects data as triples, a triple is a statement about data consisting of three parts, the subject, predicate, and object. An example could be the Cellulose synthase A catalytic subunit 8 from the plant Arabidopsis thaliana, it has the id Q8LPK5 in the Uniprot (UniProt Consortium 2019) protein database. Uniprot offers API access to their data as triples. The connection between Q8LPK5 and Arabidopsis could be represented as

<http://purl.uniprot.org/uniprot/Q8LPK5>
<http://purl.uniprot.org/core/organism>
<http://purl.uniprot.org/taxonomy/3702>

The predicate in this case is from the Uniprot internal schema and is of the type organism. The definition of organism in this case is "The organism in which a protein occurs." The Subject is our protein of interest, and the organism is then defined in the Object, and is a taxonomy id referring to Arabidopsis thaliana. Another example is

<http://purl.uniprot.org/uniprot/Q8LPK5>
<http://www.w3.org/2000/01/rdf-schema#seeAlso>
<http://rdf.ebi.ac.uk/resource/ensembl.transcript/AT4G18780.1>

Where the predicate is #seeAlso from a schema provided by W3, it links according to the specification, a resource to another "that might provide additional information about the subject resource." As can be seen here, it is possible to mix URIs from different sources, one is an internal Uniprot URI, and the other is referring to one from W3. The URI serves to standardize the context and meaning, by creating a schema and definition for the connections. Just because a database is schemaless, does not mean that it should be used without schemas, it just gives the flexibility to change and expand as needed. A good place to look for public schemas is schema.org.

The other option for storing data in a graph, is the Labeled Property Graph (LPG), in LPG you have a set of nodes and edges. Both nodes and edges have a unique ID and can contain key-value pairs to characterize them.

Both are valid approaches for building a knowledge graph, and which one fits best need to be evaluated for a given use case, based on many variables, such as what questions we want to be able to answer, which infrastructure is available, what do we need regarding performance and analytics capabilities. It is also important to remember that just as not all data types fit well in relational databases, so is it also that not all fit well in graphs, there is no one-size-fits-all solution for all needs.

7.2.7 Building Your Own Knowledge Graph

When starting to think about implementing a knowledge graph in a business, it is important first to identify a need and what questions you want to answer, and how they add value to the business. Then start with a minimum viable product to demonstrate the value. Always keep stakeholders closely informed to ensure that buy-in is created.

First step is usually to gather and process relevant datasets as well as identifying necessary taxonomies, ontologies and controlled vocabularies that would serve best in achieving the goal. It is beneficial in the beginning to identify datasets that do not change often, as well as keeping size in mind. This minimizes the need to spend too

much effort on updating data and scaling infrastructure. Generally, start small and then grow when enough interest and buy-in has been created, and more resources are made available.

It is important to clean the data before uploading, remove invalid entries, adjusting dates to be in the same format etc. Then it is time to get an overview of your data and design your semantic data model using ontologies etc. on how to use data together. There is no best fit all model, it all depends on the questions you want to answer. Often the data model will evolve with your knowledge graph.

Integrate data loading with Extract Transform and Load (ETL) tools to ensure quality and consistency when moving data from one system or format to the graph, Generate Semantic metadata to make it easier to find, and reuse data. This usually goes hand in hand with a strategy for FAIR data (Wilkinson et al. 2016) (see Box. 7.2).

Augment your graph via reasoning analytics and text analysis. Enrich your data by extracting new relationships from text, apply inference algorithms to the graph to identify hidden relationships, and extend your knowledge graph with information from the graph itself. For example, degree or betweenness of nodes. It is also possible to train models to evaluate if a connection is missing, or if it is added wrong, this kind of use-case for machine learning can be beneficial especially when manual data entry has been part of the process. In the end your graph will now have more data than the sum of its constituent datasets. Lastly, set up procedures to maintain and continuously load data into the graph to keep it alive.

7.2.8 Identifying Use-Cases for Applying a Knowledge Graph-Based Approach

Identifying the ideal proof of concept use case should not be difficult, a lot of organizations have already demonstrated the effectiveness. Some inspiration for popular use-cases across industries

- Recommender systems: discovering related data and content.
- Semantic data catalogs: agile data integration and improving FAIRness of the data within the organization.
- 360 views of customers, products, employees, users etc.
- Knowledge discovery: intuitive search and analytics using natural language.

One important cornerstone to identify suitable use-cases is an active survey of potential business problems among colleagues of different areas inside your organization. Solving these business problems should generate a certain value for the company, which exceeds the costs of implementation of such knowledge graph. In our experience, workshops with a good mixture of domain experts and data experts are beneficial to identify the questions to be answered on a solid data foundation. Agile approaches, for example Event Storming (Brandolini 2013), help to reduce the discrepancy for a common understanding between domain experts

and data. Factors like data availability, quality and governance are other important factors that influence this decision. Additionally, potential use-cases could be rated by the number of people they impact. Reaching a larger audience from the beginning can help creating buy-in from more people as well as reaching people with novel use-cases.

Box 7.2 FAIR Principles
What is FAIR principles?

Findable

Metadata and data should be easy to find for both humans and computers. Machine-readable metadata are essential for automatic discovery of datasets and services, so this is an essential component of the FAIRification process.

Accessible

Once the user finds the required data, she/he needs to know how they can be accessed, possibly including authentication and authorization.

Interoperable

The data usually need to be integrated with other data. In addition, the data need to interoperate with applications or workflows for analysis, storage, and processing.

Reusable

The ultimate goal of FAIR is to optimize the reuse of data. To achieve this, metadata and data should be well-described so that they can be replicated and/or combined in different settings.

Source: https://www.go-fair.org/fair-principles/

7.3 Use-Cases

7.3.1 Using Galaxy Workflows for Ad-Hoc Data Analysis on Integrated Data

Over the years, Galaxy became more and more integrated into our research software infrastructure. We utilize specific in-house developed Galaxy tools to provide input data from different data domains such as genetic, phenotypic, OMICs data as well as genetic and genomic map data that are analyzed in different Galaxy workflows to support breeding decisions. The most common datatype utilized by public Galaxy tools is the tab-separated format. In order to use the general-purpose Galaxy tools, but also provide certain data format constraints and rule sets for specific in-house tools and user guidance, e.g., for workflow definition, we follow the somewhat pragmatic approach to define in-house data types based on the Galaxy tabular data format (see Fig. 7.4).

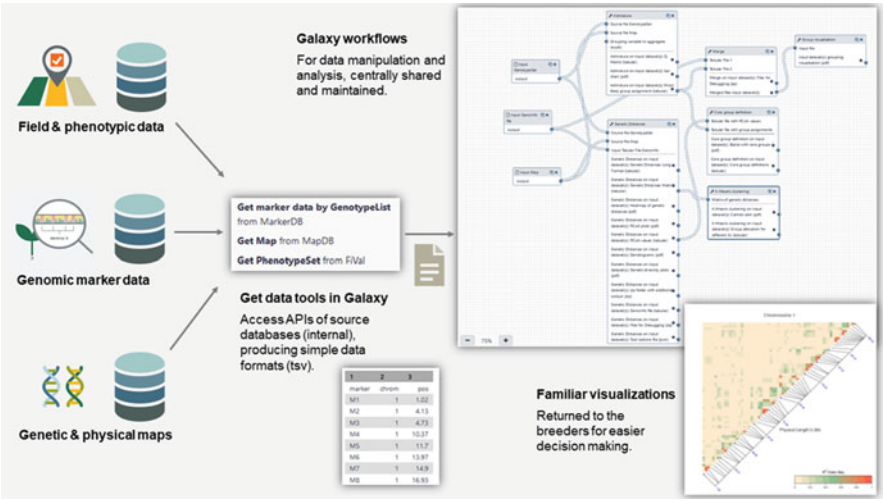

Fig. 7.4 Example of data used in Galaxy ad-hoc integration approach

More specifically, for each input data domain further analyzed in Galaxy workflows, we provide custom Galaxy tools ("Get Data" tools) that serve as connectors to other systems providing interfaces to specific integrated data. Those tools typically serve as entry points to ad-hoc analyses as part of Galaxy workflows executed in a self-service manner by our breeders. The output file(s) of the different Get Data tools are based purely on the tabular Galaxy data type for the reasons mentioned before but are specific to each data domain. This allows us to implement specific format validators on formatting and content. Additionally, this reduces errors for other in-house developed tools that depend on this specific input data, both within workflows but also for stand-alone tool runs inside Galaxy.

The genetic marker data is formatted as a named matrix (marker × genotype) which contains unphased biallelic SNP chip array data (AA, AT, AC, etc.) of the genotypes, whereas sporadic missing data is encoded as NA. Phenotypic data is encoded similarly in a genotype x trait matrix containing quantitative trait data. All SNP markers are cross-linked across different reference sequences within the different crops, thus allowing a precise location of trait-reference genome association.

Using the rich Galaxy API, we then transfer result data into downstream applications for storage and combined analysis of historic data.

To ease integration of data across sources and minimize errors, it is important that the data in each source accessed by our tools, follow the same standards and utilizes the same vocabularies and ontologies. Especially when combining with historical data, this can often be a challenge. To connect multiple heterogeneous data sources, a knowledge graph can be an advantage in ensuring that data is aggregated correctly. It allows heterogenous data to be connected with standardized machine-readable links and allows computational traversal between data sources identifying links between them and serves as a guide on where data could be aggregated.

7.3.2 *Knowledge Graphs to Enrich Genome-Wide Association Studies (GWAS) Data*

GWAS is a common approach to accelerate genomics-assisted plant breeding by detecting the genetic basis of phenotypic variation (e.g., traits of interest) on population scale based on many individuals (Tibbs Cortes et al. 2021). If certain genetic variations, usually Single Nucleotide Polymorphisms (SNPs), are found to be significantly more frequent in individuals expressing the desired trait compared to individuals that do not, the SNPs are said to be statistically associated to the trait of interest. These SNPs can serve as powerful pointers to genomic regions to assist in the selection of favorable plants for breeding and further used to support identification of candidate genes possibly involved in a certain trait. To further streamline and automate the knowledge generation in molecular breeding, we developed custom-made downstream web applications for specific approaches such as GWAS and provide APIs that allow feeding expert revised GWAS data into knowledge graphs.

In-house computed GWAS are undertaken in Galaxy on integrated genetic and phenotypic data in a way that allows traceability of the results. We then provide breeders a web-based platform to access computed results from Galaxy and store GWAS results alongside additional relevant information about the genetic material and other data in our in-house GWAS database. Finding a marker or a candidate gene is challenging. First scientists need to inspect large amounts of heterogeneous data to obtain a list of candidate genes, which then needs to converge to a ranked prediction of the most likely candidate(s) involved in the trait of interest. GWAS relies on all phenotypic data being described the same way, and accessible in the same format.

Often the SNPs cannot explain all the phenotypic variation. One reason for this is that GWAS relies on a strict P-value threshold of the SNPs after adjustment for false discovery rate to avoid false positives. This can partly be overcome by larger population sizes, however that is both costly and not always feasible. Another option is to bring in extra data to enhance it and add evidence to weaker SNPs. This could for example be gene co-expression networks, protein-protein interactions, gene regulation, protein domain information, functional information from homologues in other species, metabolic pathway information, or supporting evidence from literature.

GWAS is often applied to analyze complex traits such as resilience to drought (as opposed to monogenetic traits that follow strictly Mendelian inheritance). Associated SNPs might be distributed across many genes addressing one or more metabolic pathways. Here, trait expression can only be explained by a concerted action of multiple genetic factors that are often to a varying degree influenced by non-genetic factors such as environmental factors. To identify these, it can also be beneficial to bring in auxiliary information as described before.

A knowledge graph linking this data together with the relevant identifiers and synonyms can speed up the process of integrating this data, as well as augmenting

the results with other data sources. Enabling our researchers to get a better overview of relevant information and take information-based decisions. In the end the estimates of success need to be updated for the models being used, a feedback loop needs to be in place, for updating with experimental results on the predictions. If we want to augment the data with external sources, we need to be able to find ways of integrating this data. Sometimes there is no direct link between data points. For instance, if we want to add environmental information to our analysis for identifying candidate genes for a given trait. If the individual data points are linked, we can traverse the graph, using a graph algorithm such as Dijkstras shortest path (Dijkstra 1959). Collected data could include temperature measured at a location, a plant with a given mutation has been grown on that location during a specific time. That plant shows a particular phenotype. It is then possible to find the nodes of data where aggregation can take place to be able to connect these data and conclude about the temperature phenotype relationship.

7.3.3 Knowledge Graphs to Augment Metabolite Analysis

Plants produce a variety of small chemicals or metabolites, this could, for example, be stress hormones, measuring these metabolites is an essential part to understand more of how a given variety of plant responds. Analyzing and interpreting metabolite measurements can be time-consuming. This is a great example where knowledge graphs can assist us in making sense of the information, by augmenting the data we get out Measurement IDs, which can be matched with the corresponding metabolite, its name, synonyms, composition. It can also be linked to previous knowledge, such as literature and previous measurements.

This makes interpretation easier. At the same time, it can also be linked to internal costs of measuring, how long time does a measurement take, and what is the capacity for measuring. This can then be taken directly into context as a cost/benefit when analyzing data and deciding which metabolites are generating the most value by measuring. Questions like “What is the most optimal composition of measurements we can achieve for a given price if we want to predict a certain outcome?” can be answered. An example is seen in Fig. 7.5, a peak has been assigned with PN_10824, This can be difficult to interpret for a scientist, since this is not directly obvious what it refers to. Though, if that id was linked together with other information, it would be easy to see that it was abscisic acid, and it is a hormone that has been shown to be involved in regulating root growth. By saving the time the scientist has to spend looking for this information, and at the same time ensuring that all scientists have the same information available, we can increase efficiency and take better and more informed decisions.

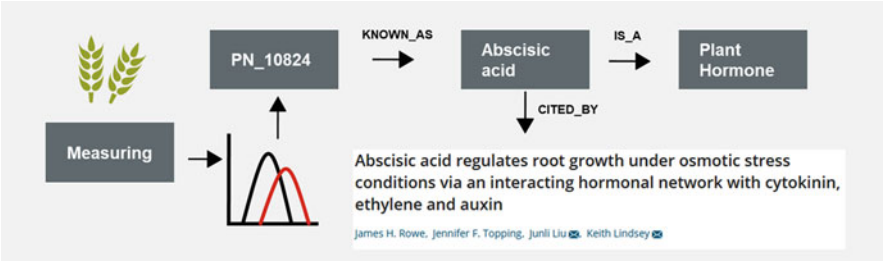

Fig. 7.5 An example of how data could be related to augment measurements of metabolites

7.4 Discussion and Conclusion

Solving the many challenges related to feeding the worlds growing population will be complex. It will involve many people and a lot of cross-disciplinary research to understand the interplay among plants, environment, people, logistics, and many other areas. Being able to handle and integrate large amounts of heterogeneous data from many sources will be an integral part of solving this challenge.

Standardized and reproducible research can help us speed up this process, by minimizing the number of errors and maximizing the utilization of the data generated. The development of the FAIR was an important step in the right direction. Novel analytical methods that can take advantage of larger and more complex datasets in the analysis are being developed. This is particularly true for machine learning, where methods such as Graph Neural Networks allow for the analysis of complex knowledge graphs. Developing more complex models and analyses could enable researchers to reach their conclusions faster with more precision. Standardized workflows and data integration is an important part of this. Since the methods are only as good as the data that goes in.

Open-source tools, standardized vocabularies and knowledge graphs are an integral part of the processes at KWS to solve these challenges. Enabling plant breeders and scientists to deliver better outcomes, storing information and as basis for improved decision-making for the future, to learn from and improve upon.

7.4.1 The Challenge of Increasing Data

It has been estimated that in 2015–2016 more data were created than in the preceding 5000 years of human history, and that amount increased so in 2017 alone, a similar amount was created. To be able to generate value, having information is not enough, the context for the information is important to be able to translate this into actionable insights and knowledge.

The data landscape in most tech businesses constantly grows more complex due to new technologies producing new measures and new tools for analyzing data. Some of it is structured data such as measurements from sensors or transactions in banks, but large amounts are unstructured, such as images, documents, relationships. A lot of knowledge is lost, due to a lack of context for the data.

One way of adding context to data is to connect it with other data. Data integration is the process of combining data from different data sources into a single, unified view. However, integrating data is one of the most time-consuming parts of a data scientist's work life.

Many enterprises suffer from data being locked in silos, making integration difficult due to different data models, descriptors, nomenclature, or unstructured data. This in the end prevents an optimal utilization of the accumulated knowledge inside an organization.

7.4.2 Combination of Approaches Needed

Data silos are a trait of many larger organizations, however, silos are a big hurdle toward many business-critical processes, for example, app development, data science, analytics, reporting and compliance. Implementing efficient enterprise data management can both decrease costs and increase performance and generate additional value for organizations and customers. There is no solution that fits all, but creating standardized pipelines and workflows, and keeping file formats as simple as possible are good rules of thumb. Providing data integration pipelines in a system like Galaxy, not only saves time for the user when they need to run an analysis, it also ensures reproducibility.

It is critical to knowledge discovery to be able to integrate different sources of data because it allows different information about the same entity to be related in new ways. A big challenge is synonymic naming and syntactically different identifiers. In a biological setting, this could be gathering different data that describe the same biological entity (e.g., gene, transcript, protein, etc.). Using ontologies can aid in the automatic integration and aggregation of data from multiple sources and ensure that data is reusable across departments. Data by itself for example in a data lake is not knowledge and has limited usage. Using graphs another layer of context can be added to the data when integrating it, this, in the end, gives more information than the sum of its parts, since the features of relationships, for example, node degree has been shown to be highly predictive as well.

Adding semantic or self-descriptive links and features to the data allows both computers to read it, but also makes onboarding of new staff members and exploratory data analysis easier since it is possible to read directly what a given piece of data represents. One way of dealing with this is to use an integration layer between the data sources and the end view. The integration layer will then be queried using for example Cypher or SPARQL, to then get the results from underlying data sources, the query will be translated into the query language of each data source. The integration layer is based on Ontologies and structured vocabularies to identify how

data should be mapped. This allows the utilization of machine learning and enables researchers to reach their conclusions faster with more precision. Standardized workflows and data integration is a crucial part of this.

References

Afgan E, Baker D, Batut B, Van Den Beek M, Bouvier D, Čech M, Chilton J, Clements D, Coraor N, Grüning BA, Guerler A (2018) The galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update. Nucleic Acids Res 46(W1):W537–W544

Arnaud E, Cooper L, Shrestha R, Menda N, Nelson RT, Matteis L, Skofic M, Bastow R, Jaiswal P, Mueller L, McLaren G (2012) Towards a reference plant trait ontology for modeling knowledge of plant traits and phenotypes. In: International Conference on Knowledge Engineering and Ontology Development, vol 2. SciTePress, Setúbal, pp 220–225

Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, Davis AP, Dolinski K, Dwight SS, Eppig JT, Harris MA (2000) Gene ontology: tool for the unification of biology. *Nat Genet* 25(1):25–29

Berners-Lee T, Hendler J, Lassila O (2001) The semantic web. Sci Am 284(5):34–43

Blankenberg D, Kuster GV, Coraor N, Ananda G, Lazarus R, Mangan M, Nekrutenko A, Taylor J (2010) Galaxy: a web-based genome analysis tool for experimentalists. Curr Protoc Mol Biol 89(1):19–10

Brandolini A (2013) Introducing event storming. goo.gl/GMzzDv. Accessed 8 Jul 2017

Bruskiewich R, Coe EH, Jaiswal P, McCouch S, Polacco M, Stein L, Vincent L, Ware D (2002) The plant ontology (TM) consortium and plant ontologies. Comp Funct Genom 3(2):137–142

Buttigieg PL, Morrison N, Smith B, Mungall CJ, Lewis SE (2013) The environment ontology: contextualising biological and biomedical entities. J Biomed Semantics 4(1):1–9

Dijkstra EW (1959) A note on two problems in connexion with graphs. Numer Math 1(1):269–271

Ehrlinger L, Wöß W (2016) Towards a definition of knowledge graphs. SEMANTiCS (Posters, Demos, SuCCESS) 48:1–4

Jonquet C, Toulet A, Arnaud E, Aubin S, Yeumo ED, Emonet V, Graybeal J, Laporte MA, Musen MA, Pesce V, Larmande P (2018) AgroPortal: a vocabulary and ontology repository for agronomy. Comput Electron Agric 144:126–143

Lassila O, Swick RR (1998) Resource description framework (RDF) model and syntax specification–W3C Recommendations. Technical report, World Wide Web Consortium. https://www.w3.org/TR/1999/REC-rdf-syntax-19990222/

McGuinness DL, Van Harmelen F (2004) OWL web ontology language overview. W3C Recommend 10(10):2004

Page L, Brin S, Motwani R, Winograd T (1999) The PageRank citation ranking: bringing order to the web. Stanford InfoLab, Stanford, CA

Shrestha R, Matteis L, Skofic M, Portugal A, McLaren G, Hyman G, Arnaud E (2012) Bridging the phenotypic and genetic data useful for integrated breeding through a data annotation using the crop ontology developed by the crop communities of practice. Front Physiol 3:326

Smith B, Ashburner M, Rosse C, Bard J, Bug W, Ceusters W, Goldberg LJ, Eilbeck K, Ireland A, Mungall CJ, Leontis N (2007) The OBO foundry: coordinated evolution of ontologies to support biomedical data integration. *Nat Biotechnol* 25(11):1251–1255

Tibbs Cortes L, Zhang Z, Yu J (2021) Status and prospects of genome-wide association studies in plants. In: The plant genome, p e20077

UniProt Consortium (2019) UniProt: a worldwide hub of protein knowledge. Nucleic Acids Res 47(D1):D506–D515

Wilkinson MD, Dumontier M, Aalbersberg IJ, Appleton G, Axton M, Baak A, Blomberg N, Boiten JW, da Silva Santos LB, Bourne PE, Bouwman J (2016) The FAIR guiding principles for scientific data management and stewardship. Sci Data 3(1):1–9