





RDFtex: Knowledge Exchange Between LaTeX-Based Research Publications and Scientific Knowledge Graphs

Leon Martin^(✉)  and Andreas Henrich 

Media Informatics, University of Bamberg, Bamberg, Germany
{leon.martin, andreas.henrich}@uni-bamberg.de

Abstract. Scientific Knowledge Graphs (SciKGs) aim to integrate scientific knowledge in a machine-readable manner. For populating SciKGs, research publications pose a central source of knowledge. The goal is to represent both contextual information, i.e., metadata, and contentual information, i.e., original contributions like definitions and experimental results, of research publications in SciKGs. However, typical forms of research publications like traditional papers do not provide means of integrating contributions into SciKGs. Furthermore, they do not support making direct use of the rich information SciKGs provide. To tackle this, the present paper proposes *RDFtex*, a framework enabling (1) the import of contributions represented in SciKGs to facilitate the preparation of \LaTeX -based research publications and (2) the export of original contributions to facilitate their integration into SciKGs. As a proof of concept, an *RDFtex* implementation is provided. We demonstrate the framework's functionality using the example of the present paper itself since it was prepared using this implementation.

Keywords: Research data management · Data and research infrastructure · \LaTeX research publications

1 Introduction

Between 2008 and 2018, i.e., in one mere decade, the number of research publications published each year grew from about 1.8 million to about 2.6 million¹. The resulting flood of publications and scientific data poses different challenges for researchers. For instance, keeping track of relevant related work and state-of-the-art experimental results has become increasingly difficult. Hence, the need for new means of organizing publications and scientific data has risen, eventually leading to the proposal of Scientific Knowledge Graphs (SciKGs) [1, 6, 9] that aim to integrate scientific information into a knowledge base. SciKGs have the potential to fundamentally transform the way researchers acquire information for

¹ <https://nces.nsf.gov/pubs/nsb20206> (accessed 2022/07/19).

preparing their publications and share their contributions. However, the envisaged shift towards what is herein called *knowledge graph augmented research* raises questions about the form of research publications, i.e., their suitability for this new research paradigm. In a previous paper [10], we discussed three publication forms (including the predominant *document-based publications*) regarding their utility with respect to knowledge graph augmented research. The investigation has shown that all of them are flawed in this context. Therefore, based on the identified advantages and disadvantages, a set of requirements (cf. Sect. 2) for a publication form that is specifically designed for knowledge graph augmented research was compiled.

Building upon this, the present paper proposes RDFtex as an attempt to bridge the gap between L^AT_EX research publications and SciKGs. RDFtex is a framework enabling a bidirectional knowledge exchange between L^AT_EX research publications and a scientific knowledge graph. To implement the said knowledge exchange, RDFtex comprises two main functionalities:

1. The import functionality allows the import of research contributions from a SciKG in L^AT_EX documents via custom import commands.
2. The export functionality allows the export of original research contributions from L^AT_EX documents to a SciKG via custom export commands.

To showcase RDFtex’s functionalities, we provide an implementation of RDFtex² serving as a proof of concept. This paper gives an impression of RDFtex since it has been produced using the implementation and a makeshift SciKG called *MinSKG* (cf. Sect. 3).

The remainder of this paper is structured as follows: Sect. 2 describes relevant foundations including related work, followed by a thorough introduction of RDFtex’s functionalities in Sect. 3. The discussion in Sect. 4 investigates advantages and limitations of the framework. Finally, Sect. 5 draws a conclusion.

2 Foundations

The Resource Description Framework (RDF) [2] provides a generic approach for representing knowledge in the form of triples, where Internationalised Resource Identifiers (IRIs), a generalization of Uniform Resource Identifiers (URIs), are used as identifiers. Each triple comprises a subject (an IRI or a blank node), a predicate (an IRI), and an object (an IRI, a literal, or a blank node). The predicate represents a property, i.e., a binary relation between subject and object. Collections of such RDF triples constitute so called knowledge graphs.

Typically, triplestores [12] are employed to handle and interact with RDF-based knowledge graphs. To retrieve information from knowledge graphs, they usually provide an interface that accepts queries written in the SPARQL Protocol And RDF Query Language (SPARQL) [13]. To exchange and archive knowledge graphs, serialization formats such as Turtle, RDF/XML, and N3 exist [2].

² The implementation is written in Python 3.9. The code and other used resources are available at <https://github.com/uniba-mi/rdftex> (accessed 2022/07/19).

In contrast to more general knowledge graphs that gather information across domains like Wikidata³ and DBpedia⁴, SciKGs, i.e., knowledge graphs that are tailored to a (certain) scientific domain, such as the Open Research Knowledge Graph (ORKG) [6] and KnowLife [3] represent rich knowledge bases specifically for scientific information. To acquire this knowledge, research publications pose a central resource. Hence, SciKGs typically do not only aim to capture *contextual* information, i.e., metadata, of research publications but also *contentual* information, i.e., the original contributions the publications provide⁵. Figure 1 gives an example of the difference between the two types of information⁶.

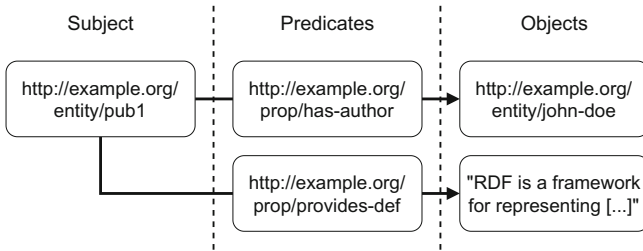


Fig. 1. A simple exemplary knowledge graph consisting of two RDF triples. The upper triple provides contextual information, the lower triple contentual information of the publication *pub1*. All non-literal triple members are identified using IRIs. (Figure and caption adopted from [10].)

To facilitate building high-quality and complete SciKGs, the process of integrating scientific information from research publications should be as simple as possible and at best be performed automatically with only little human intervention for quality assurance. From the authors' perspective, SciKGs provide the opportunity to facilitate the preparation of new publications if the rich information they provide is leveraged. However, an investigation of three different available publication forms in [10] shows that all of them have disadvantages regarding knowledge graph augmented research. *Document-based publications*, commonly called papers, pose the problem that authors have to introduce the same concepts across multiple publications to the readers even if this information is readily available in SciKGs. This results in redundant passages—typically found in a paper's introductory sections—that provide no added value while consuming valuable preparation time and effort. Furthermore, keywords and other classification systems are not sufficient to allow a direct integration of a publication into a SciKG since they are usually not formatted in an RDF-compatible

³ <https://www.wikidata.org> (accessed 2022/07/19).

⁴ <https://www.dbpedia.org> (accessed 2022/07/19).

⁵ We use this terminology to make the distinction between metadata and the actual content more clear since both comprise semantic information but at different levels of abstraction.

⁶ The use of available vocabularies for the predicates was omitted here.

way. *RDF-transformed publications* aim to close the gap between document-based publications and RDF by applying techniques like *SciIE* [9] to generate RDF graphs based on document-based publications. Nevertheless, the problems regarding the redundant passages continue to exist since authors still have to prepare regular papers in the first place.

Finally, *nanopublications* [8] (cf. Definition 1) focus on the integration with RDF by directly encoding the contributions as RDF triples. This, however, implies additional training effort for authors while readers will have to get accustomed to this publication form. As long as nanopublications are not well-established in the scientific community, the need for an interim solution for integrating the predominant form of publications, i.e., document-based publications, with SciKGs arises, to which RDFtex is our answer.

Definition 1. *Nanopublications [provide] a granular and principled way of publishing scientific (and other types of) data in a provenance-centric manner. Such a nanopublication consists of an atomic snippet of a formal statement [...] that comes with information about where this knowledge came from [...] and with metadata about the nanopublication as a whole [...]. All these three parts are represented as Linked Data (in RDF) [...]. [8]*

Based on the discussed advantages and disadvantages, five requirements for a publication form tailored for the use in SciKGs were proposed in [10]. RDFtex addresses four of them⁷. The first requirement states that authors shall still be able to prepare their main contributions in natural language. RDFtex can meet this requirement only partially since the underlying L^AT_EX documents require non-natural language statements by design. The import and export commands introduced by RDFtex also do not use natural language. Hence, RDFtex permits the usage of natural language to the same degree regular L^AT_EX documents do. The second requirement prescribes a means of importing knowledge from SciKGs with the goal of avoiding the need to prepare redundant passages across publications. RDFtex allows importing contributions from a SciKG via import commands. In RDFtex’s preprocessing step, actual content based on the contributions’ information is inserted. Vice versa, the third requirement demands means of exporting knowledge from publications to a SciKG to facilitate the integration of research publications. For this purpose, RDFtex allows marking up original contributions in publications via export commands that are then parsed in the preprocessing step to generate RDF documents based on the publications and their contributions. The final requirement states that readers shall receive a version of the publication that is enriched using information imported from the SciKG. In the preprocessing step, the import commands are replaced by templates that are populated with information retrieved from the SciKG. Compiling the preprocessed L^AT_EX documents to PDF afterwards results in one coherent document that is enriched using the information from the SciKG.

⁷ The remaining requirement addressing the need for additional tooling to obtain relevant URIs from a SciKG lies beyond the scope of the present paper (cf. Sect. 4).

2.1 Related Work

In the past, different approaches for linking parts of resources or documents in other documents have been proposed. One early example, Project Xanadu [11], introduced so called *xanadocs*. Essentially, a xanadoc is a so called Edit Decision List (EDL) file that consists of a set of links—typically URIs—referring to other documents with numerical start and length information for identifying spans of text in the linked resources. Special xanadoc viewers generate one coherent document by retrieving and composing the texts spans from the linked documents at the specified start positions with the specified lengths. Similar to xanadocs and with compliance to RDF, RDFtex makes use of URIs to reference content/entities, which correspond to scientific contributions in our case.

Another approach that is related to RDFtex is the so called Resource Description Framework in Attributes (RDFa) [5], which is a technique that provides a set of special attributes for XML-based languages to mark up elements with machine readable RDF information. Listing 2.1 demonstrates its core functionality using the example of an HTML snippet.

Listing 2.1. RDFa’s core functionality using the example of an HTML snippet. The code was adopted from [5]. Note the tree structure of XML-based languages, which is exploited by RDFa.

```
<body prefix="dc:http://purl.org/dc/terms/">
  <div resource="/alice/posts/trouble_with_bob">
    <h2 property="dc:title">The trouble with Bob</h2>
    ...
    <h3 property="dc:creator" resource="#me">Alice</h3>
    ...
  </div>
</body>
```

By exploiting HTML’s tree structure, the RDFa information in the example can be transformed into a knowledge graph featuring one blog post entity that acts as the subject of two triples, where the **property** attributes of the nested elements specify the predicates and the element contents correspond to the objects. RDFtex’s export functionality follows RDFa’s idea of marking up content with RDF information to provide machine readability. In RDFtex’s preprocessing step, an RDF document is generated based on the marked up information that can then be used as a basis for integrating the publication into a SciKG.

Furthermore, there are projects that investigate means of annotating the contents of \LaTeX documents with additional semantic information for further processing. For instance, *sTeX* [7] allows annotating mathematical knowledge in \LaTeX documents. The annotations can then be transformed into mathematical knowledge management representation formats. As another example, *SALT* [4] is a comprehensive framework that aims to capture the semantic content of \LaTeX documents using a federation of three interlinked ontologies. Compared to these projects, RDFtex provides the novelty that knowledge cannot only be marked

up and exported but also imported. Another key difference is that the described projects do not aim to interoperate with a SciKG.

3 Concept

Introducing the framework, Fig. 2 shows that RDFtex operates on files with the `.rdf.tex` file extension and adds a composite preprocessing step to the basic workflow for producing a PDF document from `.tex` files. The preprocessing step includes the export and the import of contributions, which will be explained below. As indicated by the dashed arrow, the RDF document resulting from the export can be integrated into the SciKG later on. Syntactically, `.rdf.tex` files are regular `.tex` files that moreover permit the usage of custom RDFtex commands. The custom file extension facilitates identifying the source files with the custom commands for preprocessing. Apart from the occasional usage of the custom RDFtex commands, authors can prepare their publication using `.rdf.tex` files as they are used to; the usage of \LaTeX templates is also supported. In the preprocessing step, a regular `.tex` document is generated for each `.rdf.tex` document by replacing the custom commands with actual content or removing them. The resulting `.tex` files can then be compiled as usual. Note that the framework can also be integrated into a fully automated workflow⁸.

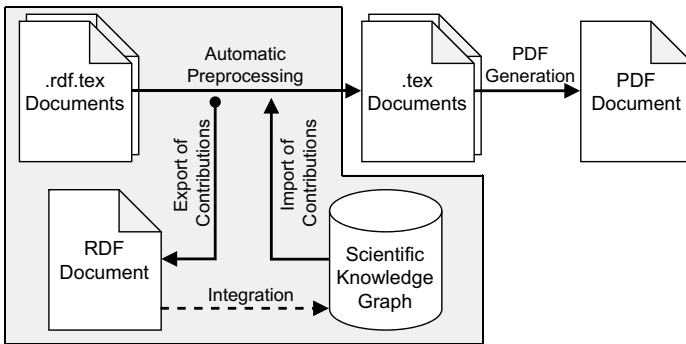


Fig. 2. The process for producing a PDF file using RDFtex. The gray box indicates additional steps and resources introduced or used by RDFtex. (Color figure online)

The present paper has been prepared using RDFtex. Thus, examples originating from this paper will be used in the following to explain the import and export functionality. As said before, preparing a \LaTeX publication using RDFtex

⁸ Our proof-of-concept preprocessor implementation can be configured to be executed whenever a `.rdf.tex` file changes. Similarly, tools like *latexmk* (<https://ctan.org/pkg/latexmk>, accessed 2022/07/19) provide the option to compile \LaTeX projects whenever a `.tex` file changes. Combining the two, the PDF is compiled fully automatically whenever changes are made to any `.rdf.tex` file.

requires a suitable SciKG with which the knowledge exchange can be practiced. However, available SciKGs do not yet contain the papers and contributions referenced in this paper. Hence, we created the so called *MinSKG*. The MinSKG is a minimal scientific knowledge graph populated with all publications that are used as references for the present paper. The contextual information of the publications, i.e., the metadata, was added automatically by parsing their entries from the `bibtex` file. The contentual information, i.e., their contributions, was added manually. Note that only the contributions that are actually imported in the present paper have been added. The script for building the MinSKG and the MinSKG itself is also provided in RDFtex’s repository (see Footnote 2).

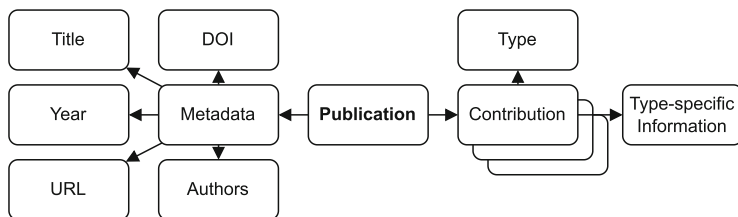


Fig. 3. The information available for each publication represented in the MinSKG. The type-specific information of a contribution depends on its type. Note that edge labels, i.e., predicates, and URIs are omitted for readability.

Figure 3 shows the structure of the information for publications represented in the MinSKG. The structure is loosely based on the ORKG to ensure that RDFtex operates on a SciKG with a realistic structure. However, the MinSKG just serves as a small makeshift SciKG (cf. Sect. 4) that has to be replaced by an actual SciKG in a comprehensive implementation of the framework.

3.1 Import of Contributions

RDFtex’s import command enables authors to import contributions from a SciKG. During the preprocessing step, the import commands within the `.rdf.tex` files are parsed and snippets of actual content are produced based on the contribution information retrieved from the MinSKG. The snippets are added to the `.tex` files in place of the import commands. To identify the contributions to be imported, URIs are used. Hence, the import commands use this syntax:

```
\rdfimport{<label>}{<citation-key>}{<contrib-url>}
```

As depicted, the `\rdfimport` command expects three parameters. The first parameter `<label>` corresponds to the label the authors want to assign to the generated content for future reference in their paper. Second, `<citation-key>` is a placeholder for the `bibtex` citation key of the publication from which the imported contribution originates. The final parameter `<contrib-url>` denotes

the URI of the contribution to be imported. The paragraphs to follow will build on the example of Fig. 1, which we imported in our `.rdf.tex` file as follows⁹:

```
\rdfimport{fig:contentual-contextual} \
  {Martin21} \
  {https://example.org/scikg/publications/Martin21/contrib2}
```

The example shows that the import commands are agnostic regarding the type of the imported contribution preventing the need for learning different syntaxes.

To obtain the contribution’s information, a SPARQL query of the form

```
SELECT ?p ?o WHERE {<contrib-url> ?p ?o .}
```

is performed on the MinSKG. With respect to the MinSKG’s structure (cf. Fig. 3), this query yields all predicates and objects of triples where `<contrib-url>` is the subject. This information as well as the remaining two parameters are used in the next step to generate the content snippets. The number of triples available depends on the type of the contribution. In the case of the imported figure from the example above, the MinSKG provides information about the figure’s MIME type, the URL at which the actual figure can be found, a description, and the contribution type itself, which in this case would be `Figure`.

The type of the contribution to be imported also determines the template that is applied to generate the content. The templates are written in \LaTeX to allow for a direct injection in the `.tex` files and they incorporate the respective information that the different contribution types in the MinSKG provide. The template that is applied for importing figures looks as follows:

```
\begin{figure}[htb!]
  \centering
  \includegraphics[max width=0.7\columnwidth]{<figure-url>}
  \caption{<figure-desc> \
    (Figure and caption adopted from~\cite{<citation-key>}.)}
  \label{spslabelsps}
\end{figure}
```

As shown, the example uses the placeholders `<figure-url>` and `<figure-desc>`, which correspond to the location of the figure and its description. All placeholders within the import templates are filled with information retrieved from the MinSKG. Thus, it is important to ensure that no contributions are added to the MinSKG without the necessary predicates and objects. Otherwise, they cannot be imported successfully. In such cases, the preprocessor shows meaningful error messages. For some of the supported contribution types, templates with custom \LaTeX environments are used whose definitions are automatically inserted into the `.tex` files if necessary.

⁹ In the following, single “\”-symbols indicate lines that are too long. In reality, there are no line breaks.

3.2 Export and Integration of Contributions

For the export of original contributions, RDFtex enables authors to mark up relevant text passages in their `.rdf.tex` files. In the following, the export of our proof-of-concept preprocessor implementation serves as the running example for demonstrating RDFtex’s export functionality. The export functionality leverages two custom L^AT_EX commands. The first command is used to declare the export of a certain contribution. The rationale for introducing a declaration statement is that the exact string specifying the contribution type in the MinSKG, e.g., `ExpResult` (indicating an experimental result), might not be explicitly mentioned in the text. At the same time, the type information is mandatory since it determines the templates to be applied in the import process as explained above. The same holds for other predicates and objects that denote the MIME type of a figure, for example. To tackle this, the export declaration command

```
\rdfexport{<contrib-name>}{<type>}{<predicate-uri=object>,...}
```

is introduced. `<contrib-name>` corresponds to the name of the contribution. The specified name serves as a local identifier for the contribution. When the actual export takes place, a generated preliminary URI will be used as an identifier instead to comply with the RDF standard. The second parameter `<type>` denotes the contribution’s type and the third parameter accommodates optional `<predicate-uri=object>` statements, which are placeholders for predicate object combinations that are required for a valid export but cannot be included appropriately in the text. For example, to add the MIME type information for a figure to be exported, the third parameter could be set to:

```
https://example.org/scikg/terms/figure_mime=image/png
```

Applied to the running example, the following snippet declares the export of the RDFtex software:

```
\rdfexport{RDFtex Implementation}{Software}{}
```

Note that the third parameter was omitted here since all information required for a valid software export are stated explicitly in the text such that they can be marked up directly using the second export command.

The second command is similar to the property attributes of RDFa (cf. Listing 2.1) for marking up predicates. While RDFa can exploit the nested element structures of XML-based languages to associate subjects with predicates and objects, the triple components are potentially spread across sections, in our case. Therefore, each RDFtex property command has to explicitly reference the contribution they belong to via the `<contrib-name>` used in the export declaration command, resulting in the following syntax:

```
\rdfproperty{<contrib-name>}{<predicate-uri>}{<object>}
```

The parameter `<predicate-uri>` denotes a predicate used for describing contributions in the MinSKG and `<object>` encloses the part of the content that is to be used as the triple’s object. For instance, to add a description of the RDFtex software to the MinSKG, we modified the line from Sect. 1 to the following:

```
\rdfproperty{RDFtex Implementation} \
  {https://example.org/scikg/terms/software_description} \
  {RDFtex is a framework enabling a bidirectional...}
```

When lines with `rdffexport` or `rdfproperty` commands are encountered during preprocessing, their arguments are collected for the respective contributions identified via `<contrib-name>`. For the `.tex` file generation, lines with `rdffexport` commands are omitted and `rdfproperty` commands are replaced with only their `<object>` argument, thus removing the custom commands.

Theoretically, the export of a contribution could be performed using only the `rdffexport` command due to the power of its third parameter. However, the usage of the `rdfproperty` command offers the possibility to use parts of the text in the publication directly as objects for the predicates. Among others, this serves the objective of avoiding redundant text passages.

When all files have been processed, the collected information is checked for completeness to ensure that the exported contributions can be imported again. At this point, the validated exports could be directly integrated into the MinSKG. However, some information about the publication might not be obtainable yet, e.g., the DOI has to be assigned by publishers first, or has to adhere to certain criteria such as IRI naming schemes, which can only be verified by the maintainers of a SciKG. Hence, to avoid adding information to the SciKG that is incomplete or does not meet the quality standards, a preliminary RDF document that complies with the MinSKG's structure (cf. Fig. 3) is generated based on the exported contributions instead. For this purpose, a placeholder IRI is assigned to the paper's entity, which should later be replaced with a persistent identifier. We suggest passing the RDF document alongside the camera-ready publication to the publishers. The publishers should then add the missing information before submitting the document to the SciKG maintainers who finally adjust the IRI and perform the integration.

3.3 Prefix Syntax

To mitigate the readability problems arising from lengthy URIs, many RDF-related technologies including RDFa allow defining prefixes that can then be used to abbreviate full URIs. Following this approach, RDFtex provides another command for registering prefixes:

```
\rdfprefix{<prefix>}{<url>}
```

The parameter `<prefix>` states the prefix to be used in place of `<url>` in the `.rdf.tex` files. For this, the `rdffprefix` commands have to be placed above the lines that use the prefixes in each `.rdf.tex` file. Using the prefix command, the `rdfproperty` command for adding the software description from above can be rewritten as follows:

```
\rdfprefix{mskg}{https://example.org/scikg/terms/}
\rdfproperty{RDFtex Implementation} \
  {mskg:software_description} \
  {RDFtex is a framework enabling a bidirectional...}
```

Table 1. An overview of the custom RDFtex \LaTeX commands.

Command	Purpose	Parameters
<code>rdfimport</code>	Denotes the import of a contribution from the SciKG	A label that will be assigned to the generated content snippet, the citation key, and the URI pointing to the contribution
<code>rdfexport</code>	Registers an original contribution for export	A contribution name to reference the export locally, the type of the original contribution, and an optional set of other predicates and objects that are assigned to the contribution entity if applicable
<code>rdfproperty</code>	Marks up an attribute of an original contribution to be exported	The contribution name of the export, the predicate, and the text representing the object
<code>rdfprefix</code>	Sets a prefix for abbreviating the terms of a vocabulary	The prefix and the written out form of the vocabulary URI

Summarizing Sect. 3, Table 1 provides an overview of the custom RDFtex commands, their purpose, and their parameters.

4 Discussion

RDFtex and the underlying concept of exchanging knowledge between research publications and SciKGs represent one option to facilitate the move towards knowledge graph augmented research. Our goal was to provide an extension to a well-established publication form that feels natural while providing compatibility to the SciKG world. Using RDFtex, authors gain the ability to make direct use of contributions that are already represented in SciKGs. They can also facilitate the integration of their publications using RDFtex’s export functionality, which is vital for being visible in the new research paradigm. At the same time, RDFtex makes only minor changes to the familiar way of preparing a \LaTeX contribution.

From the authors’ perspective, using the custom RDFtex commands requires some training and somewhat impairs the readability of the \LaTeX code. However, the concise syntax and the introduced prefix mechanism alleviate this problem to a certain degree. Speaking of URIs, authors also have to obtain the relevant URIs from the SciKG to use the import functionality—a topic that is not covered in the present paper because SciKG maintainers and search engines are better suited to provide tools for this task¹⁰. Also, tools for collaboratively preparing

¹⁰ For example, Semantic Scholar (<https://www.semanticscholar.org>; accessed 2022/07/19) and similar search engines could provide the URIs to a publication’s contributions alongside the links to the publication itself, in the future.

publications like Overleaf¹¹ do not support RDFtex yet. Implementation-wise, including RDFtex in such tools is feasible, though. Furthermore, one can already set up an automatic build pipeline as described in Footnote 8 and use a version control system to enable collaboration.

Runtime Evaluation. Our proof-of-concept implementation demonstrates that it is feasible to scan the `.rdf.tex` files for custom commands line by line and only once, while the lines of the `.tex` files are written simultaneously, thus requiring linear time. Since the present paper has been prepared using our proof-of-concept implementation, it was possible to monitor its runtime. Overall, this paper comprises two imports, namely Fig. 1 and Definition 1. Also, RDFtex as a software artifact, Fig. 2, Fig. 3, the MinSKG as a dataset, and the experimental results that are stated next have been exported as original contributions. Given these imports and exports, the preprocessing took only 1.15 s on a standard desktop PC across 100 runs on average, even though the MinSKG was deployed on a server in order to take into account the network overhead for the tests. At the same time, the MinSKG is still a small makeshift which results in faster than normal SPARQL answering, thus limiting the results. That being said, retrieving all predicates and objects of the entity *RDFa* from Wikidata via its SPARQL endpoint¹², i.e., a query that is comparable to the one used by RDFtex (cf. Sect. 3.1), takes just about 0.25 s. Also, caching could be used to prevent retrieving the same contribution data multiple times.

Initial User Feedback. To get initial user feedback, we asked three computer scientists and one astrophysicist who are familiar with \LaTeX to perform imports and exports by means of our implementation on a blank `.rdf.tex` file using the think-aloud method. The test persons had access to the documentation of the provided repository (see Footnote 2). As expected, there is a strong need for tooling to obtain the necessary URIs because the test persons had problems finding the correct URIs in the MinSKG. RDFtex’s concept, however, was received well, especially the import functionality due to its simplicity. The export functionality proved to be more complex due to the larger number of involved commands. Understandably, two test persons mentioned that they would use the framework only when the benefits of knowledge graph augmented research become noticeable.

5 Conclusion

The present paper proposed RDFtex, a framework for producing \LaTeX research publications using imports and exports to and from a SciKG. For this purpose, RDFtex introduces four custom \LaTeX commands. The envisaged shift towards knowledge graph augmented research, which motivated this work, poses a significant change for the research culture. It depends on the scientific community

¹¹ <https://www.overleaf.com> (accessed 2022/07/19).

¹² <https://query.wikidata.org> (accessed 2022/07/19).

if this change will actually take place, though. If so, we think that RDFtex is a promising approach to facilitate the move towards the new research paradigm since it only introduces minor differences compared to the preparation of traditional L^AT_EX publications while narrowing the gap between papers and RDF.

References

1. Auer, S., Kovtun, V., Prinz, M., Kasprzik, A., Stocker, M., Vidal, M.: Towards a knowledge graph for science. In: Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics, WIMS 2018, Novi Sad, Serbia, 25–27 June 2018, pp. 1:1–1:6. ACM (2018). <https://doi.org/10.1145/3227609.3227689>
2. Cyganiak, R., Hyland-Wood, D., Lanthaler, M.: RDF 1.1 concepts and abstract syntax (2014). <https://www.w3.org/TR/rdf11-concepts>. Accessed 19 July 2022
3. Ernst, P., Siu, A., Weikum, G.: KnowLife: a versatile approach for constructing a large knowledge graph for biomedical sciences. BMC Bioinform. **16**, 157:1–157:13 (2015). <https://doi.org/10.1186/s12859-015-0549-5>
4. Groza, T., Handschuh, S., Möller, K., Decker, S.: SALT - semantically annotated latex for scientific publications. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519, pp. 518–532. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72667-8_37
5. Herman, I., Adida, B., Sporny, M., Birbeck, M.: RDFa 1.1 primer (2015). <https://www.w3.org/TR/rdfa-primer>. Accessed 19 July 2022
6. Jaradeh, M.Y., et al.: Open research knowledge graph: next generation infrastructure for semantic scholarly knowledge. In: Proceedings of the 10th International Conference on Knowledge Capture, K-CAP 2019, Marina Del Rey, CA, USA, 19–21 November 2019, pp. 243–246. ACM (2019). <https://doi.org/10.1145/3360901.3364435>
7. Kohlhase, A., Kohlhase, M., Lange, C.: S^te^x+: a system for flexible formalization of linked data. In: I-SEMANTICS. ACM (2010)
8. Kuhn, T., et al.: Nanopublications: a growing resource of provenance-centric scientific linked data. CoRR (2018). <https://arxiv.org/abs/1809.06532>
9. Luan, Y., He, L., Ostendorf, M., Hajishirzi, H.: Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018, pp. 3219–3232. Association for Computational Linguistics (2018). <https://doi.org/10.18653/v1/d18-1360>
10. Martin, L., Jegan, R., Henrich, A.: On the form of research publications for use in scientific knowledge graphs. In: Wissensorganisation 2021: 16. Tagung der Deutschen Sektion der Internationalen Gesellschaft für Wissensorganisation (ISKO) (WissOrg 2021) (2021, accepted)
11. Project Xanadu: Project Xanadu (2021). <https://www.xanadu.net>. Accessed 19 July 2022
12. Saleem, M., Szárnyas, G., Conrads, F., Bukhari, S.A.C., Mehmood, Q., Ngomo, A.N.: How representative is a SPARQL benchmark? An analysis of RDF triplestore benchmarks. In: Liu, L., et al. (eds.) The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, 13–17 May 2019, pp. 1623–1633. ACM (2019). <https://doi.org/10.1145/3308558.3313556>
13. W3C SPARQL Working Group: SPARQL 1.1 overview (2013). <https://www.w3.org/TR/sparql11-overview>. Accessed 19 July 2022