Transactions on Graph Data and Knowledge

Volume 3 | Issue 3 | December, 2025

ISSN 2942-7517

ACM Classification 2012
Theory of computation → Constraint and logic programming; Computing methodologies → Logic programming and answer set programming; Information systems → Electronic data interchange; Information systems → Resource Description Framework (RDF); Information systems → Web Ontology Language (OWL); Information systems → Ontologies; Computing methodologies → Ontology engineering; Theory of computation → Semantics and reasoning; Computing methodologies → Information extraction; Computing methodologies → Language resources; Computing methodologies → Semantic networks; Information systems → Graph-based database models

Published online and open access by
Schloss Dagstuhl – Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing, Saarbrücken/Wadern, Germany.
Online available at
`https://www.dagstuhl.de/dagpub/2942-7517`.

Publication date
December, 2025

Digital Object Identifier
10.4230/TGDK.3.3.0

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at https://dnb.d-nb.de.

Aims and Scope
Transactions on Graph Data and Knowledge (TGDK) is an Open Access journal that publishes original research articles and survey articles on graph-based abstractions for data and knowledge, and the techniques that such abstractions enable with respect to integration, querying, reasoning and learning. The scope of the journal thus intersects with areas such as Graph Algorithms, Graph Databases, Graph Representation Learning, Knowledge Graphs, Knowledge Representation, Linked Data and the Semantic Web. Also in-scope for the journal is research investigating graph-based abstractions of data and knowledge in the context of Data Integration, Data Science, Information Extraction, Information Retrieval, Machine Learning, Natural Language Processing, and the Web.

The journal is Open Access without fees for readers or for authors (also known as Diamond Open Access).

Contents

List of Authors

Maribel Acosta (2)
Technical University of Munich, Germany

Shqiponja Ahmetaj (1)
TU Wien, Austria; Vienna University of Economics and Business, Austria

Abraham Bernstein (4)
University of Zurich, Switzerland

Bruce F. Chorpita (3)
University of California, Los Angeles, CA, USA

Robert David (1)
Vienna University of Economics and Business, Austria; Semantic Web Company GmbH, Vienna, Austria

Irlan Grangel-González (5)
Corporate Research, Robert Bosch GmbH, Renningen, Germany

Evgeny Kharlamov (5)
Corporate Research, Robert Bosch GmbH, Renningen, Germany

Deborah L. McGuinness (3)
Rensselaer Polytechnic Institute, Troy, NY, USA

Johannes Mäkelburg (2)
Technical University of Munich, Germany

Adrian Paschke (5)
Data Analytic Center (DANA), Fraunhofer Institute FOKUS, Berlin, Germany; AG Corporate Semantic Web, Freie Universität Berlin, Germany

Paulo Pinheiro (3)
Instituto Piaget, Lisbon, Portugal

Axel Polleres (1)
Vienna University of Economics and Business, Austria; Complexity Science Hub, Vienna, Austria

Kelsey Rook (3)
Rensselaer Polytechnic Institute, Troy, NY, USA

Florian Ruosch (4)
University of Zurich, Switzerland

Henrique Santos (3)
Rensselaer Polytechnic Institute, Troy, NY, USA

Cristina Sarasua (4)
University of Zurich, Switzerland

Wilma Johanna Schmidt (5)
Corporate Research, Robert Bosch GmbH, Renningen, Germany; AG Corporate Semantic Web, Freie Universität Berlin, Germany

Manuel S. Sprung (3)
University of California, Los Angeles, CA, USA

Mantas Šimkus (1)
TU Wien, Austria; Umeå University, Sweden

A Logic Programming Approach to Repairing SHACL Constraint Violations

Shqiponja Ahmetaj
TU Wien, Austria
Vienna University of Economics and Business, Austria

Robert David[1]
Vienna University of Economics and Business, Austria
Semantic Web Company GmbH, Vienna, Austria

Axel Polleres
Vienna University of Economics and Business, Austria
Complexity Science Hub, Vienna, Austria

Mantas Šimkus
TU Wien, Austria
Umeå University, Sweden

Abstract

The Shapes Constraint Language (SHACL) is a recent standard, a W3C recommendation, for *validating* RDF graphs against *shape* constraints to be checked on *target nodes* of a data graph. The standard also describes the notion of *validation reports*, which detail the results of the validation process. In case of violation of constraints, the validation report should explain the reasons for non-validation, offering guidance on how to identify or fix violations in the data graph. Since the specification left it open to SHACL processors to define such explanations, a recent work proposed the use of explanations in the style of database *repairs*, where a repair is a set of additions to or deletions from the data graph so that the resulting graph validates against the constraints. In this paper, we study such repairs for non-recursive SHACL, the largest fragment of SHACL that is fully defined in the specification. We propose an algorithm to compute repairs by encoding the explanation problem – using Answer Set Programming (ASP) – into a logic program, where the answer sets contain (minimal) repairs. We then study a scenario where it is not possible to simultaneously repair all the targets, which may be the case due to overall unsatisfiability or conflicting constraints. We introduce a relaxed notion of validation, which allows to validate a (maximal) subset of the targets and adapt the ASP translation to take into account this relaxation. Finally, we add support for repairing constraints which use property paths and equality of paths. Our implementation in clingo is – to the best of our knowledge – the first implementation of a repair program for SHACL.

2012 ACM Subject Classification Theory of computation → Constraint and logic programming; Computing methodologies → Logic programming and answer set programming

Keywords and phrases SHACL, Shapes Constraint Language, Database Repairs, Knowledge Graphs, Semantic Web, Answer Set Programming

Digital Object Identifier 10.4230/TGDK.3.3.1

Related Version *Previous Version*: `https://iswc2022.semanticweb.org/wp-content/uploads/2022/11/978-3-031-19433-7_22.pdf`

Supplementary Material *Software (Source Code)*: `https://github.com/robert-david/shacl-repairs` [23] archived at `swh:1:dir:9de47c8b97cba079add751f9e2a64421238a67f3`

Funding This work was supported by the Austrian Science Fund (FWF) Cluster of Excellence "BILAI" [10.55776/COE12]. Ahmetaj was also supported by the Austrian Science Fund (FWF) projects netidee SCIENCE [T1349-N] and the Vienna Science and Technology Fund (WWTF) [10.47379/ICT2201].

Received 2024-07-19 **Accepted** 2025-09-12 **Published** 2025-12-10

[1] Corresponding author

1 Introduction

Semantic Web standards provide means to represent and link heterogeneous data sources in knowledge graphs [30], thereby potentially solving common data integration problems. Indeed, this approach became increasingly popular in enterprises as *enterprise knowledge graphs (EKG)*, which integrate data silos of an enterprise into one consolidated knowledge graph [27]. However, in practice, this flexible and expressive approach to data integration requires powerful tools for ensuring data quality, including ways to avoid creating invalid data and inconsistencies in the consolidated knowledge graphs. Although our work is motivated by the high quality requirements of EKGs in practice, it is by far not limited to this scenario and generally applicable to graph data. To this end, the W3C proposed the Shapes Constraint Language SHACL, in order to enable validation of RDF graphs against a set of shape constraints [32]. In this setting, the validation requirements are specified in a *shapes graph* (C, T) that consists of a collection C of constraints and a specification T of nodes, called *targets*, to which the constraints should be applied.

The result of validating an RDF graph G against a shapes graph (C, T) is presented as a *validation report*, which indicates whether the input graph has passed the validation test. In the event of constraint violations, the report should provide details explaining the issues and violations found during the validation process. However, the SHACL standard offers limited guidance regarding the specific content of validation reports, except for the requirement that these reports must include information about the violated nodes, constraints, and the affected part of the input graph. We follow the proposal of [3] to define *explanations* for SHACL non-validation in the style of database *repairs* [7], namely as a set of additions and deletions of facts to the input data graph that will cause a previously invalid setting to become valid. Consider for instance a requirement, described as a shape constraint StudentShape, stating that persons need to be enrolled in at least one course to qualify as students and the target requires to validate StudentShape for *Ben*. In the abstract syntax, we define C to contain the constraint $\mathsf{StudentShape} \leftarrow \exists \mathit{enrolledIn}.\mathit{Course}$. and T to be $\{\mathsf{StudentShape}(\mathit{Ben})\}$. In case the shapes graph (C, T) is applied to the data graph $G = \{\mathit{enrolledIn}(\mathit{Ben}, C_1)\}$, stating that *Ben* has the property *enrolledIn*, a violation will be reported. An intuitive explanation for the violation is that there is no *Course*-fact about node C_1, and a repair would add the fact $\mathit{Course}(C_1)$ to the data graph G. Indeed, in many common scenarios for knowledge graphs (like the automated integration of heterogeneous data sources) inconsistencies might appear frequently and can be mitigated using SHACL validation [28]. Consequently, there is a need to *automatically* identify repairs that can be applied to the data graph in order to achieve consistency.

The main contributions of our work are as follows:

- We propose to compute repairs of a data graph by encoding the problem into *Answer Set Programming (ASP)* [25], a formalism sufficiently expressive to capture the high complexity of the repair problem for SHACL [3]. In particular, we show how to transform a given data graph G and a SHACL shapes graph (C, T) into an ASP program P such that the answer sets (stable models) of P can be seen as a collection of plausible repairs of G w.r.t. the shapes graph (C, T). Since efficient ASP solvers exist (we decided for clingo [29]), this provides a promising way to generate data repairs in practice. The repair generation task is challenging, because a given data graph might be repaired in many different ways. In fact, since fresh nodes could be introduced to the data graph during the repair process, an infinite number of repairs is possible. This needs to be handled carefully, and several design choices have to be made.
- For presentation clarity, we initially present a *basic encoding* of the repair task into ASP, making the core idea of our approach easily accessible. In this encoding, the repair program tries to find a repair of the input shapes graph for a basic fragment of SHACL that excludes

qualified number restrictions and complex paths but allows existential quantification. It also supports target declarations for nodes, classes, and the domain and range of properties. This encoding employs a particular strategy for introducing new nodes in the data graph: when a value for a property needs to be added (e.g., for a violated *sh*:*minCount* constraint), a fresh node is always introduced. We argue that this is a reasonable approach in general and closely aligns with the standard notion of *Skolemization*. Furthermore, leveraging the optimization features of ASP, our repair program ensures that among all possible repairs it can produce, only those that are minimal in terms of cardinality are returned. This guarantees that constraint violations are resolved with the fewest possible changes to the data graph.

- There are scenarios where enforcing the repair program to always introduce fresh values for satisfying cardinality constraints may exclude certain expected minimal repairs or even prevent any repairs from being generated. In some cases, reusing existing nodes is not only desirable, but also necessary. However, to maintain data quality as much as possible, our approach prioritizes introducing fresh values whenever feasible, resorting to existing constants only when necessary. We then present an extended version of our basic encoding that supports reusing existing nodes while still favoring the introduction of fresh values whenever possible. Moreover, we extend the fragment from the basic encoding to incorporate other SHACL features such as *cardinality constraints*, *constants*, and *property paths*. In particular, we add support for sequential and inverse paths, as well as equality constraints over paths and properties.
- We observe that requiring a repair to resolve violations for *all* targets specified in the input may be too strong. If the data graph has one inherently unsatisfiable target (e.g., because of some conflicting or contradictory constraints), then the repair program will not have any models and it will provide no guidance on how to proceed with fixing the data graph. To address this issue, we introduce the notion of *maximal repairs*, which aims to repair as many targets as possible. We show how our encoding can be extended to generate repairs under this new notion by leveraging the optimization features of clingo and incorporating disjunctive rules that allow certain targets to be skipped when necessary.
- We have implemented and tested these encodings using the clingo ASP system, which showed that our approach is promising for managing and improving the quality of RDF graphs in practice. More precisely, we used the Java programming language to implement the translation of the SHACL repair programs, which can be executed using clingo. We tested the implementation using a set of unit tests and a subset of the data shapes test suite to test the functional correctness and performance tests for Wikidata as a real-world data set.

The repair program and the ASP implementation presented in this paper establish a foundation for repairing RDF graphs in the presence of SHACL constraints, laying the groundwork for practical tools that enhance data quality in real-world applications. A preliminary version of this work has been published in [4]. This journal version builds upon [4] with the following key contributions: (i) an expanded SHACL fragment that now supports (limited) path expressions and equality over paths, covering a substantial portion of SHACL Core constraint components; (ii) broader support for SHACL targets, extending beyond node targets to include class-based and property-based targets, subject to minor syntactic restrictions that ensure the fragment remains non-recursive and avoids cyclic dependencies between constraints and targets; (iii) proofs for the proposed rewritings; and (iv) a real-world evaluation using Wikidata.

1.1 Related Work

There is a large body of work in the area of databases on repairing integrity constraints violations; for an overview of this topic see [14]. The closest in spirit to our work is [35], which specifies database repairs in the presence of integrity constraints using disjunctive logic programs with the answer set semantics. These repairs minimally modify a database to achieve conformance with a set of integrity constraints. More specifically, given an inconsistent database instance D and a set of integrity constraints, [35] proposes a repair program, whose stable models are in a one-to-one correspondence with the repairs of D. We follow this general idea for the graph-based RDF data model scenario in the presence of SHACL constraints. The adaptation is challenging because SHACL includes features not present in traditional integrity constraints, such as unrestricted negation in the rule body, which carries significant semantic consequences.

In the context of Description Logic, there is extensive research on repairing the data (called ABox) so that certain unwanted consequences from a knowledge base are removed. Baader et al. study optimal repairs [10, 11, 12], which repair the ABox while preserving as much as possible from the consequences of an ontology (or TBox). The TBox is assumed to be static and cannot be changed in the repair process. They also look at optimal repairs in the context of contractions in belief change [8, 9, 13]. We follow a similar approach in the sense of assuming SHACL constraints to be carefully designed and without errors, and focusing only on repairing the data graph.

Also related to our work is [20], where Calvanese et al. focus on explanations for non-entailment of a given tuple from a knowledge base over Description Logic ontologies. Explanations in this setting are considered as minimal sets of facts such that, if added to the input data, the query would be entailed, while the knowledge base would remain consistent with the added information. Similar to our work, the explanation problem is formalized as an abductive task and considers preferences, where subset-minimal and cardinality-minimal explanations and their complexity are studied. Analogously, [21] addresses the problem of explaining why a query is not entailed under existential rules.

A large body of works on so-called *consistent query answering* has developed since the seminal paper [7]. The idea is to exploit the repairs of an inconsistent database to provide answers to a query. Various inconsistency-tolerant semantics based on what repairs to select for accepting query answers have been studied in various database and knowledge representation settings [7, 14, 33, 17, 35], including implemented systems [35, 16, 15]. A recent work [5] investigates consistent query answering under SHACL constraints, using a core fragment of SPARQL – the standardized language for querying RDF data – as the query language. Although our work does not yet address queries, the repair implementation we provide represents a crucial first step toward enabling consistent query answering in this context.

Further related but orthogonal work to ours involves axiom pinpointing, which focuses on identifying minimal sets of axioms sufficient to entail (or not entail) a specific expression. Such sets are called justifications and represent explanations of an entailment. An overview of axiom pinpointing is provided by Peñaloza [37] and Kalyanpur et al. [31] presents work on computing justifications in the context of Description Logic ontologies. There are works that study provenance for database query results, which originated in database theory and was extended to the Description Logic setting in the context of Ontology-based Data Access (OBDA) [36, 18]. Li et al. [34] propose to repair Description Logic ontologies using a combination of axiom weakening and completing. This approach aims to strike a balance between minimizing the negative impacts of removing unwanted axioms while preserving correct consequences. Work on static verification [19, 2] study the problem of verifying if a sequence of updates (additions and deletions) applied to an input data graph will preserve or violate a set of constraints expressed as Description Logic ontologies.

1.2 Organization of the Paper

The paper is structured as follows.

- Section 2 describes SHACL validation, normalization of SHACL constraints and answer set programming, which represent the foundations of our repair approach and the implementation.

- Section 3 introduces our notion of repairs for SHACL non-validation and the design choices we consider for our repair program.
- Section 4 provides an encoding of the repair program into an ASP program. We provide rules for generating repairs and optimizing them to be of a small size. In additions, in case not all targets can be repairs, we suggest a more relaxed version that repairs maximal subsets of targets.
- Section 5 extends this encoding to repair constraints with arbitrary cardinality and to allow the reuse of existing constants in this case.
- Section 6 further extends this cardinality constraint encoding by supporting the repair of SHACL property paths and equality of paths.
- Section 7 describes the ASP implementation and includes the translation of SHACL shapes from RDF Turtle syntax into our abstract syntax.
- Section 8 tests this implementation. We present our test scenarios, which are unit tests, test cases based on the official SHACL data shapes test suite and performance tests based on a real-world scenario over Wikidata, and report the results.
- Section 9 concludes the paper by summarizing the contributions and proposing directions for future work.

2 SHACL Validation and Answer Set Programming

In this section, we describe SHACL and the notion of *validation* against RDF graphs. The Shapes Constraint Language SHACL [32] is the W3C recommendation for validating graphs against sets of constraints. A SHACL processor can validate a *data graph* against constraints in a *shapes graph* and in the case of violations provide details to the user as part of a validation report. For an introduction to data validation, SHACL, and its close relative ShEx, we refer to [28, 1]. To ease presentation, SHACL constraints will be represented using an abstract syntax and normal form, which will also serve as a basis for implementation. We also describe answer set programming (ASP), which we use to implement the repair program and the notation for ASP that we use in the paper.

2.1 SHACL Validation

We use the abstract syntax from [3] for RDF and SHACL. Note that in this work we focus on the "Core Constraint Components" of SHACL with restricted path expressions.

Data graphs

We first define *data graphs*[2], which are RDF graphs to be validated against shape constraints. Assume countably infinite, mutually disjoint sets N_N, N_C, and N_P of *nodes* (or *constants*), *class names*, and *property names*, respectively. A *data graph* G is a finite set of *(ground) RDF atoms* of the form $B(c)$ and $p(c, d)$, where B is a class name, p is a property name, and c, d are nodes.

Syntax of SHACL

Let N_S be a countably infinite set of *shape names*, disjoint from N_N, N_C, and N_P. A *path expression* E is a regular expression built from the operator $\cdot$, symbols in N_P, and expressions p^- for each $p \in N_P$, where p^- is the inverse property of p.

[2] `https://www.w3.org/TR/shacl/#data-graph`

A *shape expression* ϕ is of the form:

$$\phi, \phi' ::= \top \mid s \mid B \mid c \mid \phi \wedge \phi' \mid \neg\phi \mid \geq_n E.\phi \mid E = p$$

where $s \in N_S$, $B \in N_C$, $c \in N_N$, n is a positive integer, E is a path expression and $p \in N_P$. In what follows, we may write $\phi \vee \phi'$ instead of $\neg(\neg\phi \wedge \neg\phi')$, $\exists E.\phi$ instead of $\geq_1 E.\phi$, and $\geq_n E$ instead of $\geq_n E.\phi$ if ϕ is $\top$. SHACL constraints are represented in the form of *(shape) constraints*, which are expressions of the form $s \leftarrow \phi$, with $s \in N_S$ and ϕ a shape expression. A *shape atom* is an expression of the form $s(a)$, with s a shape name and a a node. A *target* is an expression of the form (W, s), where s is a shape name and W takes one of the following forms:

- constant from N_N, also called *node target*,
- class name from N_C, also called *class target*,
- expression of the form $\exists p$ with $p \in N_P$, also called *subjects-of target*,
- expression of the form $\exists p^-$ with $p \in N_P$, also called *objects-of target*.

A *shapes graph*[3] is a pair (C, T), where C is a set of shape constraints such that each shape name occurs exactly once on the left-hand side of a shape constraint, and T is a set of targets. Informally, we call a set of constraints C *recursive* if there is a shape name s that directly or indirectly refers to itself in the shapes constraints. In the present paper, we assume non-recursive constraints.

Evaluation of shape expressions

A *(shapes) assignment* for a data graph G is an expansion $I = G \cup L$ of G with a set L of shape atoms such that a occurs in G for each $s(a) \in L$. We denote with $V(G)$ (resp. $V(I)$) the set of nodes appearing in G (resp. I). The evaluation of a shape expression ϕ over an assignment I is defined in terms of an evaluation function $[\![\cdot]\!]^I$, which maps any path expression E to a binary relation $[\![E]\!]^I \subseteq V(I) \times V(I)$, and any shape expression ϕ to a set of nodes $[\![\phi]\!]^I \subseteq V(I)$.

$$[\![\top]\!]^I = V(G) \qquad [\![s]\!]^I = \{a \mid s(a) \in I\}$$
$$[\![B]\!]^I = \{a \mid B(a) \in I\} \qquad [\![c]\!]^I = c$$
$$[\![\phi_1 \wedge \phi_2]\!]^I = [\![\phi_1]\!]^I \cap [\![\phi_2]\!]^I \qquad [\![\neg\phi]\!]^I = V(I) \setminus [\![\phi]\!]^I$$
$$[\![p]\!]^I = \{(a, b) \mid p(a, b) \in I\} \qquad [\![p^-]\!]^I = \{(a, b) \mid p(b, a) \in I\}$$
$$[\![\geq_n E.\phi]\!]^I = \{a \mid \{(a, b) \in [\![E]\!]^I \text{ and } b \in [\![\phi]\!]^I\} \geq n\} \qquad [\![E \cdot E']\!]^I = [\![E]\!]^I \circ [\![E']\!]^I$$
$$[\![E = p]\!]^I = \{a \mid \forall b : (a, b) \in [\![p]\!]^I \text{ iff } (a, b) \in [\![E]\!]^I\}$$

SHACL validation

We now present the semantics for validation of a shapes graph (C, T) by a data graph G. There are several validation semantics for SHACL [22, 6], which coincide on non-recursive SHACL. Here, we only present the supported model semantics [22]. In particular, we require that all the nodes that occur in C and T appear also in G.

▶ **Definition 1.** *Assume a SHACL document (C, T) and a data graph G. An assignment I for G is a* (supported) model *of C if $[\![\phi]\!]^I = s^I$ for all $s \leftarrow \phi \in C$. The data graph G* validates *(C, T) if there exists an assignment I for G such that (i) I is a model of C, and (ii) $[\![W]\!]^I \in s^I$ for every target $(W, s) \in T$.*

[3] `https://www.w3.org/TR/shacl/#shapes-graph`

Normal Form

To ease presentation, in the rest of the paper we focus on *normalized* sets of SHACL constraints. That is, each SHACL constraint is constructed using the following normal forms:

(NF1) $s \leftarrow \top$	*(NF2)* $s \leftarrow B$	*(NF3)* $s \leftarrow c$	
(NF4) $s \leftarrow s_1 \wedge \cdots \wedge s_n$	*(NF5)* $s \leftarrow \neg s'$	*(NF6)* $s \leftarrow \geq_n E.s'$	*(NF7)* $s \leftarrow E = p$

It was shown in [6] that a set of constraints C can be transformed in polynomial time into a set of constraints C' in normal form such that for every data graph G and target T, G validates (C, T) if and only if G validates (C', T). Note that the normalization process recursively introduces fresh shape names for sub-expressions until the normal form is reached. Moreover, if the input shapes graph is non-recursive, then clearly the normalized one is also non-recursive.

▶ **Example 2.** Assume a shapes graph with shape constraints for StudentShape (left) and a data graph (right), written in Turtle syntax.

```
    :StudentShape a sh:NodeShape;           :Ben :enrolledIn :C1 .
        sh:targetNode :Ben;
        sh:property [
            sh:path :enrolledIn;
            sh:qualifiedMinCount 1;
            sh:qualifiedValueShape [
                sh:class :Course;
            ]
        ] .
```

The shapes graph states that each node conforming to StudentShape must be enrolled in at least one course and it should be verified at node *Ben*. The target is represented in the Turtle syntax by the *sh:targetNode* property, and the statements below *sh:property* define the constraints to be verified on the node *Ben*. The constraints specify via *sh:path* to check whether there is an *enrolledIn* outgoing edge from target Ben to at least 1 node (*sh:qualifiedMinCount*) that is an instance of the *Course* class, specified with *sh:class*. In the abstract syntax, we write the data graph G, set of constraints C, and targets T as follows:

$$G = \{\mathit{enrolledIn}(\mathit{Ben}, C_1)\},$$
$$C = \{\mathsf{StudentShape} \leftarrow \exists \mathit{enrolledIn}.\mathit{Course}\},$$
$$T = \{\mathsf{StudentShape}(\mathit{Ben})\}$$

The normalized version C' of C contains the constraints

$$\mathsf{StudentShape} \leftarrow \exists \mathit{enrolledIn}.s,$$
$$s \leftarrow \mathit{Course}$$

where s is a fresh shape name. Clearly, extending G by assigning the shape name StudentShape to *Ben* does not satisfy the target StudentShape(*Ben*), since *Ben* is not enrolled in any (node of class) *Course*. Hence, G does not validate (C, T).

We note that the translation of SHACL shapes represented in RDF syntax into the abstract syntax is described in Section 7 as part of the ASP implementation.

2.2 Answer Set Programming

Answer Set Programming (ASP) is a declarative problem solving paradigm based on Logic Programming and Nonmonotonic Reasoning [25]. The idea of ASP is to describe (or model) a problem as a nonmonotonic logic program and let an ASP solver compute the solutions to the problem. The program consists of logic rules (with head and body), facts (head without a body) and constraints (body without a head). ASP has extensions for negation and disjunction in rule heads for modeling nonmonotonic programs. The ASP paradigm lets users focus on describing the problem itself instead of coding a solution for it, which is computed by an ASP solver. The solutions, so-called answer sets or stable models, are minimal models of a logic program. There can be none, one, or multiple stable models for a program.

The implementation of the repair program is done by generating an ASP program from the SHACL shapes and the data graph. By building on the minimality of ASP models, the repair program provides minimal repairs as part of the minimal models.

We introduce here some basic notation about Answer Set Programming (ASP) used throughout the paper and refer to [25] for more details on the language. We assume countably infinite, mutually disjoint sets $\mathsf{Preds} \supset N_C \cup N_P$ and Var of *predicate symbols*, and *variables*, respectively. A *term* is a variable from Var or a node from N_N. The notion of an *atom* is extended from RDF atoms here to include expressions of the form $q(t_1, \ldots, t_n)$, where $q \in \mathsf{Preds}$ is an n-ary predicate symbol and $t_1, \ldots, t_n$ are terms; an atom is *ground* if its terms are nodes. A database is a set of ground atoms. An answer set program consists of a set of rules of the form $\psi \leftarrow \varphi$, where φ may be a conjunction of positive and negated atoms, and ψ is a (possibly empty) disjunction of atoms. We may call ψ the *head* of the rule and φ the *body* of the rule. We may write a rule $h_1, \ldots, h_n \leftarrow \varphi$ instead of a set of rules $h_1 \leftarrow \varphi, \ldots, h_n \leftarrow \varphi$. Roughly, a rule is satisfied by a database D in case the following holds: if there is a way to ground the rule by instantiating all its variables such that D contains the positive atoms in the body of the instantiated rule and does not contain the negative atoms, then it contains some atom occurring in the head of the rule. The semantics of answer set programs is given in terms of *stable models*. Intuitively, a stable model for (G, P), where G is a data graph and P a program, is a database D that minimally extends G to satisfy all rules in P. We illustrate answer set programs with an example about 3-colorability.

▶ **Example 3.** Let $G = \{\mathsf{edge}(a, b), \mathsf{edge}(b, c), \mathsf{edge}(c, a), N(a), N(b), N(c)\}$ be a data graph storing a triangle over the nodes a, b, and c, and let P be a program with the following rules:

$$\begin{array}{ll} R(X) \vee B(X) \vee G(X) \leftarrow N(X) & \leftarrow \mathsf{edge}(X, Y), R(X), R(Y) \\ \leftarrow \mathsf{edge}(X, Y), B(X), B(Y) & \leftarrow \mathsf{edge}(X, Y), G(X), G(Y). \end{array}$$

P states that every node must be colored with red R, blue B, or green G, and adjacent vertices must not be colored with the same color. Clearly, there are three possibilities to color the nodes and hence, three answer sets for (G, P) that minimally extend G to satisfy the rules. E.g., one stable model is $M = G \cup \{R(a), B(b), G(c)\}$.

To implement the repair program we selected the clingo ASP system for grounding and solving logic programs [29], which is available online.[4] Clingo provides additional features, like optimization functions, that will be present in the repair program and they will be explained when used by the specific rules.

[4] `https://potassco.org/clingo/`

3 Exploring Design Choices for SHACL Repairs

In this section, we introduce the notion of repairs that we use in this work, analyze the kind of repairs that may be desirable in practice, and describe the design choices we will consider for the repair program we propose. For formally defining repairs, we use the notion of explanations for non-validation in SHACL introduced in [3], where a repair is a set of facts that are added or removed from the input data graph so that the resulting graph validates the input shapes graph. We recall a slightly modified definition here.

▶ **Definition 4.** *A* repair problem *is a tuple $\Psi = (G, C, T)$, where G is a data graph, and (C, T) is a shapes graph such that G does not validate (C, T). A* repair *for Ψ is a pair (A, D) of two sets of RDF atoms, where $D \subseteq G$, such that $(G \setminus D) \cup A$ validates (C, T).*

Note that the original definition of a repair problem in [3] includes a *hypothesis* set H given in the input, which allows to limit the space of possible additions by imposing the inclusion $A \subseteq H$. In this work, we do not consider a given input set H of hypotheses and thereby do not limit the possible additions in such a way. Instead, we assume that H consists of all the atoms that can be formed from the class and property names, the nodes occurring in the input, and a finite set of fresh nodes that will be introduced by the repair rules. Since the input constraints are non-recursive, it can be easily argued that the number of fresh nodes introduced by the repair program is (possibly exponentially) bound by the number of constraints. Thus, in what follows, we omit H from the input and write (G, C, T) for a repair problem. In particular, when a property atom needs to be added in the repair, we prefer to introduce fresh nodes instead of reusing nodes from the input.

When designing a repair program, we need to make some choices. First, as also argued in [3], computing all possible repairs is not desirable: we naturally want the repairs to modify the data graph in a minimal way, i.e., additions and deletions that are not relevant for fixing the constraint violations should be excluded. For instance, the repair problem (G, C, T) in Example 2 can be solved, among other ways, by (i) adding to G the atom $\mathit{Course}(C1)$, (ii) by adding to G the atoms $\mathit{Course}(C2)$ and $\mathit{enrolledIn}(\mathit{Ben}, C2)$, or (iii) by adding to G the atoms $\mathit{Course}(C1)$ and $\mathit{Course}(C2)$. Observe that (i) is a repair that is minimal in terms of the *number* of changes that are performed, i.e., cardinality-minimal. The repair (ii) can also be considered minimal, but in the sense of subset-minimality: observe that neither $\mathit{Course}(C2)$ nor $\mathit{enrolledIn}(\mathit{Ben}, C2)$ *alone* suffice to fix the constraint violation. The repair (iii) is not minimal in either sense, because the addition of $\mathit{Course}(C1)$ alone is sufficient to perform the repair.

Another issue is how to repair cardinality constraints of the form (NF6). To satisfy them, we can either choose to generate fresh nodes, or we may try to reuse the existing nodes of the input data graph. There are scenarios where reusing nodes is not desired as we want to fix the violations while minimally changing the data graph. Reusing nodes may introduce wrong information from a real-world perspective and thus lower the quality of data. Consider the constraint

$$\mathsf{StudentShape} \leftarrow \exists \mathit{hasStudentID}$$

specifying that students must have a student ID and let the data graph include the atom $\mathit{hasStudentID}(\mathit{Ben}, \mathit{ID}1)$. To validate the target $\mathsf{StudentShape}(\mathit{Ann})$, a meaningful repair would be to generate a new value $\_\mathit{ID}$ as placeholder value and add $\mathit{hasStudentID}(\mathit{Ann}, \_\mathit{ID})$ instead of reusing *ID1*. Reusing $\mathit{ID}1$ and adding $\mathit{hasStudentID}(\mathit{Ann}, \mathit{ID}1)$ would assign Ben's student ID to Ann, which is likely to be a problem from a real-world perspective (or, resp., might violate further constraints in a more elaborate scenario). Indeed, constructing an entirely new value that does not yet exist in the data graph comes without any underlying semantics beyond what the shape constraints describe, and therefore avoids unintentional (re)use from a real-world perspective. Such placeholders can be replaced in a later step by the user with meaningful real-world values.

Unfortunately, in turn forcing the repair program to always introduce fresh values for cardinality constraints may sometimes leave out expected (minimal) repairs and may even not produce any repairs at all. Consider the constraint

$$\mathsf{StudentShape} \leftarrow \exists \mathit{hasStudentID} \wedge \leq_1 \mathit{hasStudentID}$$

stating that students must have exactly one student ID. Let $G = \{\mathit{hasStudentID}(\mathit{Ann}, \mathit{ID}1)\}$ and assume we want to validate that Ann conforms to StudentShape. We may attempt to satisfy the constraint by adding the atoms $\mathit{hasStudentID}(\mathit{Ann}, \mathit{new}_1)$ for a fresh node new_1. However, then Ann would have two main addresses, which is not allowed. The only way to fix the violation is to reuse the node $\mathit{ID}1$.

Also, for the repair problem from Example 2, mentioned above, by forcing to introduce fresh values we would miss the intuitive minimal repair that simply adds $\mathit{Course}(C1)$. In conclusion, there are scenarios where reusing existing nodes may be desired and even necessary. However, to preserve the quality of the data as much as possible, we want to prioritize the introduction of fresh values whenever possible and reuse existing constants only when necessary. We study both versions. More precisely, in Section 4, we propose a repair program that always introduces fresh values, and in Section 5, we present the extended version that allows to reuse constants, but introduces fresh values whenever possible.

Also, Section 6 further extends the repair program to support repairs for path expressions E, which requires to add property atoms to generate property paths for additions and to delete property atoms to cut property paths for deletion.

4 Generating Repairs

In this section, we present an encoding of the repair problem to ASP that captures the violations and proposes changes to the data graph so that the violations are fixed. It is given in terms of an answer set program whose stable models will provide repairs as a set of additions and deletions to the input data graph. We are especially interested in cardinality-minimal repairs, that can be generated by the program. We do not consider subset-minimal repairs here and leave it for future work.

To ease presentation, we describe here the encoding for a restricted setting, where only existential constraints of the form $s \leftarrow \geq_1 p.s'$, i.e., a special case of cardinality constraints of form (NF6), are allowed; we label them with (NF6′). In particular, rules will always introduce fresh values to repair existential constraints. We refer to Section 5 for the extension that support unrestricted cardinality constraints and allows the reuse of constants from the input. Note that we also do not consider here constraints of the form (NF7). The unrestricted version of (NF6) and constraints of the form (NF7) will be treated in Section 6.

For the ASP encoding, we focus on shapes graphs (C, T), where C contains non-recursive constraints and T only contains node targets of the form $s(c)$. As for the latter, we can convert the other types of targets into node targets by simply *grounding* them w.r.t. the input data graph. More specifically, for a data graph G and an arbitrary shapes graph (C, T), we can generate an equivalent shapes graph (C, T_{gr}) over only node targets, where

$$T_{gr} = \{(c, s) \mid c \in [\![W]\!]^G \text{ for each target } (W, s) \in T\}$$

is the grounding of T w.r.t. G. It is easy to verify, that G validates (C, T) if and only if G validates (C, T_{gr}). However, this equivalence may not hold when considering repairs which change the input data graph and may update the grounding of the targets. In particular, the grounding may need to be recomputed after a repair is applied, as illustrated in the following example.

▶ **Example 5.** Consider a data graph $G = \{B_1(a), B_1(b), B_2(b)\}$ and the shapes graph (C, T), where C has the constraint $s \leftarrow B_2$ and the target is $T = \{(B_1, s)\}$. Clearly, G does not validate (C, T), and hence neither the corresponding shapes graph (C, T_{gr}), where $T_{gr} = \{(a, s), (b, s)\}$ is the grounding of T w.r.t. G. Consider the repair $R = (\emptyset, \{B_1(a)\})$, which removes only the fact $B_1(a)$ from G. Clearly, R is a repair for (G, C, T) since $G' = G \setminus \{B_1(a)\}$ validates (C, T), but it is not a repair for (G, C, T_{gr}). Clearly, every repair for (G, C, T_{gr}) is also a repair for (G, C, T).

Intuitively, the above issues appear because the grounding of the targets may change depending on the repair. If the repair modifies how class or property names in the targets are grounded, then a repair that works for the grounded version of the shapes graph may not necessarily be valid for the ungrounded version and vice versa.

▶ **Proposition 6.** *Let (C, T) be a shapes graph. Then, for every data graph G and pair $R = (A, D)$ that does not use the class and property names appearing in T, it holds that R is a repair for (G, C, T_{gr}) iff R is a repair for (G, C, T), where T_{gr} is the grounding of T w.r.t. G.*

To ensure that the repairs obtained from the ASP encoding do not use the class and property names appearing in the targets, it suffices to restrict the input shapes graphs (C, T) to not allow the shapes names in the constraints C to use the class and property names appearing in the targets T. We say that a shape name s *directly uses* ℓ, where ℓ is a shape, class, or property name, if there exists a constraint $s \leftarrow \phi$ in C such that ℓ occurs in ϕ. We say that a shape name s *uses* ℓ, if s directly uses ℓ, or if there exists a shape name s' such that: (1) s directly uses s' and (2) s' uses ℓ. A shapes graph (C, T) is target-restricted if, for every shape name s occurring in T, it is the case that s does not use in C any of the class and property names appearing in T. For target-restricted shapes graphs (C, T), the ASP program guarantees that repairs of the (C, T_{gr}) are also repairs of (C, T).

In the following, we assume non-recursive target-restricted shapes graphs and consider the ground versions over only node targets. For simplicity, we write node targets as shape atoms, that is we may write $T = \{s(a)\}$ instead of $T = \{(a, s)\}$. Note that if the input shapes graph is only over node targets, then it is automatically target-restricted.

4.1 Encoding into ASP

Assume a repair problem $\Psi = (G, C, T)$, where (C, T) is a non-recursive shapes graph, C is in normal form, and T is a set of node targets. We construct a program P_Ψ, such that the stable models of (G, P_Ψ) will provide repairs for Ψ. Following the standard notation for repairs in databases [14], we will use special constants to annotate atoms:

(i) t^{**} intuitively states that the atom is true in the repair,
(ii) t^{*} states that the atom is true in the input data graph or becomes true by some rule,
(iii) t states that the atom may need to be true, and
(iv) f states that the atom may need to be false.

Intuitively, the repair program implements a top-down target-oriented approach, and starts by first making true all the shape atoms in the target. From this on, the rules for constraints specified by the shapes capture violations on the targets in the rule body and propose repairs in the rule head using the annotations described above. The rules will add annotated atoms, which represent additions and deletions, that can be applied to the data graph to fix the violations. Additions and deletions can interact, eventually stabilizing into a model that generates a (not necessarily minimal) repair.

For every constraint specified by a shape in the shapes graph, the repair program P_Ψ consists of four kinds of rules:

$P_{\textsf{Annotation}}$ is a set of rules that collect existing atoms or atoms that are proposed to be in the repaired data graph.

$P_{\textsf{Repair}}$ is a set of rules that capture violations of the constraints by the data graph (also by missing data) and propose atoms to be added or deleted so that the fixed data graph satisfies the constraints.

$P_{\textsf{Interpretation}}$ is a set of rules that collect all the atoms that will be in the repaired data graph. These are existing atoms from the data graph or atoms proposed to be added by the program.

$P_{\textsf{Constraints}}$ is a set of rules that filter out models that do not provide repairs. For instance, we add rules that prohibit the same atom to be both added and deleted by the repair program.

We are now ready to describe the repair program.

Adding the shape atoms in the target as facts

First, for each atom $s(a) \in T$, we add the rule $s\_(a, t^*) \leftarrow$, where $s\_$ is a fresh binary relation. We note that T is constructed in a preliminary step to the repair program, meaning the repair additions and deletions will not change the grounding of the targets without applying the repair and recomputing.

$P_{\textsf{Annotation}}$

For each class name B, property name p and inverse property name p^- occurring in G and C, we create a new binary predicate $B\_$ and ternary predicates $p\_$ and $p^-\_$, respectively. We add the following rules to P_Ψ:

$$B\_(X, t^*) \leftarrow B(X) \qquad p\_(X, Y, t^*) \leftarrow p(X, Y)$$
$$B\_(X, t^*) \leftarrow B\_(X, t) \qquad p\_(X, Y, t^*) \leftarrow p\_(X, Y, t)$$

$$p\_(Y, X, t) \leftarrow p^-\_(X, Y, t) \qquad p^-\_(X, Y, t^*) \leftarrow p\_(Y, X, t^*)$$
$$p\_(Y, X, f) \leftarrow p^-\_(X, Y, f) \qquad p^-\_(X, Y, f) \leftarrow p\_(Y, X, f)$$

$P_{\textsf{Repair}}$

For each constraint $s \leftarrow \phi$ in C, we add specific rules that consider in the body the scenarios where s at a certain node is suggested to be true or false in the repair program, and propose in the head ways to make ϕ true or false, respectively. We note that the presence of negation in constraints may enforce that a shape atom is false at specific nodes. We present the repair rules for each normal form that ϕ can take, that is for each type of constraint of the form (NF1) to (NF5) and (NF6′). We add rules for both $s\_(X, t^*)$ and $s\_(x, f)$.

- If the constraint is of the form (NF1) or (NF3), then we do nothing here and treat them later as constraints.
- If ϕ is a class name B, that is of form (NF2), then we use the fresh binary predicate $B\_$ and add the rules:

$$B\_(X, t) \leftarrow s\_(X, t^*) \qquad B\_(X, f) \leftarrow s\_(X, f)$$

- If ϕ is of the form $s_1 \wedge \cdots \wedge s_n$, that is of form (NF4), then we use fresh binary predicates $s_i\_$ and add the rules:

$$s_1\_(X, t^*), \ldots, s_n\_(X, t^*) \leftarrow s\_(X, t^*) \qquad s_1\_(X, f) \vee \cdots \vee s_n\_(X, f) \leftarrow s\_(X, f)$$

- If ϕ is of the form $\neg s'$, that is of form (NF5), we add the rules:

$$s'\_(X,f) \leftarrow s\_(X,t^*) \qquad\qquad s'\_(X,t^*) \leftarrow s\_(X,f)$$

- If ϕ is of the form $\exists r.s'$, that is of form (NF6′), then we have to consider the scenarios where r is a property name p or an inverse property p^-. For the case where s is suggested to be true at X, i.e., for $s\_(X,t^*)$, we add a new p-edge from X to a fresh node and assign the node to s'. To this end, we use a function $@new(s,X,r)$, which maps a shape name s, a node X and a property name or inverse property name r to a new unique value Y. For the case where s is suggested to be false at X, i.e., for $s\_(X,f)$, we add disjunctive rules that, for all r-edges from X to some Y with s' true in Y, makes one of these atoms false.

$$s'\_(@new(s,X,r),t^*), r\_(X,@new(s,X,r),t) \leftarrow s\_(X,t^*)$$
$$r\_(X,Y,f) \vee s'\_(Y,f) \leftarrow s\_(X,f), r\_(X,Y,t^*)$$

$P_{\textsf{Interpretation}}$

For every class name B, property name p and inverse property name p^- occurring in the input, we add the following rules:

$$B\_(X,t^{**}) \leftarrow B\_(X,t^*), not\ B\_(X,f) \qquad p\_(X,Y,t^{**}) \leftarrow p\_(X,Y,t^*), not\ p\_(X,Y,f)$$
$$p^-\_(X,Y,t^{**}) \leftarrow p^-\_(X,Y,t^*), not\ p^-\_(X,Y,f)$$

Intuitively, these rules will generate the atoms that will participate in the repaired data graph, that is the atoms that were added to the data graph, and those atoms from the data graph that were not deleted by the rules.

$P_{\textsf{Constraints}}$

We add to P_Ψ sets of rules that will act as constraints and filter out models that are not repairs.

(1) For each constraint of the form $s \leftarrow \top$, that is of form (NF1), we add $\leftarrow s\_(Y,f)$.

(2) For each constraint of the form $s \leftarrow c$, that is of form (NF3), we add the rules:

$$\leftarrow s\_(X,t^*), X \neq c \qquad\qquad \leftarrow s\_(c,f)$$

(3) For each class name B, property name p and inverse property name p^- in the input, we add:

$$\leftarrow B\_(X,t), B\_(x,f) \qquad\qquad \leftarrow p\_(X,Y,t), p\_(X,Y,f)$$
$$\leftarrow p^-\_(X,Y,t), p^-\_(X,Y,f)$$

Rules (1) and (2) ensure that models preserve constraints of type (NF1) and (NF3) that cannot be repaired, and (3) ensures that no atom is *both inserted and deleted* from G. The atoms marked with t^{**} in a stable model of P_Ψ form a repaired data graph that validates (C,T).

▶ **Theorem 7.** *Assume a repair problem $\Psi = (G,C,T)$. For every stable model M of (G,P_Ψ), the data graph G' validates (C,T), where G' is the set of all atoms of the form $B(a)$, $p(a,b)$ such that $B\_(a,t^{**})$ and $p\_(a,b,t^{**})$ are in M.*

Proof. We need to show that for every stable model M of (G,P_Ψ), the data graph G' validates (C,T), and in particular that the updated graph G' satisfies two conditions: (i) there exists an assignment I for G' that is a model of C, and (ii) all targets in T are satisfied under I. The proof leverages the structure of the ASP program P_Ψ and its components. First, we note that G' consists of all atoms $B(a)$ such that $B\_(a, t^{**}) \in M$ and all atoms $p(a, b)$ such that $p\_(a, b, t^{**}) \in M$; note that these atoms are confirmed to be true, that is atoms from the input data graph that were not deleted and atoms that were added by the rules. This follows from the rules in $P_{\text{Interpretation}}$ and the constraints in $P_{\text{Constraints}}$ (Rule 3), which ensure that no atom is both added and deleted.

It suffices to show that there exists an assignment I, such that for every $s(a) \in T$ it holds that $a \in [\![\varphi]\!]^I$, where $s \leftarrow \varphi$ is the constraint for s in C. Clearly, since the constraints are non-recursive, such assignment I can be extended minimally to satisfy all constraints in C for all nodes in G'.

Let M be a stable model of P_ψ. We define $I = G' \cup L$, where L contains all the shape atoms $s(a)$ such that $s\_(a, t^*) \in M$ and $s\_(a, f) \notin M$. By construction, all targets T occur in L. We show a stronger claim: for every node a and shape s with $s(a) \in I$, then $a \in [\![\varphi]\!]^I$ where $s \leftarrow \varphi \in C$. We prove this claim by induction on the depth $d(s)$ of shape s in the dependency graph. Let $s \leftarrow \varphi \in C$ be the constraint defining s in C. Recall that there is a unique constraint for each shape name in C. The depth $d(s)$ is defined recursively as follows: $d(s) = 0$ if there is no shape names appearing in φ, $d(s) = 1$ if all shape names s' appearing in φ have depth 0, that is $d(s') = 0$; and $d(s) = i$ if there exists a shape names s' in φ with $d(s') = i - 1$ and for all other shape names s'' occurring in φ it holds that $d(s'') \leq i - 1$. To prove the claim, we consider normalized constraints of form (NF1)–(NF5), (NF6′).

For the base case ($d(s) = 0$), we consider all the types of constraints with depth 0, that is s does not depend on other shape names. First, consider a constraint $s \leftarrow \top$ of the form (NF1). The rule $\leftarrow s\_(Y, f)$ added in $P_{\text{Constraints}}$ ensures no node is assigned $s\_(Y, f)$, and hence $s\_(a, f) \notin M$. Clearly, that $s\_(a, t^*) \in M$, then $a \in [\![\top]\!]^I = V(G')$ trivially. Thus, $a \in [\![\varphi]\!]_I$. Next, consider a constraint $s \leftarrow B$ of the form (NF2). By construction the following two rules are in P_Ψ: $B\_(X, t) \leftarrow s\_(X, t^*)$ and $B\_(X, f) \leftarrow s\_(X, f)$. If $s(a)$ is true in I it means that $s\_(a, t^*) \in M$ and $s\_(a, f) \notin M$. Hence, $B\_(a, t) \in M$ and $B\_(a, f) \notin M$. The fact that $B\_(a, t) \in M$ implies that $B\_(a, t^*) \in M$, which together with $B\_(a, f) \notin M$ imply that $B\_(a, t^{**}) \in M$. By construction of M, this implies that $B(a) \in G'$ and hence $a \in [\![B]\!]^I$. Finally, assume $s(a)$ was added because of a constraint of type (NF3) $s \leftarrow c$. By construction, the rules $\leftarrow s\_(X, t^*), X \neq c$ and $\leftarrow s\_(c, f)$ are in P_ψ. If $s\_(a, t^*) \in M$, then $a = c$. Thus, $a = c \in [\![c]\!]^I$.

Assume the hypothesis holds for all s' with $d(s') < d(s)$. We show the claim for constraints $s \leftarrow \varphi$ of the form (NF4), (NF5), and (NF6′). Assume a constraint of the form $s \leftarrow s_1 \wedge \cdots \wedge s_n$ (NF4). The repair rule $s_1\_(X, t^*), \ldots, s_n\_(X, t^*) \leftarrow s\_(X, t^*)$ is in P_ψ. Since M is a model, then $s\_(a, t^*) \in M$, then $s_i\_(a, t^*) \in M$ for all $1 \leq i \leq n$. By induction hypothesis ($d(s_i) < d(s)$), it holds that $a \in [\![s_i]\!]^I$ for all i. Hence, $a \in \bigcap_i [\![s_i]\!]_I = [\![\varphi]\!]^I$. Next, assume the constraint is of the form $s \leftarrow \neg s'$ (NF5). The program P_ψ contains the repair rules $s'\_(X, f) \leftarrow s\_(X, t^*)$ and $s'\_(X, t^*) \leftarrow s\_(X, f)$. Since $s\_(a, t^*) \in M$, then $s'\_(a, f) \in M$. By hypothesis it follows that $a \notin [\![s']\!]^I$, so $a \in [\![\neg s']\!]^I$. Finally, we show for constraints of the form $s \leftarrow \exists p.s'$ (NF6′). The repair rules $s'\_(@new(s, X, p), t^*),\ p\_(X, @new(s, X, p), t) \leftarrow s\_(X, t^*)$ where Y adds a fresh node $@new(s, X, p)$ is in P_ψ. Let the fresh node for $@new(s, X, p)$ be c. By hypothesis, if $s\_(a, t^*) \in M$, then $p\_(a, c, t) \in M$ and $p\_(a, c, t^{**}) \in M$ (no deletion), so $p(a, c) \in G'$. Moreover, $s'\_(c, t^*) \in M$ implies $s'(c) \in I$ (by induction hypothesis ($d(s') < d(s)$). Hence, $a \in [\![\exists p.s']\!]^I$. ◀

We note that this theorem carries over to all the extensions, the version with cardinality constraints and constants and the version with property paths, we propose in this paper. It is easy to see that, since the rules are non-recursive in essence,[5] the number of fresh nodes that can be introduced in a stable model is in the worst-case exponential in the size of the input constraints. However, we do not expect to see this behavior often in practice.

[5] Technically speaking, the repair rules above may be recursive. However, if the annotation constants t, f, t^*, t^{**} are seen as part of the predicate's name (instead of being a fixed value in the last position), then the rules are non-recursive.

We illustrate the repair program with a representative example.

▶ **Example 8.** Consider the repair problem $\Psi = (G, C', T)$ from Example 2, where C' is the normalized version of C. We construct the repair program P_Ψ as follows.

For $P_{\text{Annotation}}$ we use fresh predicates *enrolledIn_* and *Course_*. The rules for *Course_* are $\mathit{Course\_}(X, t^*) \leftarrow \mathit{Course\_}(X, t)$, and $\mathit{Course\_}(X, t^*) \leftarrow \mathit{Course}(X)$; the rules for *enrolledIn_* are analogous. Intuitively, these rules will initially add the atom $\mathit{enrolledIn\_}(\mathit{Ben}, C_1, t^*)$ to the stable model. For P_{Repair}, we add the following rules, where F stands for the function $@\mathit{new}(\mathsf{StudentShape}, X, \mathit{enrolledIn})$.

$$\begin{aligned}
\mathit{enrolledIn\_}(X, F, t), s\_(F, t^*) &\leftarrow \mathsf{StudentShape\_}(X, t^*).\\
\mathit{enrolledIn\_}(X, Y, f) \vee s\_(Y, f) &\leftarrow \mathsf{StudentShape\_}(X, f), \mathit{enrolledIn\_}(X, Y, t^*)\\
\mathit{Course\_}(X, t) &\leftarrow s\_(X, t*)\\
\mathit{Course\_}(X, f) &\leftarrow s\_(X, f)
\end{aligned}$$

Intuitively, these rules together with the rules in $P_{\text{Annotation}}$ will add to the stable model the two atoms $\mathit{enrolledIn\_}(\mathit{Ben}, \mathit{new}_1, t^*)$ and $\mathit{Course\_}(\mathit{new}_1, t^*)$ with a fresh node new_1. For $P_{\text{Interpretation}}$, we add: $\mathit{Course\_}(X, t^{**}) \leftarrow \mathit{Course\_}(X, t^*), \mathit{not}\ \mathit{Course\_}(X, f)$ for *Course_* and proceed analogously for *enrolledIn_*. For $P_{\mathit{Constraints}}$, we add the (constraint) rule: $\leftarrow \mathit{Course\_}(X, t), \mathit{Course\_}(X, Y, f)$ for *Course_* and proceed analogously for *enrolledIn_*. Since no atom labeled with "f" is generated by the rules, then the three atoms mentioned above will be annotated with "t^{**}" by the rules in $P_{\text{Interpretation}}$.

Thus, there is one stable model with the atoms $\mathit{enrolledIn\_}(\mathit{Ben}, C_1, t^{**})$, $\mathit{enrolledIn\_}(\mathit{Ben}, \mathit{new}_1, t^{**})$ and $\mathit{Course\_}(\mathit{new}_1, t^{**})$. The corresponding atoms $\mathit{enrolledIn}(\mathit{Ben}, C_1)$, $\mathit{enrolledIn}(\mathit{Ben}, \mathit{new}_1)$ and $\mathit{Course}(\mathit{new}_1)$ will form the repaired data graph G' that validates (C', T). Hence, the only repair is $(A, \emptyset)$, where A contains $\mathit{enrolledIn}(\mathit{Ben}, \mathit{new}_1)$ and $\mathit{Course}(\mathit{new}_1)$.

Additions and Deletions. A stable model of the repair program contains atoms annotated with "t^{**}", which represent all the atoms that have to be in the repaired data graph, i.e., it may include original atoms from the input data graph and new atoms added by the repair rules. However, we want to represent repairs as sets of atoms that are added to and deleted from the input data graph. To achieve this, we use two fresh unary predicates *add* and *del*, and we add rules that "label" in a stable model with the label *add* all the atoms annotated with "t^{**}" that were not originally in the data graph, and label with *del* all the atoms annotated with "f" that were originally in the data graph. To this aim, for every class name B and property name p in the input, we introduce a function symbol (with the same name) whose arguments are the tuples of B and p, respectively. We show the rules for class names and property names. The rules for inverse property names are analogous.

$$\begin{aligned}
\mathit{add}(B(X)) &\leftarrow B\_(X, t^{**}), \mathit{not}\ B(X) & \mathit{del}(B(X)) &\leftarrow B\_(X, f), B(X)\\
\mathit{add}(p(X, Y)) &\leftarrow p\_(X, Y, t^{**}), \mathit{not}\ p(X, Y) & \mathit{del}(p(X, Y)) &\leftarrow p\_(X, Y, f), p(X, Y)
\end{aligned}$$

Considering Example 8, the repair will label all three atoms with *add* and it will label no atoms with *del*. The repair is represented by the additions $\mathit{add}(\mathit{enrolledIn}(\mathit{Ben}, C_1))$, $\mathit{add}(\mathit{enrolledIn}(\mathit{Ben}, \mathit{new}_1))$ and $\mathit{add}(\mathit{Course}(\mathit{new}_1))$. When applying this repair to the input data graph, it will restore consistency and the resulting repaired graph will validate against the constraints.

4.2 Generating Minimal Repairs

We are interested to generate cardinality-minimal repairs, i.e., repairs that make the least number of changes to the original data graph. More formally, given a repair $\xi = (A, D)$ for Ψ, ξ is *cardinality-minimal* if there is no repair $\xi' = (A', D')$ for Ψ such that $|A| + |D| > |A'| + |D'|$. As already noted in Section 3, repairs produced by our repair program built so far may not be cardinality-minimal, and this holds already for constraints without existential quantification. Consider the following example.

▶ **Example 9.** Let (G, C, T) be a repair problem, where G is empty, $T = \{s(a)\}$, and C contains the constraints: $s \leftarrow s1 \vee s2$, $s1 \leftarrow B_1$, and $s2 \leftarrow B_1 \wedge B_2$, where B_1 and B_2 are class names, and $s, s1, s2$ are shape names. To ease presentation, the constraints are not in normal form. Clearly, to validate the target $s(a)$ the repair program will propose to make $s1(a)$ or $s2(a)$ true. Hence, there will be two stable models: one generates a repair that adds $B_1(a)$, i.e., contains $add(B_1(a))$, and the other adds both $B_1(a)$ and, the possibly redundant fact, $B_2(a)$, i.e., contains $add(B_1(a))$ and $add(B_2(a))$.

To compute cardinality-minimal repairs, which minimize the number of additions and deletions, we introduce a post-processing step for our repair program that selects the desired stable models based on a cost function. We count the distinct atoms for additions and deletions and add a cost to each of them. The repair program should only return stable models that minimize this cost. More specifically, we add the $\#minimize\{1, W : add(W); 1, V : del(V)\}$ optimization rule to P_Ψ, which uses a cost 1 for each addition or deletion. We can also change the cost for additions and deletions depending on different repair scenarios, where one could have a higher cost for additions over deletions or vice versa.

4.3 Repairing Maximal Subsets of the Target Set

In this section, we discuss the situation where it is not possible to repair all of the targets, e.g., because of conflicting constraints in shape assignments to the target shape atoms or because of unsatisfiable constraints. Consider for instance the constraint $s \leftarrow B \wedge \neg B$, where B is a class name. Clearly, there is no repair for any shape atom over s in the target, since there is no way to repair the body of the constraint. Similarly, consider the constraints $s1 \leftarrow B$ and $s2 \leftarrow \neg B$ and targets $s1(a)$ and $s2(a)$; in this case adding $B(a)$ violates the second constraint and not adding it violates the first constraint. In both scenarios, the repair program will return no stable model, and hence, no repair. However, it still might be possible to repair a subset of the shape targets. In practice, we want to repair as many targets as possible. To support such a scenario, we introduce the concept of *maximal repairs*, which is a relaxation of the previous notion of repairs.

▶ **Definition 10.** *Let $\Psi = (G, C, T)$ be a repair problem. A pair (A, D) of sets of atoms is called a* maximal repair *for Ψ if there exists $T' \subseteq T$ such that (i) (A, D) is a repair for (G, C, T') and (ii) there is no $T'' \subseteq T$ with $|T''| > |T'|$ and (G, C, T'') having some repair.*

To represent this in the repair program, we add rules to non-deterministically select a target for repairing or skip a target if the repair program cannot repair it. This approach could be viewed similar in spirit to SHACL's *sh:deactivated* (`https://www.w3.org/TR/shacl/#deactivated`) directive that allows for deactivating certain shapes, with the difference that we "deactivate" *targets* instead of whole shapes which are automatically selected by the repair program based on optimization criteria. To this end, for each shape atom $s(a)$ in the input target set T, instead of adding all $s\_(a, t^*)$ as facts, we add rules to non-deterministically select or skip repair targets. If there are no conflicting or unsatisfiable constraints, then the stable models provide repairs for all the targets. However, if a repair of a target shape atom is not possible, because some shape constraints advise t as well as f, then the repair program will skip this target shape atom and the stable models will provide repairs only for the remaining shape atoms in T. We introduce two predicates *actualTarget* and *skipTarget*, where *actualTarget* represents a shape atom in the target that will be selected to repair, whereas *skipTarget* represents a shape atom in the target that is skipped and will not be repaired. For each $s(a)$ in T we add the rules:

$$\mathit{actualTarget}(a,s) \vee \mathit{skipTarget}(a,s) \leftarrow s(a) \qquad s\_(a,t^*) \leftarrow \mathit{actualTarget}(a,s)$$

We want to first repair as many target shape atoms as possible, and then minimize the number of additions and deletions needed for these repairs. To this end, we add the $\#\mathit{minimize}\{1@3, X, s : \mathit{skipTarget}(X,s)\}$ optimization rule to P_Ψ to minimize the number of skipped targets and the $\#\mathit{minimize}\{1@2, W : \mathit{add}(W); 1@2, V : \mathit{del}(V)\}$ rule to minimize the additions and deletion. Note that we choose a higher priority level for minimizing the number of skipped targets (1@3) than for minimizing additions and deletions (1@2). This rule minimizes the *skipTarget* atoms and therefore maximizes the *actualTarget* atoms based on the cardinality.

To illustrate the concept of *maximal repairs*, consider the following example.

▶ **Example 11.** Let (G, C, T) be a repair problem, where

$$\begin{aligned}
G &= \{\mathit{enrolledIn}(\mathit{Ben}, C_1)\}\\
T &= \{\mathsf{StudentShape}(\mathit{Ben}), \mathsf{TeacherShape}(\mathit{Ben})\}\\
C &= \{\mathsf{TeacherShape} \leftarrow \exists \mathit{teaches}.\mathit{Course} \wedge \neg\mathsf{StudentShape},\\
&\qquad \mathsf{StudentShape} \leftarrow \exists \mathit{enrolledIn}.\mathit{Course}\}
\end{aligned}$$

Thus, Ben is a target node for both StudentShape and TeacherShape. However, the first constraint states that a node cannot be a TeacherShape and StudentShape at the same time, which causes a contradiction when applied to Ben. A model of the program can only include either the atom $\mathit{actualTarget}(\mathit{Ben}, \mathsf{TeacherShape})$ or the atom $\mathit{actualTarget}(\mathit{Ben}, \mathsf{StudentShape})$. The repair program chooses $\mathit{skipTarget}(\mathit{Ben}, \mathsf{TeacherShape})$ to skip the shape atom $\mathsf{TeacherShape}(\mathit{Ben})$ and repairs the shape assignment for $\mathsf{StudentShape}(\mathit{Ben})$. In this case, this is the maximum possible number of repaired shape targets. Note that skipping either $\mathsf{StudentShape}(\mathit{Ben})$ or $\mathsf{TeacherShape}(\mathit{Ben})$ will result in a repair, but only the choice to skip $\mathsf{TeacherShape}(\mathit{Ben})$ will result in a cardinality-minimal repair.

Changing the optimization cost to skip targets allows to specify a preference among targets or shapes, thereby adapting to different repair scenarios.

5 Extension with Cardinality Constraints and Constants

In Section 4, we proposed a repair program for a restricted setting with cardinality constraints of the form $s \leftarrow \geq_n p.s'$ with $n = 1$. We now explain the extension to support cardinality constraints with unrestricted n, which are of form $s \leftarrow \geq_n p.s'$. We label such constraints with (NF6$''$). In addition to supporting the generation of new values, we now also allow to *reuse existing constants* from the input, which may even be necessary to generate some repair. E.g., consider an empty data graph G, the set of constraints $C = \{s \leftarrow \exists p.s', s' \leftarrow c\}$, and the target $T = \{s(a)\}$. Since the second constraint forces the selection of the constant c when generating a value for p, the only possible repair for (G, C, T) is to add the atom $p(a, c)$. However, we prioritize picking a fresh node over an existing one if picking a constant is not necessary.

More precisely, for a repair problem $\Psi = (G, C, T)$, where C contains constraints of the form (NF1)-(NF5) and (NF6$''$), we construct the repair program P'_Ψ, which contains all the rules from P_Ψ except for the rules for (NF6$'$) in P_{Repair}, i.e., the rules for existential constraints, which will be replaced by the rules described here.

Repairing cardinality constraints. If ϕ is of the form $\geq_n p.s'$, that is of form (NF6"), then for repairing the case $s\_(X, t^*)$ we need to insert at least n p-edges to nodes verifying s'. We first collect all nodes from C that are part of constraints to make sure that all necessary nodes are available to be picked for additions of property atoms. For every node c in C, we add: $\mathit{const}(c) \leftarrow$

- For the case where s is suggested to be true at X, i.e., for $s\_(X, t^*)$, we add the following rules.

$$\mathit{choose}(s, X, p, 0) \vee \cdots \vee \mathit{choose}(s, X, p, n) \leftarrow s\_(X, t^*) \tag{1}$$

$$p\_(X, @\mathit{new}(s, X, p, 1..i), t) \leftarrow \mathit{choose}(s, X, p, i), i \neq 0 \tag{2}$$

$$0\ \{p\_(X, Y, t) : \mathit{const}(Y)\}\ |\mathit{const}(Y)| \leftarrow s\_(X, t^*) \tag{3}$$

$$n\ \{s'(Y, t^*) : p\_(X, Y, t^{**})\}\ \mathit{max}(n, |\mathit{const}(Y)|) \leftarrow s\_(X, t^*) \tag{4}$$

In the following, we explain the rules (1) - (4) in detail. For adding atoms over $p\_$ to satisfy $\geq_n p.s'$, we either generate fresh nodes using the function *@new* or we pick from collected constants. For generating atoms with fresh nodes, we add a disjunctive rule (1) and use a fresh *choose* predicate, which is used to non-deterministically pick a number from 0 up to n for adding atoms over $p\_$ to fix the cardinality constraint. To add the actual atoms, we add a rule (2) that produces this number of atoms using the $@\mathit{new}$ function, which will generate a new unique value Y for every (s, X, p, i) tuple with s a shape name, p a property name, and $i \in \{1 \dots n\}$. With these two rules, we can generate as many atoms over $p\_$ as necessary to satisfy the cardinality constraint. Similarly to adding atoms with fresh nodes, we can also pick constants from C. We add a rule (3) to pick a number of 0 up to the maximum number of constants – using clingo's choice constructs, which allow to be parameterised with a lower and upper bound of elements from the head to be chosen – which will only pick constants if necessary because of constraints. In addition to adding atoms over p, we need to satisfy s' on a number of n nodes. We add a rule (4) to pick at least n, but might pick up to as many values as there are constants, so that we can satisfy the cardinality as well as any constraints that require specific constants. Note that an expression of the form $l\ \{W : V\}\ m$ intuitively allows to generate in the model a number between l and m W-atoms whenever V-atoms are also true.

- For the case where s is suggested to be false at X, i.e., for $s\_(X, f)$, we pick from all atoms $p(X, Y)$ to either delete the atom or falsify s' at Y. We add a disjunctive rule to pick one or the other (but not both).

$$\ell\ \{\psi_1 \vee \psi_2\}\ \ell \leftarrow s\_(X, f), \#\mathit{count}\{Y : p\_(X, Y, t^*)\} = m, m > (n-1) \tag{5}$$

where $\ell = m - (n-1)$, ψ_1 is the expression $p\_(X, Y, f) : p(X, Y), \mathit{not}\ s'\_(Y, f)$ and ψ_2 is $s'\_(Y, f) : p\_(X, Y, t^*), \mathit{not}\ p\_(X, Y, f)$. To make $s\_$ false at X, we have two disjunctive options that we can falsify. The first option is to falsify the $p\_$ atom. This can only be selected if $s'\_$ was not falsified at node Y. The second option is to falsify the $s'\_$ at node Y, which in return should only be possible if the $p\_$ atom was not falsified. By picking $m - (n-1)$ choices, we make sure that only the maximum allowed cardinality will be in the repaired graph.

Constant Reuse Optimization. The rules above are allowed to pick any constants that are needed to satisfy constraints in the current model. However, we want to pick a constant from C only if it is necessary to satisfy a constraint. To achieve this, for every constraint of the form $s \leftarrow \geq_n p.s'$, that is of the form (NF6), we add the $\#\mathit{minimize}\{1@1, X, Y : p\_(X, Y, t), \mathit{const}(Y)\}$ optimization rule to P'_Ψ that minimizes the use of constants among the different minimal repairs. We choose a lower priority level (1@1) for minimizing the use of constants after minimizing additions and deletions with a priority (1@2) and after minimizing the number of skipped target atoms (1@3). We first want to have minimal repairs and then among them to pick the ones with a minimal number of constants. Note that this encoding may produce different repairs from the encoding in Section 3 on the same example as illustrated below.

▶ **Example 12.** Consider again Example 8. The repair program P'_Ψ will generate three repairs.

$$A_1 = \{Course(C_1)\}, \qquad D_1 = \{\}$$
$$A_2 = \{Course(C_1), enrolledIn(Ben, new_1)\}, \qquad D_2 = \{\}$$
$$A_3 = \{enrolledIn(Ben, new_1), Course(new_1)\}, \qquad D_3 = \{\}$$

The first repair will only add $Course(C_1)$. Intuitively, rule (1) adds $choose(s, X, p, 0)$ to the model, rule (2) and (3) will not add atoms, and rule (4) adds $s\_(C_1, t^*)$ which together with the other rules treated in Example 8 add $Course\_(C_1, t)$ and $Course\_(C_1, t^{**})$, and hence result in the repair adding $Course(C_1)$. The second repair will add $enrolledIn(Ben, new_1)$ in addition to $Course(C_1)$ because of picking $i = 1$ in rule (1), and generating a fresh value new_1 in (2), while still picking C_1 for $s\_$. The third repair will assign new_1 to $s\_$, thus resulting in the repair $(A, \emptyset)$ from Example 8. The optimization feature will return only the minimal repair that only adds $Course(C_1)$.

6 Extension with Property Path Repairs

The repair program P'_Ψ from the previous section provides repairs for shape constraints without path expressions. More precisely, it can support constraints of the form (NF1)-(NF5) and a restriction of constraints of the form $s \leftarrow \geq_n E.s'$, that is of form (NF6), where E can only be a property name instead of a path expression. We now show the ASP rules to support path expressions and path equality, and hence unrestricted constraints of the form (NF6) and (NF7), respectively. This completes our encoding into ASP for the SHACL fragment considered in this work.

For a repair problem $\Psi = (G, C, T)$, where C is a non-recursive set of SHACL constraints of the form (NF1)-(NF7) and T is a set of node targets, we construct a repair program P^{comp}_Ψ whose stable models provide repairs for Ψ. In particular, P^{comp}_Ψ contains all the rules from the repair program P_Ψ from Section 4 except for the rules for (NF6′) in P_{Repair}. These rules will be replaced with more general rules that capture the full constraints of the form (NF6). We will also add rules that capture constraints of the form (NF7). As in Section 5, we prioritize picking a fresh node over an existing one if picking a constant is not necessary.

▶ **Example 13.** Consider the following example. We introduce a new ReviewedPublicationShape, which verifies that a reviewed publication must have reviewers from at least three different institutions. We define this constraint using a property path with a sequence path as follows for a repair problem (G, C, T).

$$G = \{hasReviewer(Pub1, Rev1), hasReviewer(Pub1, Rev2), hasReviewer(Pub1, Rev3),$$
$$worksFor(Rev1, WUWien), worksFor(Rev2, WUWien), worksFor(Rev3, TUWien)\}$$
$$T = \{\mathsf{ReviewedPublicationShape}(Pub1)\}$$
$$C = \{\mathsf{ReviewedPublicationShape} \leftarrow \geq_3 hasReviewer \cdot worksFor\}$$

We assign ReviewedPublicationShape to $Pub1$. For the shape assignment to validate, we need property paths to reach at least 3 different nodes starting from node $Pub1$. However, although there are three different property paths to nodes via the reviewers $Rev1, Rev2$ and $Rev3$, only two different nodes, $WUWien$ and $TUWien$, are reachable. We need at least one additional node to be reachable via the property path to make ReviewedPublicationShape validate $Pub1$.

To repair property paths, we will first add an auxiliary rule that will be useful in the rest of the section. The idea is that for each constraint with a path expression occurring in the body of the constraint, we want to collect all the pairs of nodes that the path connects and that are relevant for the particular constraint. Specifically, for every constraint of the form $s \leftarrow \geq_n E.s'$, that is of form (NF6), or of the form $s \leftarrow E = p$, that is of form (NF7), we collect all the pairs of nodes that intuitively are the start and end of E-paths from nodes verifying s. Assume E is of the form $r_1 \cdot r_2 \cdots\cdot r_u$ and each r_i is either a property name p_i or an inverse property name p_i^-. We use a fresh auxiliary ternary predicate $p^{sE}\_$ to collect such pairs and add to P_Ψ^{comp} for each such constraint a rule:

$$p^{sE}\_(X,Y,t^*) \leftarrow s\_(X,\_), r_1\_(X,X_1,t^*),\ldots,r_u\_(X_{u-1},Y,t^*) \tag{6}$$

For simplicity, we call instances (a,b) of $p^{sE}\_$ pairs of *source* and *end* nodes, representing the source node a verifying the shape name s and the end node b reachable from a through the path E.

Repairing constraints of form (NF6)

Assume ϕ is of the form $\geq_n E.s'$, that is the constraint for s is $s \leftarrow \geq_n E.s'$, where E is of the form $r_1 \cdot r_2 \cdots\cdot r_u$. Intuitively, for repairing the case where s is suggested to be true, that is for $s\_(X,t^*)$, we need to make sure there are E-paths to at least n nodes verifying s'. Analogously, for repairing the case where s is suggested to be false, that is for $s\_(X,f)$, we need to make sure that less than n nodes are reachable from X through E-paths. We now present the rules for each case, which will be added in the P_{Repair} part of the program.

- We describe the rules for the case when $s\_(X,t^*)$. We first add the following rules:

$$\mathit{choose}(s,X,p^{sE},0) \vee \cdots \vee \mathit{choose}(s,X,p^{sE},n) \leftarrow s\_(X,t^*) \tag{7}$$
$$p^{sE}\_(X,@\mathit{new}(s,X,p^{sE},1..i),t) \leftarrow \mathit{choose}(s,X,p^{sE},i), i \neq 0 \tag{8}$$
$$0\ \{p^{sE}\_(X,Y,t) : \mathit{const}(Y)\}\ |\mathit{const}(Y)| \leftarrow s\_(X,t^*) \tag{9}$$
$$n\ \{s'\_(Y,t^*) : p^{sE}\_(X,Y,t^{**})\}\ \mathit{max}(n,|\mathit{const}(Y)|) \leftarrow s\_(X,t^*) \tag{10}$$

To make sure we consider all E-paths in G and do not generate more values than needed, we make use of the auxiliary predicate $p^{sE}\_$. We add the rules (7)-(10) analogously to the rules (1)-(4) from the previous section, which generate just enough fresh values Y as end nodes in $p^{sE}\_$ to satisfy the cardinality n. Similarly, we pick a number up to the maximum number of constants for $p^{sE}\_$ and we pick a number of n end nodes in $p^{sE}\_$ to satisfy s'. For constant reuse optimization, we add the $\#\mathit{minimize}\{1@1, X, Y : p^{sE}\_(X,Y,t), \mathit{const}(Y)\}$ optimization rule to P_Ψ^{comp}.

We add rules, which for each property r_i at some node X_v suggest to either generate a fresh value to be r_i-connected to X_v or keep some existing r_i atom. We first add the case for generating r_1 separately and then for each $1 \leq i \leq u-2$, we add the following rules:

$$\mathit{choose}(s,X,r_1,0) \vee \mathit{choose}(s,X,r_1,1) \leftarrow p^{sE}\_(X,Y,t) \tag{11}$$
$$r_1\_(X,@\mathit{new}(s,X,r_1,1),t) \leftarrow \mathit{choose}(s,X,r_1,1) \tag{12}$$
$$\mathit{choose}(s,X_i,r_{i+1},0) \vee \mathit{choose}(s,X_i,r_{i+1},1) \leftarrow p^{sE}\_(X,Y,t), r_1\_(X,X_1,t^{**}),\ldots,r_i\_(X_{i-1},X_i,t^{**}) \tag{13}$$
$$r_{i+1}\_(X_i,@\mathit{new}(s,X_i,r_{i+1},1),t) \leftarrow \mathit{choose}(s,X_i,r_{i+1},1), r_1\_(X,X_1,t^{**}),\ldots,r_i\_(X_{i-1},X_i,t^{**}) \tag{14}$$

Note that in the above rules X_0 stands for X and X_u stands for Y. We have generated the paths from nodes verifying s up to r_{u-1}. The following rule allows to generate last edges r_u from X_{u-1} to some Y that can either be a constant generated by rule (9), or a fresh value generated by rule (8). This rule together with rule (10) and rule (6) will make sure there are enough E-paths connected to enough values to satisfy the cardinality n. We add the rule:

$$0\ \{r_u\_(X_{u-1}, Y, t)\}\ 1 \leftarrow p^{sE}\_(X, Y, t), r_1\_(X, X_1, t^{**}), \ldots, r_{u-1}\_(X_{u-2}, X_{u-1}, t^{**}) \tag{15}$$

- For the case where a property shape s is suggested to be false at X with a cardinality of n, i.e., for $s\_(X, f)$, we pick from all atoms $p^{sE}\_(X, Y, t^*)$ to either cut the E-path or falsify s' at Y. We add the following rules:

$$\ell\ \{\psi_3 \vee \psi_4\}\ \ell \leftarrow s\_(X, f), \#count\{Y : p^{sE}\_(X, Y, t^*)\} = m, m > (n-1) \tag{16}$$

where $\ell = m - (n-1)$, ψ_3 is the expression $p^{sE}\_(X, Y, f) : p^{sE}\_(X, Y, t^*), not\ s'\_(Y, f)$ and ψ_4 is the expression $s'\_(Y, f) : p^{sE}\_(X, Y, t^*), not\ p^{sE}\_(X, Y, f)$.

$$\begin{aligned} &r_1\_(X, X_1, f) \vee \cdots \vee r_u\_(X_{u-1}, Y, f) \\ &\qquad\qquad \leftarrow p^{sE}\_(X, Y, f), r_1\_(X, X_1, t^*), \ldots, r_u\_(X_{u-1}, Y, t^*) \end{aligned} \tag{17}$$

We add rule (16) analogous to rule (5) to pick from all E-paths $p^{sE}\_(X, Y, t^*)$. To make $s\_$ false at X, we have two disjunctive options that we can falsify. The first option is to cut the E-path by falsifying one of the atoms $r_1 \cdot r_2 \cdots r_u$. This can only be selected if $s'\_$ was not falsified at node Y. The second option is to falsify the $s'\_$ at node Y, which in return should only be possible if the E-path from start node X to end node Y was not falsified via $p^{sE}\_(X, Y, f)$ atom. Again, by picking $m - (n-1)$ choices, we make sure that only the maximum allowed cardinality will be in the repaired graph. Consequently, we add rule (17) to cut paths, if an E-path was suggested to be false via $p^{sE}\_(X, Y, f)$ atom, by picking from all E-path atoms $r_1 \cdot r_2 \cdots r_u$ (exactly) one atom to falsify, thereby cutting the E-path. Note that multiple (potentially overlapping) cuts for different E-paths will be minimized regarding cardinality by the optimization rules.

Repairing constraints of form (NF7)

Finally, assume ϕ is of the form $E = p$, where E is a path of the form $r_1 \cdot r_2 \cdots r_u$ and p is a property name. Intuitively, for repairing the case where s is suggested to be true, that is for $s\_(X, t^*)$, we need to make sure there is a property atom $p(X, Y)$ for every E-path from X to Y and vice versa. Analogously, for repairing the case where s is suggested to be false, that is for $s\_(X, f)$, we need to make sure that for at least one E-path from X to Y there is no property atom $p(X, Y)$ or vice versa. We now present the rules for each case, which will be added in the P_{Repair} part of the program.

- For the case where s is suggested to be true at $p^{sE}\_(X, Y, t^*)$ with equality to a property $p(X, Y)$, we add the following rules to make sure that for every E-path from X to Y there is also a property atom $p(X, Y)$ in the repair. We repair this case by using a disjunctive rule to either add or delete property atoms.

$$p\_(X, Y, t) \vee p^{sE}\_(X, Y, f) \leftarrow s\_(X, t^*), p^{sE}\_(X, Y, t^*) \tag{18}$$

$$p^{sE}\_(X, Y, t) \vee p\_(X, Y, f) \leftarrow s\_(X, t^*), p\_(X, Y, t^*) \tag{19}$$

The above rules will make sure to generate properties and auxiliary properties to ensure the equality. Rule (18) suggests for every E-path from X to Y that is suggested to be true to either make true $p(X, Y)$ or to make false the E-path. Rule (19) does the same for every $p(X, Y)$ suggested to be true.

Now the auxiliary properties together with rules (11) to (15) will propose actual paths from the source to the end nodes generated in the auxiliary properties. Since these rules may or may not construct the paths, we add constraints (25) and (26), which together with interpretation rule (23) will enforce the generation of the entire path.

- For the case where s is suggested to be false, we add rules which make sure that there is at least one $p^{sE}\_(X, Y, t^*)$ with no $p\_(X, Y, t^*)$ or one $p\_(X, Y, t^*)$ with no $p^{sE}\_(X, Y, t^*)$. Again, we can repair this case by using a disjunctive rule to either add or delete property atoms.

$$notEquals(s, X, p^{sE}, p) \leftarrow s\_(X, f), p^{sE}\_(X, Y, t^*), not\ p\_(X, Y, t^*) \tag{20}$$

$$notEquals(s, X, p^{sE}, p) \leftarrow s\_(X, f), not\ p^{sE}\_(X, Y, t^*), p\_(X, Y, t^*) \tag{21}$$

$$\begin{aligned} &1\ \{p^{sE}\_(X, Y, f) : p\_(X, Y, t^*) \vee p\_(X, Y, f) : p^{sE}\_(X, Y, t^*)\vee \\ &p^{sE}\_(X, @new(s, X, p^{sE}, 1), t) \vee p\_(X, @new(s, X, p, 1), t)\}\ 1 \\ &\qquad\qquad\qquad \leftarrow s\_(X, f), not\ notEquals(s, X, p^{sE}, p) \end{aligned} \tag{22}$$

We add two rules (20) and (21), which use a fresh *notEquals* predicate, to collect cases where there is a $p^{sE}\_(X, Y, t^*)$ with no $p\_(X, Y, t^*)$ or there is a $p\_(X, Y, t^*)$ with no $p^{sE}\_(X, Y, t^*)$. If no such a case exists, then the repair rule (22) creates exactly one such case by choosing one of four disjunctive options. Either it suggests $p^{sE}\_(X, Y, f)$ where there is a $p\_(X, Y, t^*)$, suggests $p\_(X, Y, f)$ where there is a $p^{sE}\_(X, Y, t^*)$, suggests a fresh $p^{sE}\_(X, @new(s, X, p^{sE}, 1), t)$ or suggests a fresh $p\_(X, @new(s, X, p, 1), t)$. Each of these options will falsify the equality for $s\_(X, f)$.

$P_{\text{Interpretation}}$

We add an interpretation rule which makes $p^{sE}\_$ true for the repair if all property atoms on the path $r_1 \cdot r_2 \cdot \dots \cdot r_u$ are true in the repair.

$$\begin{aligned} &p^{sE}\_(X, Y, t^{**}) \\ &\qquad \leftarrow p^{sE}\_(X, Y, t^*), not\ p^{sE}\_(X, Y, f), r_1\_(X, X_1, t^{**}), \dots, r_u\_(X_{u-1}, Y, t^{**}) \end{aligned} \tag{23}$$

$P_{\text{Constraints}}$

We add a constraint rule for $p^{sE}\_$ to filter models that are not repairs.

$$\leftarrow p^{sE}\_(X, Y, t^*), p^{sE}\_(X, Y, f) \tag{24}$$

We also add constraints to filter models which do not repair an equality.

$$\leftarrow s\_(X, t^*), p^{sE}\_(X, Y, t^{**}), not\ p\_(X, Y, t^{**}) \tag{25}$$

$$\leftarrow s\_(X, t^*), p\_(X, Y, t^{**}), not\ p^{sE}\_(X, Y, t^{**}) \tag{26}$$

The above rules conclude our encoding into ASP of the SHACL fragment considered in this paper. The correctness of the encoding is captured in the following theorem.

▶ **Theorem 14.** *Assume a repair problem $\Psi = (G, C, T)$, where C is a non-recursive set of SHACL constraints of the form (NF1)-(NF7) and T is a set of node targets. We construct a repair program P^{comp}_{Ψ} where, for every stable model M of P^{comp}_{Ψ}, (A, D) is a maximal repair for Ψ, where (A, D) is obtained as follows:*

- *A contains all atoms of the form $B(a)$, $p(a, b)$ not occurring in G such that $B\_(a, t^{**})$ and $p\_(a, b, t^{**})$ are in M, and*
- *D contains all atoms of the form $B(a)$, $p(a, b)$ from G such that $B\_(a, f)$ and $p\_(a, b, f)$ in M.*

Idea. Let $\Psi = (G, C, T)$ be a repair problem where C is a non-recursive set of SHACL constraints in normal form (NF1)–(NF7) and T is a set of node targets. Let P_Ψ^{comp} be the ASP program constructed in Section 6. Let M be an arbitrary stable model of of $(G, P_\Psi^{\text{comp}})$, let (A, D) be the repair from M obtained as above, and let G' be the repaired data graph obtained as follows $G' = (G \cup A) \setminus D$. First, that (A, D) is a maximal repair for Ψ by Definition 10 means that there exists $T' \subseteq T$ such that (i) (A, D) is a repair for (G, C, T') and (ii) there is no $T'' \subseteq T$ with $|T''| > |T'|$ and (G, C, T'') having some repair. Intuitively, this is ensured by the soft constraints, which allow to skip node targets if a stable model for all the node target is not found. Let $T^* \subseteq T$ be the set of node targets $s(a)$ from T such that $s\_(a, t^*) \in M$. Then, by construction, $actualTarget(a, s) \in M$ for each such node target $s(a) \in T^*$, which together with the optimization rule that minimizes the $skipTarget$ atoms and M being a stable model, implies that there is no node target $s'(a') \in T \setminus T^*$ that could be repaired.

Finally, we show that (A, D) is indeed a repair for (G, C, T^*). Let I be a shape assignment defined as $I = G' \cup L$, where $L = \{s(a) \mid s\_(a, t^*) \in M \text{ and } s\_(a, f) \notin M\}$. Arguing analogously to the proof of Theorem 7, it suffices to show that for every node a and shape s with $s(a) \in I$, then $a \in [\![\varphi]\!]^I$ where $s \leftarrow \varphi \in C$. The shape assignment I can be naturally extended to a model of all the constraints by simply evaluating the bodies of the constraints starting from shapes with depth 0, updating I by adding the corresponding shape atoms, and then proceeding with depth 1 and so on until no new shape atoms are entailed. Let I' be the new shape assignment obtained this way. Clearly, I' satisfies all the constraints and includes the targets.

The proof is analogous to the proof of Theorem 7, that is by induction on the depth $d(s)$ of each shape name s in the dependency graph of C. We show the claim (the induction step) only for the additional constraints of the form (NF6) that are supported in P_Ψ^{comp}. The base case for constraints of the form (NF7) uses similar arguments.

Suppose $s(a) \in I$ and the constraint for s in C is $s \leftarrow \geq_n E.s'$ of the form (NF6). We need to show that $a \in [\![\geq_n E.s']\!]^I$, that is there exist at least n distinct nodes Y such that $(a, Y) \in [\![E]\!]_I$ and $s'(Y) \in I$. By construction of I, it must be that $s\_(a, t^*) \in M$ and $s\_(a, f) \notin M$.

From rule (7), the presence of $s\_(a, t^*) \in M$ implies that M contains exactly one atom of the form $choose(s, a, p^{sE}, i)$ for some $i \in \{0, \ldots, n\}$. If $i > 0$, then rule (8) ensures the presence of atoms $p^{sE}\_(a, @new(s, a, p^{sE}, j), t)$ in M for $j = 1, \ldots, i$ (i.e., i fresh nodes are created). In addition, rule (9) allows the selection of existing constants Y_k such that $p^{sE}\_(a, Y_k, t) \in M$. Rule (10) then enforces that at least n such nodes Y satisfy $s'\_(Y, t^*) \in M$.

Now consider each such Y for which $p^{sE}\_(a, Y, t^{**}) \in M$. Rule (23) guarantees that $(a, Y) \in [\![E]\!]^I$: it ensures that $p^{sE}\_(a, Y, t^*) \in M$, $p^{sE}\_(a, Y, f) \notin M$, and all atoms representing the path $r_1\_(a, X_1, t^{**}), \ldots, r_u\_(X_{u-1}, Y, t^{**})$ are present in M. Additionally, rules (11)–(15) construct the actual path edges to validate this.

Since the depth of s' is strictly less than that of s, we apply the induction hypothesis: for all Y with $s'\_(Y, t^*) \in M$, it follows that $s'(Y) \in I$, that is $Y \in [\![s']\!]^I$.

The nodes Y are distinct by construction since fresh nodes are uniquely generated via $@new$, and constants are inherently distinct. Finally, rule (10) ensures that there are at least n such distinct Y satisfying both $(a, Y) \in [\![E]\!]^I$ and $Y \in [\![s']\!]^I$. Hence, $a \in [\![\geq n\, E.s']\!]^I$, as required. ◀

In the following, we illustrate the property path repair program with examples.

▶ **Example 15.** We revisit Example 13 and change the constraints for ReviewedPpublicationShape to not only validate for three different institutions, but also to validate that at least three different reviewers review a publication. We define these two constraints using property paths, where one path is a sequence path. To illustrate the repair for inverse paths, we exchange the *hasReviewer* property with a *reviews* property, which navigates backwards from reviewer to publication. We also change G to include two reviewers for $Pub1$ with two different institutions $WUWien$ and $TUWien$.

$$\begin{aligned}
G &= \{\mathit{hasReviewer}(Pub1, Rev1), \mathit{hasReviewer}(Pub1, Rev2),\\
&\quad \mathit{worksFor}(Rev1, WUWien), \mathit{worksFor}(Rev2, TUWien)\}\\
T &= \{\mathsf{ReviewedPublicationShape}(Pub1)\}\\
C &= \{\mathsf{ReviewedPublicationShape} \leftarrow \geq_3 \mathit{reviews}^- \wedge \geq_3 \mathit{reviews}^- \cdot \mathit{worksFor}\}
\end{aligned}$$

The ReviewedPublicationShape contains two property constraints. The first constraint is a minimum cardinality of three on the (inverse) path *reviews*$^-$, which indicate a reviewer for a publication. The second constraint is again a minimum cardinality of three on the (sequence) path *reviews*$^-$ $\cdot$ *worksFor*, which indicate that there have to be at least three institutions that the reviewers work for. With these two constraints we express that there need to be at least three reviewers from at least three different institutions for a reviewed publication. We represent the path for *reviews*$^-$ with an auxiliary property atom $p_1^{sE}\_\_$ and the path for *reviews*$^-$ $\cdot$ *worksFor* with an auxiliary property atom $p_2^{sE}\_\_$.

We assign ReviewedPublicationShape to $Pub1$. However, it does not validate. There are only two reviewers $Rev1$ and $Rev2$ with institutions $WUWien$ and $TUWien$, respectively. To repair the shape assignment, the repair program picks three $p_1^{sE}\_\_$ atoms to be made true for the path *reviews*$^-$ and three $p_2^{sE}\_\_$ atoms to be made true for the path *reviews*$^-$ $\cdot$ *worksFor* via rule (10). To satisfy this rule, it generates one fresh value for each of $p_1^{sE}\_\_$ and $p_2^{sE}\_\_$ via rules (7) and (8). As a further consequence, the repair program adds fresh values for property atoms *reviews*$^-$ and *worksFor* via rules (11), (12) and (15), thereby generating the paths to satisfy both cardinality constraints of $\geq_3$. We get the following minimal repairs.

$$\begin{aligned}
A_1 &= \{\mathit{reviews}(new_1, Pub1), \mathit{worksFor}(Rev1, new_2)\}, & D_1 &= \{\}\\
A_2 &= \{\mathit{reviews}(new_1, Pub1), \mathit{worksFor}(Rev2, new_2)\}, & D_2 &= \{\}\\
A_3 &= \{\mathit{reviews}(new_1, Pub1), \mathit{worksFor}(new_1, new_2)\}, & D_3 &= \{\}
\end{aligned}$$

The minimal repairs fix the cardinality constraints by adding a property atom *reviews*$(new_1, Pub1)$ with a fresh node new_1 and by adding a property atom *reviews* with a fresh node new_2 to either $Rev1$, $Rev2$ or new_1. The result is a data graph with three end nodes for *supports*$^-$ and three end nodes for *supports*$^-$ $\cdot$ *worksFor*, which validates the shape assignment for ReviewedPublicationShape.

▶ **Example 16.** Consider the following example. We introduce a new CourseLimitShape to represent a limitation of the number of participants for a course. Also, we want to keep the registered participants in sync with the IDs of the registered students.

$$\begin{aligned}
G &= \{\mathit{enrolledIn}(Ann, Course1), \mathit{enrolledIn}(Ben, Course1), \mathit{enrolledIn}(Bob, Course1),\\
&\quad \mathit{hasStudentID}(Ann, 2119110), \mathit{hasStudentID}(Ben, 1716110), \mathit{hasStudentID}(Bob, 9427084),\\
&\quad \mathit{participantID}(Course1, 2119110)\}\\
T &= \{\mathsf{CourseLimitShape}(Course1)\}\\
C &= \{\mathsf{CourseLimitShape} \leftarrow \leq_2 \mathit{enrolledIn}^- \cdot \mathit{hasStudentID}\\
&\quad \wedge\ \mathit{enrolledIn}^- \cdot \mathit{hasStudentID} = \mathit{participantID}\}
\end{aligned}$$

The CourseLimitShape, which indicates that a course can only have a maximum of two participants, is assigned to course $Course1$. The constraint is checked by means of the (sequence) path *enrolledIn*$^-$ $\cdot$ *hasStudentID*, which uses an inverse property *enrolledIn*$^-$ to navigate backwards from a course to a student. Also, a course gets the student IDs of the participants noted via the property *participantID* which has to be equal to the student IDs of students enrolled in the course via the path $enrolledIn^- \cdot hasStudentID$.

We assign CourseLimitShape to $Course1$. However, it does not validate, because there are too many students enrolled in $Course1$ (Ann, Ben and Bob). Also the equality constraint on *participantID* is not satisfied for student ID nodes 1716110 and 9427084. To repair the shape assignment, the repair program picks one $p^{sE}\_$ atom (3 existing $p^{sE}\_$ atoms minus a maximum cardinality of 2) to be made false via rule (16), and as a further consequence picks from atoms on the path $enrolledIn^- \cdot hasStudentID$ to be made false via rule (17), thereby cutting the path and satisfying the cardinality constraint of $\leq_2$. Also, the remaining two paths need to satisfy the equality on *participantID* via rules (18) and (19). We get the following minimal repairs.

$$\begin{aligned}
A_1 &= \{participantID(Course1, 9427084)\}, & D_1 &= \{enrolledIn(Ben, Course1)\}\\
A_2 &= \{participantID(Course1, 9427084)\}, & D_2 &= \{hasStudentID(Ben, 1716110)\}\\
A_3 &= \{participantID(Course1, 1716110)\}, & D_3 &= \{enrolledIn(Bob, Course1)\}\\
A_4 &= \{participantID(Course1, 1716110)\}, & D_4 &= \{hasStudentID(Bob, 9427084)\}
\end{aligned}$$

The minimal repairs fix the cardinality constraint by either removing Ben or removing Bob from $Course1$ by cutting the path via false advise on either *enrolledIn* or *hasStudentID*, while at the same time adding the *participantID* of the remaining node (9427084 or 1716110) via true advise. The result is a data graph with two $Course1$ participants with their student IDs also noted via *participantID* for $Course1$, which validates the shape assignment for CourseLimitShape.

7 ASP Implementation

We developed a prototypical system for implementing SHACL repair programs using the Java programming language and the ASP system clingo.[6] The prototype parses a SHACL shapes graph and a data graph, both in Turtle syntax, and translates them into a repair program as a set of clingo rules and facts. The repair program can then be executed using clingo, which returns the stable models with (sub)sets of repaired shape target nodes and sets of addition triples and deletion triples as repairs for the data graph. The implementation is available online on github.[7]

Generating SHACL repair programs

In the following, we explain the process of running SHACL repairs in detail. Figure 1 shows the processing flow through the individual processing steps.

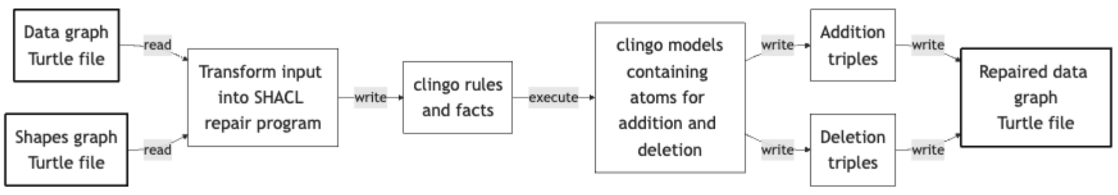


Figure 1 SHACL repair process.

[6] https://potassco.org/clingo/
[7] https://github.com/robert-david/shacl-repairs

- First, input data, which are Turtle files containing the data graph and the shapes graph, are read by the Java program.
- The Java program processes the input and translates it into a SHACL repair program consisting of clingo rules and facts.
- clingo then runs the repair program and returns the answer sets (stable models) containing atoms for additions and deletions.
- The Java program reads the answer sets and writes addition and deletion triples in Turtle syntax.
- The Java program uses the addition and deletion triples to change the input data graph and create a repaired data graph, which is then written into a file in Turtle syntax.

Translating SHACL shape constraints from RDF syntax to abstract syntax

For implementing the translation, we first need to translate from SHACL shapes in RDF Turtle syntax into our abstract syntax, which is then further translated into ASP to generate the repair program. In the following, we show the translation of SHACL shape constraints from Turtle syntax into our abstract syntax. For some of the constraints, like class membership using *sh:class*, there is a simple translation into an equivalent shape expression in abstract syntax. For other constraints, like minimum cardinality constraint *sh:minCount*, we need to translate into a complex shape expression.

Table 1 shows the list of translations from SHACL constraint components in RDF syntax defined in [32] to complex shape expressions in abstract syntax.

▶ **Example 17.** We present an example to illustrate how this translation is done for a SHACL shape consisting of several constraints. The shape *:StudentShape* below is translated into a shape expression as described in Section 2.

```
:StudentShape a sh:NodeShape;
    sh:targetNode :Ben;
    sh:property [
        sh:path :enrolledIn;
        sh:minCount 1;
        sh:maxCount 1;
        sh:or (
            [ sh:class :Course ]
            [ sh:in ("ID1" "ID2") ]
        )
    ] .
```

The translation into a shape expression is as follows. We introduce new shape names for conjunctions (*sh:and*) and disjunctions (*sh:or*) for better readability. Note that SHACL constraints on the same node or property are interpreted as a conjunction as stated in [32].

$$: StudentShape \leftarrow s_1 \land s_2$$

$$s_1 \leftarrow \geq_1 : enrolledIn.s_3 \land \neg \geq_1 : enrolledIn.\neg s_3$$

$$s_2 \leftarrow \neg \geq_2 : enrolledIn.s_4 \land \neg \geq_1 : enrolledIn.\neg s_3$$

$$s_3 \leftarrow s_5 \lor s_6$$

$$s_5 \leftarrow : Course$$

$$s_6 \leftarrow \neg(\neg "ID1" \land \neg "ID2") \land \neg(\neg "ID1" \land \lor "ID2")$$

This shape expression of *StudentShape* can be translated into normal form and consequently into ASP rules as explained in Section 5.

Table 1 SHACL Constraints Translation.

SHACL constraint component	Example SHACL constraint in RDF	Complex shape expression in abstract syntax
Cardinality constraint components		
sh:minCount	[sh:path E ; sh:minCount 2]	$s \leftarrow \geq_2 E.\top$
sh:maxCount	[sh:path E ; sh:maxCount 2]	$s \leftarrow \neg \geq_3 E.\top$
Logical constraint components		
sh:and	sh:and (:A :B)	$s \leftarrow A \wedge B$
sh:or	sh:or (:A :B)	$s \leftarrow A \vee B$
sh:not	sh:not :A	$s \leftarrow \neg A$
sh:xone	sh:xone (:A :B)	$s \leftarrow (A \wedge \neg B) \vee (\neg A \wedge B)$
Shape-based constraint components		
sh:qualifiedMinCount	[sh:path E ; sh:qualifiedMinCount 2 ; sh:qualifiedValueShape :A]	$s \leftarrow \geq_2 E.A$
sh:qualifiedMaxCount	[sh:path E ; sh:qualifiedMaxCount 2 ; sh:qualifiedValueShape :A]	$s \leftarrow \neg \geq_3 E.A$
Other constraint components		
sh:hasValue	[sh:path E ; sh:hasValue :c]	$s \leftarrow \exists E.c$
sh:in	[sh:path E ; sh:in (:c1 :c2)]	$s \leftarrow \exists E.(c_1 \vee c_2)$

8 Tests for SHACL repair programs

To validate our implementation of the SHACL repair program, we decided to test against the official SHACL data shapes test suite, *SHACL Test Suite and Implementation Report*.[8] We then consider a real-world scenario over Wikidata as a data graph and test the performance against SHACL shapes with Wikidata constraints [26].

The test suite is intended for SHACL processors to test against the test cases and identify how far they correctly cover the SHACL W3C recommendation with their implementation. It also provides a list of published test results from implementations that have submitted their test results. The tests are maintained by members of the Working Group. In our view, this is the best candidate to test the SHACL repair program, because the test suite is officially approved and maintained by the SHACL authors, it provides a sufficient coverage regarding the SHACL W3C recommendation for testing validation and it is publicly available. Besides the SHACL data shapes test suite, we also developed unit tests as part of our implementation, which are intended to provide a sufficient coverage for the implemented SHACL repairs.

For testing performance in a real-world scenario, we use sample data from the Wikidata KG and SHACL shapes intended to check Wikidata constraints. We select a fixed set of shapes and scale the sample size to get insights about the performance of our approach with respect to the size of the data graph.

In the following, we first describe the unit tests that were defined as part of the implementation. Then, we go into the details of the SHACL data shapes test suite and describe which test cases were repaired and which gaps are still there regarding our implementation. We then present our test set of SHACL shapes for Wikidata constraints, describe how we create the data graph samples and discuss the performance test results.

Unit Test Suite

To verify the repair program implementation, we created a unit test suite that covers all the implemented shape constraints. The idea is to have minimal independent examples that the repair implementation can test against. We grouped these examples in seven groups for class constraints, property constraints, property path constraints, equality on property paths, value (constant) constraints, logical constraints and constraints that will cause a conflict either within a shape or between multiple shape assignments. The seven groups also include logical expressions for conjunction, disjunction and negation, while the (dedicated) logical constraints tests cover the *sh:xone* constraint component. The unit test suite consists of a total of 91 test cases.

Data Shapes Test Suite

The main target of our tests is the SHACL data shapes test suite. We looked into the test suite to identify a maximal subset of test cases that is covered by our current implementation. We identified a subset of 33 test cases from the 121 test cases in the test suite. We ran the SHACL repair program against the selected test cases and validated the results by comparing the resulting repairs with the validation results provided as part of the test cases. Although the validation results do not directly show the repair options (as described in [3]), they provide insights to determine if the resulting repairs are viable. In the following, we provide the details of the repair test results for the selected 33 test cases of the SHACL data shapes test suite. We group them into 3 groups, which are *node*, *path* and *property*, as named to indicate the purpose in the SHACL

[8] `https://w3c.github.io/data-shapes/data-shapes-test-suite/`

Table 2 SHACL Unit Tests.

Unit Test Suite	
Constraint Test Group	Nr of Tests
Class	8
Property	9
Property path	35
Equals	6
Value (constant)	19
Logical (sh:xone)	4
Conflict	10

data shapes test suite. All test cases were repaired successfully, if a repair is possible. For 30 test cases, the repairs resulted in a data graph that does not cause any violations on SHACL validation. For 3 test cases there are skipped targets and a full repair of the data graph is not possible regarding cardinality-minimal repairs as defined in this paper. The test scenarios showed that our implementation works according to the defined SHACL repair program to repair the test cases.

Table 3 SHACL Test Results – Node constraints.

SHACL data shapes test suite – node			
Test Name	Nr of Repair Models	Ok/repaired/skip Targets	Comments
and-001	1	1/2/0	sh:and
and-002	2	1/2/0	sh:and
class-001	1	2/2/0	sh:class
class-002	1	2/2/0	sh:class
class-003	1	2/4/0	sh:class with multiple classes, overlapping target sets
hasValue-001	1	1/0/1	sh:hasValue
in-001	1	3/0/1	sh:in
node-001	1	1/1/0	sh:node
not-001	1	1/1/0	sh:not
not-002	1	1/1/0	sh:not
or-001	2	2/3/0	sh:or
xone-001	3	2/1/0	sh:xone
xone-duplicate	1	0/0/2	validation report for shape xone-duplicate

Table 4 SHACL Test Results – Path constraints.

SHACL data shapes test suite – path			
Test Name	Nr of Repair Models	Ok/repaired/skip Targets	Comments
path-complex-002	1	0/2/0	path sequence and sh:inversePath
path-inverse-001	3	1/2/0	sh:inversePath
path-sequence-001	1	2/2/0	path sequence
path-sequence-002	1	2/2/0	path sequence
path-strange-001	3	2/1/0	two valid paths together
path-strange-002	3	2/1/0	valid and invalid paths together

Table 5 SHACL Test Results – Property constraints.

SHACL data shapes test suite – property			
Test Name	Nr of Repair Models	Ok/repaired/skip Targets	Comments
and-001	5	1/3/0	sh:and
class-001	4	2/1/0	sh:class
equals-001	1	2/4/0	sh:equals
hasValue-001	1	2/1/0	sh:hasValue
in-001	1	2/1/0	sh:in
maxCount-001	2	1/1/0	sh:maxCount
maxCount-002	1	1/1/0	sh:maxCount
minCount-001	1	1/1/0	sh:minCount
minCount-002	1	1/0/0	sh:minCount
node-001	1	3/1/0	sh:node
node-002	2	1/1/0	sh:node
not-001	2	2/1/0	sh:not
or-001	3	2/1/0	sh:or
qualifiedValueShape-001	1	0/1/0	sh:qualifiedValueShape

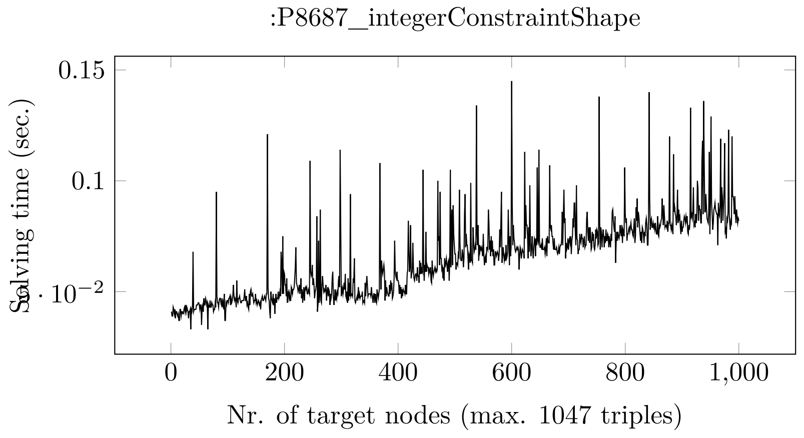
Table 6 SHACL Unit Tests.

Wikidata SHACL Shapes		
Nr	Shape Name	Constraint Components and Paths
1	P8687_integerConstraintShape	sh:datatype
2	:P26_SymmetricShape	sh:inversePath, sh:equals, sh:minCount
3	:P1083_NoBoundsShape	sequence path, sh:and, sh:maxCount
4	:P1469_ItemRequiresStatementShape	sh:in, sh:qualifiedValueShape, sh:qualifiedMinCount
5	:P1283_ValueRequiresStatementShapes	sh:or, sequence path, sh:in, sh:qualifiedValueShape, sh:qualifiedMinCount

Wikidata Performance Tests

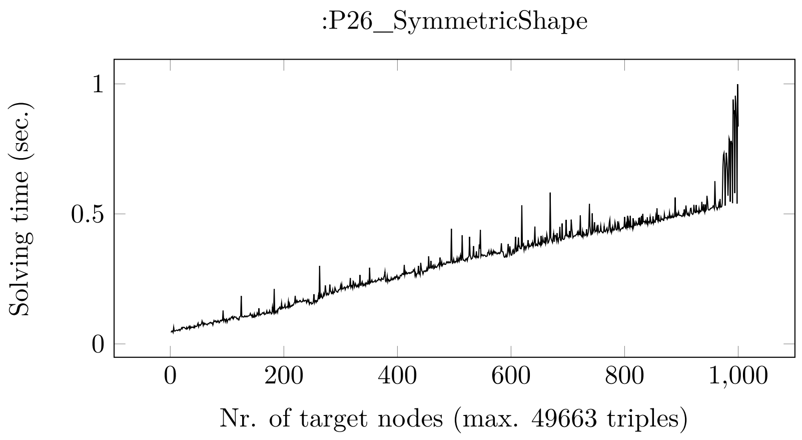
Wikidata [39] is a collaborative, free and open knowledge base providing structured data from Wikimedia projects, like Wikipedia.[9] Wikidata provides constraint types to validate data, which are defined by the community to ensure data quality for collaborative editing. However, Wikidata constraints are not enforced and violations in the data can be found.

Recent work on Wikidata constraint checking by Ferranti et al. [26] provided publicly available SHACL shapes,[10] which formalize Wikidata constraints to be validated by SHACL processors. For testing the performance of SHACL repair programs, we selected 5 representative Wikidata shapes to be used to repair Wikidata RDF graph samples. The shapes (1 to 5) have an increasing number of constraints and cover the supported shape expressions. Table 6 shows the details of the selected shapes and constraints. We then retrieved RDF graph samples from Wikidata via SPARQL endpoint, where we specifically retrieved seven data sets, which cause violations regarding the seven shapes. Performance tests scale up the number of shape target nodes from 1 to a maximum of 1000. We also limited the processing time of running the repair programs in clingo to 100 minutes of time. The Wikidata performance diagrams (below) show the time to compute repairs in relation to the number of target nodes. We also note the maximum size of the (sample) data graph, which is the number of triples for the maximum number of target nodes tested.

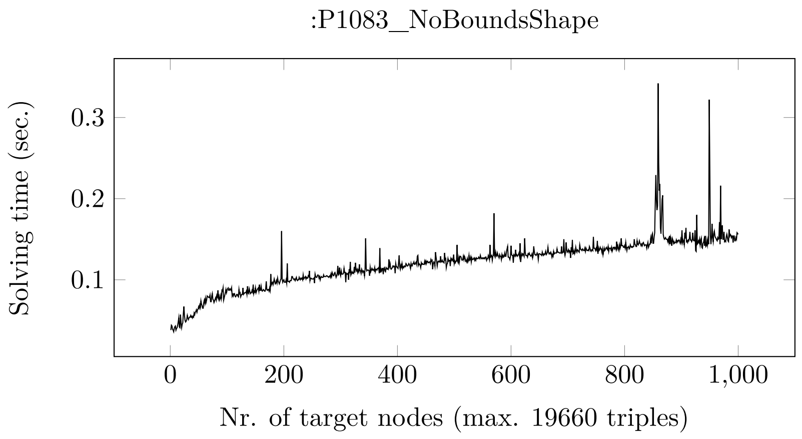


[9] https://www.wikidata.org/

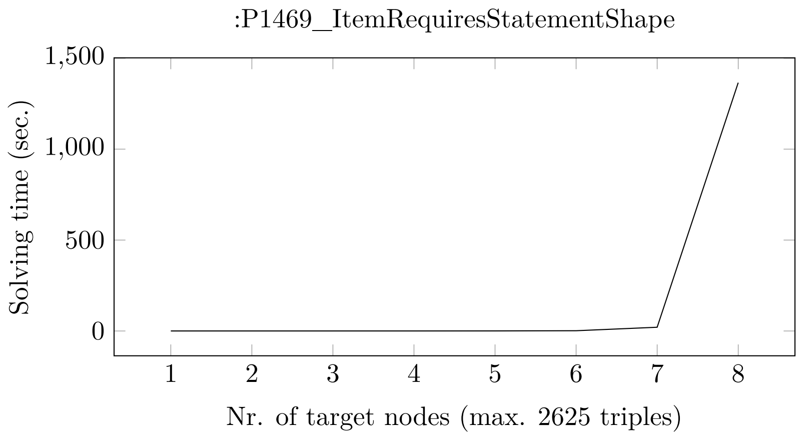
[10] https://github.com/nicolasferranti/wikidata-constraints-formalization/tree/main/constraints-formalization



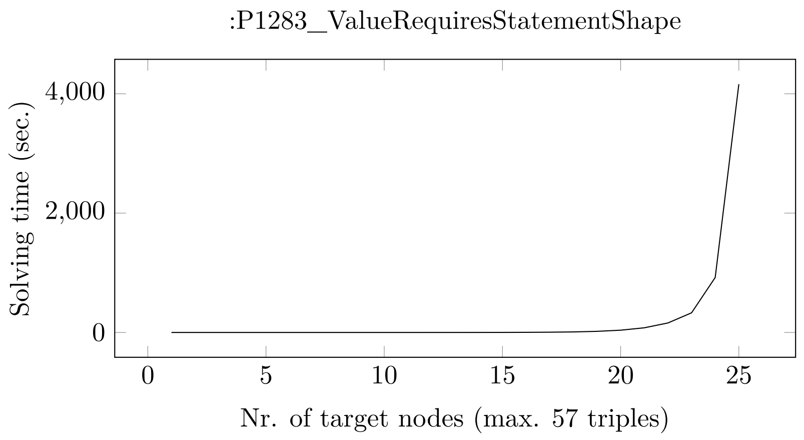
:P26_SymmetricShape
Solving time (sec.)
1
0.5
0
0
200
400
600
800
1,000
Nr. of target nodes (max. 49663 triples)



:P1083_NoBoundsShape
Solving time (sec.)
0.3
0.2
0.1
0
200
400
600
800
1,000
Nr. of target nodes (max. 19660 triples)



:P1469_ItemRequiresStatementShape
Solving time (sec.)
1,500
1,000
500
0
1
2
3
4
5
6
7
8
Nr. of target nodes (max. 2625 triples)



For the tests of shapes 1,2 and 3, we can see a linear behavior when scaling the number of target nodes. Generally, the performance scales well and we are able to solve the maximum tested target nodes within hundreds of milliseconds. However, the tests for shapes 4 and 5 show an above linear behavior and we run into performance problems already with a low number of target nodes (9 for shape 4 and 25 for shape 5). Closer evaluation showed that the reason for this behavior is the use of constant constraints (using the *sh:in* constraint) and the constant reuse optimization.

Generally, with the exception of constants constraints, our approach shows good scalability regarding the size of the data graph. Further tests with large scale data graphs are required to gain further insights into the limitations.

9 Conclusion

We presented an approach to minimally repairing a data graph so that it conforms to a set of SHACL constraints. We first analyze the types of repair that may be desirable in practice. Specifically, we observe that to repair cardinality constraints, it may not be desirable to reuse existing nodes from the graph, as this could introduce incorrect facts into the data. Therefore, we consider two scenarios: one where repairs always introduce fresh values to satisfy cardinality constraints, and another where repairs may reuse existing nodes if necessary, with a preference for using fresh nodes.

Inspired by existing work in databases, we propose Answer Set Programming encodings of the repair problem that account for these different repair scenarios. We first describe an encoding for a restricted setting, which forces the program to introduce fresh values to satisfy existential constraints. We then extend this to a setting that handles arbitrary cardinality, allowing for the reuse of existing constants when necessary. Additionally, we provide an encoding to support the repair of property path constraints and equality constraints.

In case not all the shape targets can be repaired, we optimize to repair as many of them as possible. We developed a prototypical system for implementing SHACL repair programs using the Java and the ASP system clingo. With the repair program and the ASP implementation, we lay the foundation for bringing repairs into practical scenarios, and thus improving the quality of RDF graphs in practice. We tested the implementation with unit tests, the data shapes test suite and performance tests on Wikidata as a real-world data set, which show promising results, but also some limitations, regarding the applicability in practice.

Future work. Several tasks remain for future work. For the practical side, the next step will be to select further use cases where we can apply the repairs and evaluate the practical feasibility and explore repair quality and scalability. Notably, our SHACL repair approach has already been applied in practice to resolve ambiguities in ontology RDF graphs [24]. An important next step is to also add support for repair preferences. The notion of minimal repairs considered in this paper does not take into account the possible user preferences regarding additions or deletions of concrete facts. E.g., a user might prioritize deletions of facts that are older or are coming from less reliable sources. Such setting has led, e.g., to the notions of global, Pareto and completion optimality of repairs [38]. Adapting these and other notions to SHACL is an important task for future work. This will help with decision making for users to select a repair and thereby improve the usability when applying repairs in practice.

Another important direction is to support *recursive* SHACL constraints, and hence, also *unrestricted* class-based and property-based targets. This is challenging because recursion combined with the introduction of fresh nodes may cause non-termination of the repair process, i.e., an infinite repair might be forced. However, it might be important to not rule out relevant practical use cases.

References

1 Shqiponja Ahmetaj, Iovka Boneva, Jan Hidders, Katja Hose, Maxime Jakubowski, José Emilio Labra Gayo, Wim Martens, F. Mogavero, Filip Murlak, Cem Okulmus, Axel Polleres, Ognjen Savkovic, Mantas Simkus, and Dominik Tomaszuk. Common foundations for shacl, shex, and pg-schema. *The Web Conference (WWW)*, 2025. To appear. doi:10.48550/arXiv.2502.01295.

2 Shqiponja Ahmetaj, Diego Calvanese, Magdalena Ortiz, and Mantas Šimkus. Managing change in graph-structured data using description logics. In *AAAI-14*. AAAI Press, 2014. doi:10.48550/arXiv.1404.4274.

3 Shqiponja Ahmetaj, Robert David, Magdalena Ortiz, Axel Polleres, Bojken Shehu, and Mantas Simkus. Reasoning about explanations for non-validation in SHACL. In Meghyn Bienvenu, Gerhard Lakemeyer, and Esra Erdem, editors, *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning, KR 2021, Online event, November 3-12, 2021*, pages 12–21, 2021. doi:10.24963/kr.2021/2.

4 Shqiponja Ahmetaj, Robert David, Axel Polleres, and Mantas Simkus. Repairing SHACL constraint violations using answer set programming. In *The Semantic Web - ISWC 2022 - 21st International Semantic Web Conference, Virtual Event, October 23-27, 2022, Proceedings*, volume 13489 of *Lecture Notes in Computer Science*, pages 375–391. Springer, 2022. doi:10.1007/978-3-031-19433-7_22.

5 Shqiponja Ahmetaj, Timo Camillo Merkl, and Reinhard Pichler. Consistent query answering over SHACL constraints. In Pierre Marquis, Magdalena Ortiz, and Maurice Pagnucco, editors, *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning, KR 2024, Hanoi, Vietnam. November 2-8, 2024*, 2024. doi:10.24963/KR.2024/1.

6 Medina Andresel, Julien Corman, Magdalena Ortiz, Juan L. Reutter, Ognjen Savkovic, and Mantas Šimkus. Stable model semantics for recursive SHACL. In *Proc. of The Web Conference 2020*, WWW '20, pages 1570–1580. ACM, 2020. doi:10.1145/3366423.3380229.

7 Marcelo Arenas, Leopoldo E. Bertossi, and Jan Chomicki. Consistent query answers in inconsistent databases. In *Proc. of PODS*, pages 68–79. ACM Press, 1999. doi:10.1145/303976.303983.

8 Franz Baader. Optimal Repairs in Ontology Engineering as Pseudo-Contractions in Belief Change. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, SAC '23, pages 983–990, New York, NY, USA, 2023. Association for Computing Machinery. doi:10.1145/3555776.3577719.

9 Franz Baader. Relating Optimal Repairs in Ontology Engineering with Contraction Operations in Belief Change. *SIGAPP Appl. Comput. Rev.*, 23(3):5–18, September 2023. doi:10.1145/3626307.3626308.

10 Franz Baader, Patrick Koopmann, Francesco Kriegel, and Adrian Nuradiansyah. Computing Optimal Repairs of Quantified ABoxes w.r.t. Static EL TBoxes. In *Automated Deduction – CADE 28: 28th International Conference on Automated Deduction, Virtual Event, July 12–15, 2021, Proceedings*, pages 309–326, Berlin, Heidelberg, 2021. Springer-Verlag. doi:10.1007/978-3-030-79876-5_18.

11 Franz Baader, Patrick Koopmann, Francesco Kriegel, and Adrian Nuradiansyah. Optimal ABox Repair w.r.t. Static EL TBoxes: From Quantified ABoxes Back To ABoxes. In *The Semantic Web: 19th International Conference, ESWC 2022, Hersonissos, Crete, Greece, May 29 – June 2, 2022, Proceedings*, pages 130–146, Berlin, Heidelberg, 2022. Springer-Verlag. doi:10.1007/978-3-031-06981-9_8.

12 Franz Baader and Francesco Kriegel. Pushing Optimal ABox Repair from EL Towards More Expressive Horn-DLs. In *Proceedings of the 19th*

International Conference on Principles of Knowledge Representation and Reasoning, pages 22–32, August 2022. doi:10.24963/kr.2022/3.

13 Franz Baader and Renata Wassermann. Contractions Based on Optimal Repairs. *LTCS-Report 24-03*, 2024.

14 Leopoldo E. Bertossi. *Database Repairing and Consistent Query Answering*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011. doi:10.2200/S00379ED1V01Y201108DTM020.

15 Meghyn Bienvenu, Camille Bourgaux, and François Goasdoué. Querying inconsistent description logic knowledge bases under preferred repair semantics. In *AAAI*. AAAI Press, 2014. doi:10.1609/aaai.v28i1.8855.

16 Meghyn Bienvenu, Camille Bourgaux, and François Goasdoué. Explaining inconsistency-tolerant query answering over description logic knowledge bases. In *AAAI*. AAAI Press, 2016. doi:10.1609/aaai.v30i1.10092.

17 Meghyn Bienvenu and Riccardo Rosati. Tractable approximations of consistent query answering for robust ontology-based data access. In Francesca Rossi, editor, *IJCAI*. IJCAI/AAAI, 2013. URL: http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6904.

18 Diego Calvanese, Davide Lanti, Ana Ozaki, Rafael Peñaloza, and Guohui Xiao. Enriching ontology-based data access with provenance. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI 2019)*. AAAI Press, 2019.

19 Diego Calvanese, Magdalena Ortiz, and Mantas Šimkus. Verification of Evolving Graph-structured Data under Expressive Path Constraints. In *ICDT*, 2016. doi:10.4230/LIPIcs.ICDT.2016.15.

20 Diego Calvanese, Magdalena Ortiz, Mantas Simkus, and Giorgio Stefanoni. Reasoning about explanations for negative query answers in DL-Lite. *J. Artif. Intell. Res.*, 48:635–669, 2013. doi:10.1613/jair.3870.

21 İsmail İlkan Ceylan, Thomas Lukasiewicz, Enrico Malizia, Cristian Molinaro, and Andrius Vaicenavicius. Explanations for negative query answers under existential rules. In *Proc. of KR 2020*, pages 223–232, 2020. doi:10.24963/kr.2020/23.

22 Julien Corman, Juan L. Reutter, and Ognjen Savkovic. Semantics and validation of recursive SHACL. In *Proc. of ISWC'18*. Springer, 2018. doi:10.1007/978-3-030-00671-6_19.

23 Robert David. SHACL Repair Program Implementation. Software, version 1.2.1., swhId: swh:1:dir:9de47c8b97cba079add751f9e2a644212 38a67f3 (visited on 2025-12-08). URL: https://github.com/robert-david/shacl-repairs, doi:10.4230/artifacts.25261.

24 Robert David, Albin Ahmeti, Shqiponja Ahmetaj, and Axel Polleres. OWLstrict: A Constrained OWL Fragment to avoid Ambiguities for Knowledge Graph Practitioners. In *The Semantic Web: 22nd European Semantic Web Conference, ESWC 2025, Portoroz, Slovenia, June 1–5, 2025, Proceedings, Part II*, pages 47–64, Berlin, Heidelberg, 2025. Springer-Verlag. doi:10.1007/978-3-031-94578-6_3.

25 Thomas Eiter, Giovambattista Ianni, and Thomas Krennwallner. Answer set programming: A primer. In Sergio Tessaris, Enrico Franconi, Thomas Eiter, Claudio Gutiérrez, Siegfried Handschuh, Marie-Christine Rousset, and Renate A. Schmidt, editors, *Reasoning Web. Semantic Technologies for Information Systems*, volume 5689 of *Lecture Notes in Computer Science*, pages 40–110. Springer, 2009. doi:10.1007/978-3-642-03754-2_2.

26 Nicolas Ferranti, Axel Polleres, Jairo Francisco de Souza, and Shqiponja Ahmetaj. Formalizing property constraints in wikidata. In *Wikidata@ISWC*, 2022.

27 Mikhail Galkin, Sören Auer, Maria-Esther Vidal, and Simon Scerri. Enterprise knowledge graphs: A semantic approach for knowledge management in the next generation of enterprise information systems. In *ICEIS (2)*, pages 88–98, 2017. doi:10.5220/0006325200880098.

28 José Emilio Labra Gayo, Eric Prud'hommeaux, Iovka Boneva, and Dimitris Kontokostas. *Validating RDF Data*. Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool Publishers, 2017. doi:10.2200/S00786ED1V01Y201707WBE016.

29 Maertin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub. Multi-shot asp solving with clingo. *Theory and Practice of Logic Programming*, 19(1):27–82, 2019. doi:10.1017/S1471068418000054.

30 Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutiérrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. *Knowledge Graphs*. Synthesis Lectures on Data, Semantics, and Knowledge. Morgan & Claypool Publishers, 2021. doi:10.2200/S01125ED1V01Y202109DSK022.

31 Aditya Kalyanpur, Bijan Parsia, Matthew Horridge, and Evren Sirin. Finding all justifications of OWL DL entailments. In *The Semantic Web: 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007+ASWC 2007, Busan, Korea, November 11-15, 2007. Proceedings*, pages 267–280. Springer, 2007. doi:10.1007/978-3-540-76298-0_20.

32 Holger Knublauch and Dimitris Kontokostas. Shapes constraint language (SHACL). Technical report, W3C, July 2017. URL: https://www.w3.org/TR/shacl/.

33 Domenico Lembo, Maurizio Lenzerini, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. Inconsistency-tolerant semantics for description logics. In *RR*, Lecture Notes in Computer Science. Springer, 2010. doi:10.1007/978-3-642-15918-3_9.

34 Ying Li and Patrick Lambrix. Repairing $\mathcal{EL}$ ontologies using weakening and completing. In *European Semantic Web Conference*, pages 298–315. Springer, 2023. doi:10.1007/978-3-031-33455-9_18.

35 Mónica Caniupán Marileo and Leopoldo E. Bertossi. The consistency extractor system: Answer

set programs for consistent query answering in databases. *Data Knowl. Eng.*, 69:545–572, 2010. doi:10.1016/j.datak.2010.01.005.

36 A. Ozaki and Rafael Peñaloza. Provenance in ontology-based data access. *Description Logics*, 2018. URL: https://api.semanticscholar.org/CorpusID:53179150.

37 Rafael Peñaloza. Axiom pinpointing. In *Applications and Practices in Ontology Design, Extraction, and Reasoning*, pages 162–177. IOS Press, 2020. doi:10.3233/SSW200042.

38 Slawek Staworko, Jan Chomicki, and Jerzy Marcinkowski. Prioritized repairing and consistent query answering in relational databases. *Ann. Math. Artif. Intell.*, 64(2-3):209–246, 2012. doi:10.1007/s10472-012-9288-8.

39 Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014. doi:10.1145/2629489.

Automating Invoice Validation with Knowledge Graphs: Optimizations and Practical Lessons

Johannes Mäkelburg
Technical University of Munich, Germany

Maribel Acosta
Technical University of Munich, Germany

Abstract

To increase the efficiency of creating, distributing, and processing of invoices, invoicing is handled in the form of Electronic Data Interchange (EDI). With EDI, invoices are handled in a standardized electronic or digital format rather than on paper. While EDIFACT is widely used for electronic invoicing, there is no standardized approach for validating its content. In this work, we tackle the problem of automatically validating electronic invoices in the EDIFACT format by leveraging KG technologies. We build on a previously developed pipeline that transforms EDIFACT invoices into RDF knowledge graphs (KGs). The resulting graphs are validated using SHACL constraints defined in collaboration with domain experts. In this work, we improve the pipeline by enhancing the correctness of the invoice representation, reducing validation time, and introducing error prioritization through the use of the severity predicate in SHACL. These improvements make validation results easier to interpret and significantly reduce the manual effort required. Our evaluation confirms that the approach is correct, efficient, and practical for real-world use.

2012 ACM Subject Classification Information systems → Electronic data interchange; Information systems → Resource Description Framework (RDF); Information systems → Web Ontology Language (OWL); Information systems → Ontologies

Keywords and phrases Electronic Invoice, Ontology, EDIFACT, RDF, RML, SHACL

Digital Object Identifier 10.4230/TGDK.3.3.2

Category Use Case

Supplementary Material *Software (Source Code)*: `https://github.com/DE-TUM/EDIFACT-VAL` [18]
archived at `swh:1:dir:d820a10d861cfd9e361208220cf7aad3b03ef1a6`

Funding *Johannes Mäkelburg*: Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SFB 1608 – 501798263.

Received 2025-06-05 **Accepted** 2025-10-24 **Published** 2025-12-10

Special Issue Use-Case Articles

1 Introduction

In the current business landscape, efficient financial and administrative practices are paramount for organizations. A key element in these operations is the invoicing process, which requires accuracy, timeliness, and standardization. To meet these demands, organizations are increasingly adopting Electronic Data Interchange (EDI), shifting from traditional paper-based invoices to structured electronic formats. Within EDI, EDIFACT is a widely adopted standard for representing electronic invoices, offering a uniform approach to their creation, distribution, and processing.

While EDIFACT has improved the efficiency of invoicing, a significant gap remains: the validation of EDIFACT invoices is not standardized. This affects business processes that depend on timely and accurate invoice data. One such case is the group purchasing organization Einkaufsbüro Deutscher Eisenhändler (E/D/E), which operates in 30 European countries with varying document regulations and serves as a key domain expert in the development and evaluation of our approach.

Although EDIFACT serves as the standard for handling invoices, the lack of a formal constraint language means that validation is often performed manually – a process that is both error-prone and labor-intensive. For example, consider an EDIFACT invoice where the invoice number is missing or the total amount does not match the sum of all line items. Such errors are hard to spot in the condensed EDIFACT syntax and can completely block further processing or lead to incorrect payments. With our KG-based approach, these inconsistencies are made explicit through SHACL validation: the system detects that a legally required identifier is absent or that the financial totals are inconsistent. This allows errors to be uncovered early and ensures that invoices meet both legal and business requirements before they enter downstream processes. To address this, we propose a knowledge graph (KG)-based solution for validating electronic invoices. The core of our approach is to model electronic invoices as RDF graphs. Shifting to the semantic web technologies allows for applying existing open standards and solutions for managing machine-readable data. At the core of the solution is the EDIFACT Ontology, which captures the semantics of EDIFACT message content in RDF. Second, we propose the tool EDIFACT-VAL to validate the content of the original EDIFACT messages. The tool processes the invoices in the EDIFACT format and translates them into XML. XML serves as an intermediate representation since the EDIFACT format is not compatible with existing KG tools. From the XML files, EDIFACT-VAL creates the RDF graphs using the EDIFACT Ontology and the RDF Mapping Language (RML) [4]. EDIFACT-VAL validates the invoice RDF graph using constraints defined in the Shapes Constraint Language (SHACL) [15]. The constraints are created with input from domain experts based on the EDIFACT guidelines.

This manuscript builds on our previous work EDIFACT-VAL V.1 [19] and introduces EDIFACT-VAL V.2, which includes the followoing novel contributions:

- A new invoice pre-processing approach that reduces integration effort and processing time.
- An error prioritization feature based on *sh:severity*, enabling users to systematically address the most critical validation issues first.
- Insights and lessons learned from real-world applications of EDIFACT-VAL.

We evaluate the extended approach using real-world EDIFACT invoices. The results demonstrate multiple improvements over the previous version of the EDIFACT-VAL. First, the generated RDF graphs offer a more accurate and semantically rich representation of invoice content, resulting in a more reliable and expert-validated foundation for downstream processing. Second, the runtime analysis shows that EDIFACT-VAL V.2 significantly outperforms EDIFACT-VAL V.1, primarily due to an optimized pre-processing and RDF mapping. The RDF graph generation phase alone achieves time savings of up to 66.56%, with the longest runtime observed being just 3.65 seconds. Third, the integration of error prioritization through *sh:severity* enables users to focus on the most critical validation issues, greatly simplifying the interpretation of validation results.

2 Preliminaries

First, we introduce the EDIFACT invoice concept, which forms the basis for the knowledge graphs. Then, we describe the purchase-to-pay ontology, which builds the basis for an EDIFACT ontology.

EDIFACT Invoice. EDIFACT is the most commonly used and most comprehensive international standard for electronic data interchange [13]. EDIFACT is used across almost all business sectors; the individual sectors are delimited in EDIFACT by so-called subsets. The maintenance of the standard is led by the United Nations and the Economic Commission for Europe.

Documents transmitted in EDIFACT are all types of messages of the business processes area. The structure of the messages is based on segments; these, in turn, consist of data elements and data element groups. These three components together are referred to as the EDIFACT elements.

Listing 1 Excerpt of an EDIFACT invoice.

```
1 UNH+1+INVOIC:D:96A:UN:EAN008'
2 BGM+380+4031541+43'
3 DTM+137:20220908:102'
4 NAD+IV+4317784000000::9++Einkaufsbuero DeutscherEisenhaendler:GmbH+EDE PLatz 1+Wuppertal++42389+DE'
5 LIN+1++4016671029277:EN::9'
6 PRI+AAA:16.78:::1:PCE'
7 MOA+79:100.68'
8 MOA+124:19.13'
9 UNT+37+1'
```

Listing 1 shows an excerpt of a real-world EDIFACT message. The segments are split into three sections: header-, detail- and summary section. In all three sections, some segments are required, meaning all three sections are always represented in an EDIFACT message. However, there are some segments within the sections that are optional. For example, the header section contains eight mandatory segments and six conditional segments. An example of a mandatory segment is the *NAD segment*, which provides information about one of the organizations involved in the invoice, including its role, name, unique identification number, and address. In contrast, the *PRI segment* is conditional. It specifies the unit price of an item, but this information is considered supplementary since the actual item price is already represented in the *MOA segment.* In the header, general information about the invoice is displayed, e.g., the invoice number (Line 2), the document date (Line 3), and information about the involved organizations (Line 4). Information about the sold items, including the net price (Line 6) or the article number (Line 5), is allocated in the detail section. The summary section contains the total amounts of the invoice, e.g., the total item amount (Line 7) or the total tax amount (Line 8). Above the header and below the summary section are segments allocated for the EDIFACT structure, e.g., the version and type of message (Line 1) or the end character (Line 9). The complexity of an EDIFACT invoice arises from several factors. A single message may include multiple invoices, each with its own header, detail, and summary. The number of segments varies depending on the level of detail. Invoices with many items or rich descriptions have far more segments. Finally, some segments bundle multiple pieces of information, such as the NAD segment (Line 4), which encodes role, name, address, and identifiers of an organization. These characteristics complicate validation, since it requires checking both the structural completeness of the message, such as the presence of all mandatory sections and segments, and the semantic consistency of the data, such as the correctness of totals or the proper assignment of organizational roles. In real-world settings, this structural variability directly impacts invoice handling, where issues have direct consequences: a missing identifier in the header may block legal acceptance, while mismatched totals can delay payment or introduce accounting errors.

P2P-O: Purchase to Pay Ontology. We use the Purchase-to-Pay Ontology (P2P-O) [27] as a foundation for modeling concepts from the EDIFACT standard. P2P-O is an OWL ontology that models semantic representation of invoices based on the *core invoice model* of the European Standard EN 16931-1:2017 [6]. P2P-O provides semantic classes for *item, price, documentline, organization, document, invoice* and *process*, making it a sound foundation for representing invoices.

3 Approach

Given an electronic invoice in the EDIFACT format, the problem tackled in this work is to validate the correctness of the invoice by representing the invoice as an RDF knowledge graph (KG). Our proposed solution comprises two parts: (1) an ontology to represent EDIFACT invoices (§ 3.1),

Table 1 Competency Questions for the EDIFACT Ontology.

Name	Competency Question	Ontology Class
CQ 0	What invoices are all listed in an EDIFACT message?	*E-Invoice*
CQ 1	Which organizations are involved in the invoice?	*Formal Organization*
CQ 2.1	What role does organization S play in the invoice?	*AgentRole*
CQ 2.2	Which organization is the buyer in the invoice?	*AgentRole*
CQ 3.1	What information is displayed about the involved organizations?	*AgentRole*
CQ 3.2	What is the address of the buyer?	*AgentRole*
CQ 4	What items are sold in the invoice?	*Item*
CQ 5.1	What information is displayed about the items sold?	*Item*
CQ 5.2	What is the net price of the items sold in the invoice?	*Item*
CQ 6.1	What are the invoice details of the invoice?	*Invoice-Details*
CQ 6.2	What is the invoice amount of the invoice?	*Invoice-Details*
CQ 6.3	What is the invoice number?	*Invoice-Details*
CQ 7	What information must be provided so that the file format is valid?	*EDIFACT-Structure*
CQ 8	To which business process can the invoice be assigned?	*E-Invoice*

Table 2 Overview of linked ontologies and vocabularies in the EDIFACT Ontology.

Prefix Name	Prefix	Frequency
agentRole	`https://archive.org/services/purl/domain/modular_ontology_design_library/agentrole#`	5
dc	`http://purl.org/dc/elements/1.1#`	2
frapo	`http://purl.org/cerif/frapo/`	3
schema	`http://schema.org/`	3
org	`http://www.w3.org/ns/org#`	1
p2p-o-doc-line	`https://purl.org/p2p-o/documentline#`	2
p2p-o-doc	`https://purl.org/p2p-o/document#`	1
p2p-o-inv	`https://purl.org/p2p-o/invoice#`	3
p2p-o-item	`https://purl.org/p2p-o/item#`	2
p2p-o-org	`https://purl.org/p2p-o/organization#`	4
vcard	`http://www.w3.org/2006/vcard/ns#`	2

based on the concepts presented in Sect. 2, and (2) the EDIFACT-VAL tool to perform the validation of invoices using KG technologies (§ 3.2). In this paper, we introduce a new version of EDIFACT-VAL, called EDIFACT-VAL V.2, which features an improved invoice pre-processing strategy and more tailored RML mappings. These enhancements increase the efficiency of the tool and enable a more specific validation of EDIFACT invoices.

3.1 EDIFACT Ontology

Development Process. The EDIFACT ontology was developed following best practices, like the NeOn method [31] or the formulation of ontology requirements and competency questions (Table 1). These steps were carried out in collaboration with Electronic Data Interchange experts from E/D/E. The development process is detailed in our previous work [19].

Reuse of Ontology Design Patterns and Existing Vocabularies and Ontologies. Following ontology design best practices [31, 23], we reuse existing resources to develop the EDIFACT-Ontology. An overview of the reused resources and the number of reuses can be found in Table 2. We apply the agent role pattern from the Modular Ontology Design Library (MODL) [28] for modelling the participation of organizations in invoices. This is necessary as the same organization

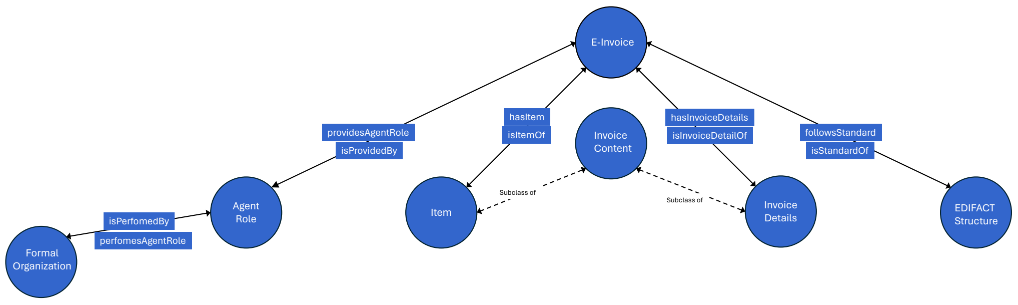


Figure 1 Main concepts of the EDIFACT ontology [19]. Image generated with WebVOWL [17].

can have many roles (seller, supplier, etc.) in the same or different invoices, to which different property values can be associated depending on its role. The ontologies listed in Table 2 also provide adequate solutions for our purpose. Most of them are reused in the class *AgentRole*; other examples are the country code from the Funding, Research Administration, and Projects Ontology [29] or the address of the vCard Ontology [21]. Also, four of the seven different main classes we defined in our EDIFACT Ontology are linked to concepts of these vocabularies or ontologies. For example, the class *FormalOrganization* is linked to the Core Organization Ontology (*org*), the *E-Invoice* to the P2P-O module *document*, and *Item* to the P2P-O module *item*.

Ontology Description. The EDIFACT Ontology is tailored to represent the concepts and fields of the EDIFACT standard. The proposed OWL ontology comprises 31 classes, 22 OWL object properties, 261 OWL data properties, and 6 annotation properties. The ontology was developed using WebProtégé [12]. Figure 1 illustrates the main concept of the EDIFACT ontologies by showing the connections between the different classes. The EDIFACT Ontology can be found under `https://purl.org/edifact/ontology`. Additionally, it has been integrated into the LOV catalogue, available at `https://lov.linkeddata.es/dataset/lov/vocabs/edifact-o`. The classes and some of their most relevant properties for the invoice KGs are discussed in detail in the following. We focus on properties that are essential for representing the structural and semantic aspects of invoices, while omitting less relevant or supplemental properties to maintain clarity and conciseness. The relevant competency questions for each class are provided in Table 1.

- *E-Invoice* This is the main class of the EDIFACT ontology, and its individuals or entities represent electronic invoices. This class is connected to other classes through object properties, including *EDIFACT-Structure* via the property *followsStandard*, *Item* via the property *hasItem*, *InvoiceDetails* via the property *hasInvoiceDetails*, and *AgentRoles* via the property *isProvidedBy* to capture the role of organizations in the invoice. The only data property of this class is *belongsToProcess*, denoting the business process of the invoice.
- *EDIFACT-Structure* This class contains all the information that ensures that the invoice file meets the requirements of the EDIFACT file format. This information appears at the beginning and end of a message and, therefore, does not belong to the content of the individual invoices. As a result, this class and the class *InvoiceContent* are disjoint, specified with the *owl:disjointWith* predicate. However, as this information is part of an invoice, this class is connected to the class *E-Invoice* via the object property *followsStandard*. The data properties of this class include *creationDate*, *dataExchangeCounter*, *messageReferenceNumber*, and *senderIndicator*. Among the datatype properties of the entities of this class, we have *creationDate*, *dataExchangeCounter*, *messageReferenceNumber*, and *senderIndicator*.

- *Invoice Details* This class contains all the information that can only occur in the header- and summary sections of the invoices. Exemplary datatype properties of this class are the document date (*hasDocumentDate*), the document number (*hasDocumentNumber*), the delivery condition (*deliveryCondition*), and the invoice amount (*hasInvoiceAmount*). The connection of the *InvoiceDetails* class to the *E-Invoice* class is done via the object property *hasInvoiceDetails*.
- *Item* This class allows for representing an item listed in the invoice, which is found in the detail section of the invoice. Examples of the datatype properties are the name of the item (*p2p-o-item:Item*), the net price of the item (*hasNetPriceOfItem*), the net weight of an item (*hasNetWeight*), the number assigned to product according to the International Article Numbering Association (*internationalArticleNumber*). Examples of the datatype properties for an item are the name of the item (*p2p-o-item:Item*), the net price of the item (*hasNetPriceOfItem*), the net weight of an item (*hasNetWeight*), the number assigned to a manufacturer's product according to the International Article Numbering Association (*internationalArticleNumber*), etc. We model this relationship as a multi-valued property by defining the object property *hasItem* (and its inverse *isItemOf*) between the *Item* and the *E-Invoice* classes.
- *InvoiceContent* This class allows for modelling the information that can be found in all three sections of the invoices. As a result, the information of the *InvoiceContent* class is defined as the union of the classes *Item* and *InvoiceDetails*, i.e., *InvoiceContent owl:unionOf (Item InvoiceDetails)*. Creating the *InvoiceContent* class ensures the consistency of the ontology by defining properties that apply to all parts of the invoice.
- *AgentRole* According to the guidelines, an organization can, or sometimes must, have many different roles. For example, in the warehousing business, one organization needs to have three roles: *Buyer Role*, *Invoicee Role*, and *Delivery Party Role*. We model these cases using the Role ontology pattern as defined by Shimizu et. al [28] and Grüninger & Fox [8]. In our ontology, the purpose of the class *AgentRole* is to represent the manifold roles an organization can have in an invoice. As a solution, several subclasses have been created, each representing one of these roles. The connection to the *E-Invoice* class exists through the object property *isProvidedBy* with the *E-Invoice* class in the range.
- *FormalOrganization* This class represents the organizations involved in the messages, addressing CQ 1. The properties assigned to the class *FormalOrganization* serve two purposes: assigning the role an organization plays in an invoice through *performsAgentRole*, and providing a globally unique identifier for organizations, i.e., *globalLocationNumber*. In the EDIFACT ontology, the only connection of the class *FormalOrganization* to another class is the class *AgentRole* via the object property *performsAgentRole* with the *AgentRole* class in the range.

3.2 The EDIFACT-VAL Tool

Now that we have modelled the terms of the EDIFACT standard in an OWL ontology (see Sect. 3.1), the EDIFACT-VAL tool processes the electronic invoices in the EDIFACT format to validate their content using KG technologies. Concretely, the tool carries out the following steps (cf. Figure 2). **(1) Invoice Pre-processing:** Translates the invoice files into XML files. **(2) RML Mapping:** Creates RDF knowledge graphs from the invoice XML files using generated RML mappings. **(3) RDF Validation:** Validates the invoice RDF graphs using constraints based on EDIFACT guidelines and reports from domain experts, which are formulated using the Shapes Constraints Language (SHACL) [15].

Compared to the old version from our previous work [19], called EDIFACT-VAL V.1, the new version, called EDIFACT-VAL V.2, features a different approach to Invoice Pre-Processing and RML Mapping. The new Invoice Pre-Processing now takes the original EDI messages as input and converts them into more tailored XML invoices. Based on these more refined XML invoices, the RML mapping can also be written in a more efficient and targeted way.

Listing 2 Representation of organizations using *AgentRole* and *FormalOrganization*.

```
@prefix invoice: <http://www.ede.com/edifact/invoice#>.
@prefix edifact-o: <https://purl.org/edifact/ontology#> .
@prefix agentRole: <https://archive.org/services/purl/domain/modular_ontology_design_library/agentrole#> .
@prefix frapo: <http://purl.org/cerif/frapo/> .
@prefix p2p-o-org: <https://purl.org/p2p-o/organization#> .
@prefix vcard: <http://www.w3.org/2006/vcard/ns#> .

invoice:r999995 a edifact-o:DeliveryPartyRole;
  frapo:hasCountryCode "DE";
  edifact-o:hasCity "Wuppertal";
  edifact-o:hasCountry "Deutschland";
  vcard:hasStreetAddress "In der Fleute 153";
  vcard:postalCode "42389";
  agentRole:isProvidedBy invoice:i999999;
  p2p-o-org:formalName "E/D/E-GmbH Anlieferstelle L205" .

invoice:r999996 a edifact-o:InvoiceeRole;
  frapo:hasCountryCode "DE";
  edifact-o:hasCity "Wuppertal";
  edifact-o:hasCountry "Deutschland";
  vcard:hasStreetAddress "EDE Platz 1";
  vcard:postalCode "42389";
  agentRole:isProvidedBy invoice:i999999;
  p2p-o-org:formalName "Einkaufsuero Deutscher Eisenhaendler GbmH".

invoice:4317784000000 a edifact-o:FormalOrganization;
  agentRole:performsAgentRole invoice:r999995, invoice:r999996 ;
  p2p-o-org:globalLocationNumber 4317784000000 .
```

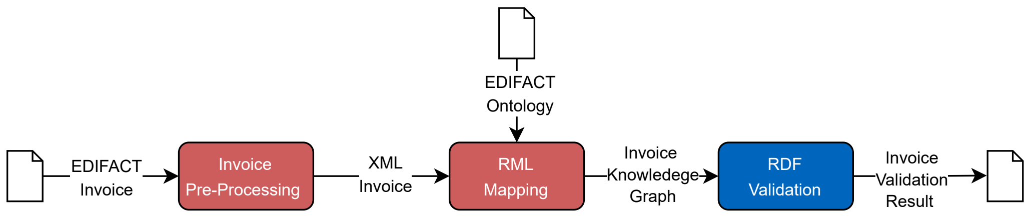


Figure 2 Overview of EDIFACT-VAL. Components in red have been updated from V1 to V2.

3.2.1 Invoice Pre-Processing

This step aims to prepare the EDIFACT message in a way that meets the requirements for the creation of an RDF graph. Despite that EDIFACT is an open standard, the file format is not yet widely compatible with existing tools, especially the ones related to knowledge graphs. A direct transformation into RDF would require custom parsers for each message variant, whereas XML provides a standardized and flexible basis for subsequent mappings. Consequently, a transformation into a compatible format is required. For both EDIFACT-VAL V.1 [19] and the updated EDIFACT-VAL V.2, we adopted XML as the intermediary format because it combines high flexibility with robust support from a broad ecosystem of processing tools. The updated EDIFACT-VAL V.2 introduces three key improvements over EDIFACT-VAL V.1: (i): direct handling of original EDIFACT invoices (no pre-existing XML needed), (ii): targeted creation of only required XML elements and attributes, and (iii): a simplified saving process that avoids redundant file versions. To demonstrate the differences between the two approach, we examine the handling of the NAD-Segment, shown in Line 4 of Listing 1. There the NAD-segment is displayed with the qualifier *IV*, representing information about the Invoicee of the invoice.

Listing 3 Result of Pre-Processing with EDIFACT-VAL V.1.

```
<Interchange>
    <Message>
        <G_Group_2 id="8" mid="InvoiceDetail1">
            <S_NAD agentRole="InvoiceeRole" details="Invoice1" id="9" mid="InvoiceDetail1" nid="Item0"
            organisation="Role2" parent="GGroup2">
                <D_3035 id="9" mid="InvoiceDetail1" nid="Item0" organisation="Role2" parent="NAD">IV</D_3035>
                <C_C082 id="9" mid="InvoiceDetail1" nid="Item0" organisation="Role2">
                    <D_3039 id="9" mid="InvoiceDetail1" nid="Item0" organisation="Role2">+43177840000</D_3039>
                    <D_3055>9</D_3055>
                </C_C082>
                <C_C080 id="9" mid="InvoiceDetail1" nid="Item0" organisation="Role2">
                    <D_3036 id="9" mid="InvoiceDetail1" nid="Item0" organisation="Role2">+Einkaufsbuero
                    DeutscherEisenhaendler:GmbH+</D_3036>
                </C_C080>
                <C_C059 id="9" mid="InvoiceDetail1" nid="Item0" organisation="Role2">
                    <D_3042>EDE PLatz 1</D_3042>
                </C_C059>
                <D_3164>Wuppertal</D_3164>
                <D_3251>42389</D_3251>
                <D_3207 id="9" mid="InvoiceDetail1" nid="Item0" organisation="Role2">DE</D_3207>
                <D_PaFu id="9" mid="InvoiceDetail1" nid="Item0" organisation="Role2">Invoicee</D_PaFu>
                <D_Country id="9" mid="InvoiceDetail1" nid="Item0" organisation="Role2">Deutschland</D_Country>
            </S_NAD>
        </G_Group_2>
        <Invoicee Item="Item1" Kaeufer="Role1" Lieferanschrift="Role3" Lieferant="Role4" Invoicee="Role2"
        details="Invoice1" id="1" mid="InvoiceDetail1">1234567000000</Invoicee>
    </Message>
</Interchange>
```

Listing 4 Result of Pre-Processing with EDIFACT-VAL V.2.

```
<Interchange>
    <Message>
        <Party agentRole="InvoiceeRole" details="Invoice1" organisation="IV1">
          <GLNNumber>+43177840000</GLNNumber>
          <CodeListResponsibleAgency>9</CodeListResponsibleAgency>
          <PartyName>Einkaufsbuero DeutscherEisenhaendler:GmbH</PartyName>
          <Street>EDE PLatz 1</Street>
          <PostalCode>Wuppertal</PostalCode>
          <CountrySubEntity>42389</CountrySubEntity>
          <SecondPostalCode>DE</SecondPostalCode>
        </Party>
    </Message>
</Interchange>
```

(i) Direct Handling of EDIFACT Invoices. A limitation of EDIFACT-VAL V.1 is its requirement for a pre-existing XML representation of the EDIFACT invoice. In contrast, EDIFACT-VAL V.2 directly accepts original EDIFACT invoices, eliminating this dependency and expanding its applicability.

(ii) Targeted XML Creation. EDIFACT messages use qualifiers to secure file compactness by packing several values into a single field. To handle these qualifiers, EDIFACT-VAL V.1 separated such segments into several XML elements, creating a one-to-many correspondence between EDIFACT elements and XML elements. As a result, the size of the XML files increased significantly due to the additional elements and attributes needed to represent this structure.

The key improvement of EDIFACT-VAL V.2 lies in its more targeted creation of XML files: only required XML elements and attributes are generated, reducing unnecessary overhead for further processing. Here, required XML elements are defined as non-empty elements that contain data of interest for the RDF graph transformation process. This includes elements that hold essential information from the EDIFACT message (e.g., invoice details, line items) and elements that aggregate multiple data elements from the EDIFACT structure into a single XML representation.

This focused generation not only reduces the overall number of elements but also leads to fewer nested structures, which in turn minimizes the need for additional attributes to map content to the classes defined in the EDIFACT Ontology (see Section 3.1). Compared to EDIFACT-VAL V.1, we change the numbering of the attributes into more identifiable names like *Item* plus an ascending number for the attribute used to identify the item. This is also the case with organizations, where we also incorporate the role and the invoices inside the message. By focusing only on these required XML elements, EDIFACT-VAL V.2 minimizes overhead and ensures that the generated XML files contain only relevant data for the graph creation process.

(iii) Simplified Saving Process. Another notable improvement of EDIFACT-VAL V.2 is that the XML file now only needs to be saved once before reaching the final version. In contrast, EDIFACT-VAL V.1 required saving the XML file two additional times, introducing further complexity and redundancy. Listing 3 and Listing 4 illustrate the XML-representations of a NAD-Segment with the qualifier *IV* for EDIFACT-VAL V.1 and EDIFACT-VAL V.2, respectively. The comparison demonstrates a significant reduction (53.57%) of the XML elements with EDIFACT-VAL V.2.

3.2.2 RDF Graph Creation for Invoices

EDIFACT invoices can be transformed into RDF representations using our proposed EDIFACT Ontology described in Sect. 3.1. To generate RDF KGs based on a (semi-structured) data source, we define mapping rules. For this, we use the RDF Mapping Language (RML) [4], which provides a way to transform heterogeneous data into the RDF data model. Yet, manually creating RML mapping is a complex task [14], especially when a large number of terms are involved. To ease the definition of the mapping rules, languages like YARRRML [32] and ShExML [7] provide a human-friendly serialization of RML. In this work, we use YARRRML [32] as it allows users to specify simpler mapping rules compared to RML;[1] these mappings are automatically transformed into the RML mappings, later used to transform the invoice XML file into an RDF graph.

YARRRML-Mapping. Our YARRRML mapping contains six mappings, each representing one different class of the EDIFACT ontology. Compared to the EDIFACT ontology, only six of the seven classes are displayed, as the class *InvoiceContent* only has the purpose of providing a domain or range for some resources. When creating each mapping, two types of rules must be defined: (i) rules for defining the data sources, i.e., the XML files of the EDIFACT invoices, and (ii) rules for generating the RDF triples. A notable advantage of EDIFACT-VAL V.2 lies in its more targeted XML construction, as described in the pre-processing step 3.2.1. Due to the less nested structure of the tailored XML file, significantly fewer data source rules are needed. In EDIFACT-VAL V.1, a total of 245 different data source rules needed to be defined, as every different EDIFACT element had a different file path. In contrast, EDIFACT-VAL V.2 requires only four data sources, as the streamlined XML structure consolidates information more effectively. Overall, EDIFACT-VAL V.2 significantly reduces the complexity of the mapping process by consolidating data sources and streamlining XML structures. This improvement is also reflected in the line counts of the mapping files, with 1,690 lines in EDIFACT-VAL V.2 compared to 2,924 lines in the EDIFACT-VAL V.1. Depending on which information is to be displayed in the mappings, the data sources are assigned to the mappings. The rules for generating the RDF triples are divided into two parts. Part one defines the rule for identifying the subject of the RDF triple. In our tool, this is done by

[1] E.g., our YARRRML file has 1,690 lines (2,924 in EDIFACT-VAL V.1) versus 9,734 lines (366,709 in EDIFACT-VAL V.1) in the generated RML file.

Listing 5 SHACL shape for checking the existence of mandatory data and syntactical correctness of EDIFACT invoice elements.

```
:InvoiceNumberShape a sh:NodeShape ;
    sh:targetClass edifact-o:InvoiceDetails ;
    sh:property [
        sh:path edifact-o:hasDocumentNumber ;
        sh:minCount 1 ;
        sh:maxCount 1 ;
        sh:datatype xsd:string ;
        sh:maxLength 12 ;
        sh:message "Invoice number must be a string with max 12 characters." ;
        sh:severity sh:Violation ;
    ] .
```

the added attributes from the preprocessing 3.2.1. This part of the mapping remains consistent across both EDIFACT-VAL V.2 and EDIFACT-VAL V.1. The second part defines the rules for the predicates and objects of the RDF triple. The predicate rule contains the ontology resources, and the object rule contains the identification of the value of the predicate. In the second part of our mapping, the primary difference between EDIFACT-VAL V.2 and EDIFACT-VAL V.1 lies in the updated XML element names resulting from the streamlined XML structure. However, the number of rules for the predicates and objects, as well as their association with the six different YARRRML mappings, remains unchanged.

RML-Mapping. The RML mappings obtained with YARRRML and the invoice XML files are then fed into the RMLMapper[2] tool, to obtain the invoice RDF graph. At this step, the impact of the streamlined XML structure introduced in EDIFACT-VAL V.2 's pre-processing step (Sect. 3.2.1) becomes evident: the generated RML file in EDIFACT-VAL V.2 has only 9,734 lines compared to 366,709 lines in EDIFACT-VAL V.1, meaning a 97.3% reduction in file size. This underscores how the simplified XML structure directly reduces the complexity and overhead of the transformation process. Exemplary transformation of the *NAD segment* from an EDIFACT message to an RDF graph can be seen in Line 4 in Listing 1 to Lines 17-24 and Lines 26-28 in Listing 2.

3.2.3 Invoice Validation using SHACL

Once the invoice RDF graph has been created, the next step is to validate the knowledge graphs. We use the Shapes Constraint Language (SHACL) [15], the W3C-recommended language, for the validation of RDF graphs. Constraints in SHACL can be specified over specific classes of the ontology using the *sh:targetClass* predicate, and over specific properties of the target class using the *sh:path* predicate. The SHACL validation process itself, as well as the shape graphs used, are the same for both EDIFACT-VAL V.1 and EDIFACT-VAL V.2. In EDIFACT-VAL V.2, we enhanced the SHACL shapes by including the use of the *sh:severity* predicate to classify constraint violations according to their impact on further invoice processing. This allows for a more differentiated constraint validation, distinguishing between errors that could hinder subsequent invoice handling and those that do not affect processing.

In our work, the shapes were developed based on input from domain experts, who also identified which constraints result in crucial violations and which do not. Errors that block further processing of an invoice, for example, when legally required information such as the invoice number or a monetary amount is missing, are classified as *sh:Violation*. By contrast, information that supports but is not essential for processing falls under *sh:Warning*; one case is the specification of quantities

[2] https://github.com/RMLio/rmlmapper-java

Listing 6 SHACL shape for checking the existence of mandatory data and syntactical correctness of EDIFACT invoice elements.

```
:BuyerRoleCheckShape a sh:NodeShape ;
    sh:targetClass edifact-o:E-Invoice ;
    sh:sparql [
        a sh:SPARQLConstraint ;
        sh:message "Each invoice must have exactly one BuyerRole,
        and the BuyerRole must be linked to a FormalOrganization." ;
        sh:severity sh:Violation ;
        sh:prefixes [
            sh:declare [
                sh:prefix "edifact-o" ;
                sh:namespace "https://purl.org/edifact/ontology#"^^xsd:anyURI ;
            ] ;
            sh:declare [
                sh:prefix "agentRole" ;
                sh:namespace "https://archive.org/services/purl/domain/
                                modular_ontology_design_library/agentrole#"^^xsd:anyURI ;
            ] ;
            sh:declare [
                sh:prefix "p2p-o-inv" ;
                sh:namespace "https://purl.org/p2p-o/invoice#"^^xsd:anyURI ;
            ]
        ] ;
        sh:select """
            SELECT $this
            WHERE {
              {
                SELECT $this (COUNT(?buyerRole) AS ?buyerCount)
                WHERE {
                  $this p2p-o-inv:hasBuyer ?buyerRole .
                }
                GROUP BY $this
                HAVING (?buyerCount != 1)
              }
              UNION
              {
                $this p2p-o-inv:hasBuyer ?buyerRole .
                FILTER NOT EXISTS { ?buyerRole a <http://example.com/BuyerRole> . }
              }
              UNION
              {
                $this p2p-o-inv:hasBuyer ?buyerRole .
                FILTER NOT EXISTS {
                  ?org a edifact-o:FormalOrganization ;
                       agentRole:performsAgentRole ?buyerRole .
                }
              }
            } """;
    ] .
```

and units of delivered goods. Lastly, details intended purely for internal purposes, such as assigning the invoice to a business process, are considered *sh:Info*, since they do not affect the factual correctness of the invoice.

We distinguish three types of shapes, each with a specific purpose:

(i) Existence of mandatory data. Shapes ensure that the required properties are present in the RDF graph according to the EDIFACT invoice guidelines. Listing 5 (Lines 5 and 6) shows a SHACL constraint for specifying that the *documentNumber* property is mandatory (*sh:minCount* 1) and single valued (*sh:maxCount* 1) for entities of the class *InvoiceDetails*.

(ii) Syntactical correctness of data. Shapes validate the format of the EDIFACT elements, including length, number, and permitted characters. Listing 5 (Lines 7 and 8) shows a SHACL constraint for checking the datatype (*sh:datatype*) and the length (*sh:maxLength*) for the *documentNumber* property of the target class *InvoiceDetails*.

Together, (i) and (ii) ensure the completeness and syntactical correctness of the document number, directly answering Competency Question 6.3 (see Table 1).

(iii) Logical correctness of data. Shapes ensure that the data in the RDF graph is logically consistent and reflects the business logic of an invoice. For example, this includes ensuring that each invoice has exactly one buyer organization linked to it, verifying the logical correctness of the data structure needed to answer Competency Question 2.2 (see Table 1). Such constraints, which involve verifying the existence of a single buyer and ensuring it is properly linked to an organization, can be expressed in SHACL using SPARQL queries, as shown in Listing 6.

4 Evaluation

In our previous work [19], we evaluated the completeness of invoice KG generation and validation with EDIFACT-VAL V.1, including a built-in mechanism to display unknown elements and tests with corrupted invoices. Building on this evaluation, this paper empirically evaluates and compares our approach,

assessing the differences between EDIFACT-VAL V.2 and EDIFACT-VAL V.1, in terms of soundness, performance, and applicability. Concretely, we focus on the following core questions:

Q1 Are RDF graphs created by EDIFACT-VAL V.1 and EDIFACT-VAL V.2 identical? (§4.2)
Q2 How does EDIFACT-VAL V.2 influence the runtime performance of EDIFACT invoice processing compared to EDIFACT-VAL V.1? (§4.3)
Q3 How does EDIFACT-VAL V.2 influence the computational performance of EDIFACT invoice processing compared to EDIFACT-VAL V.1? (§4.4)
Q4 How does EDIFACT-VAL V.2 influence error analysis by introducing error prioritization?(§4.5)
Q5 How does EDIFACT-VAL V.2 affect the applicability to real-world business processes? (§4.6)

4.1 Experimental Set Up

Tool Implementation. EDIFACT-VAL is implemented in Python 3 and available online.[3] For SHACL validation, we use pySHACL [30], an open-source Python library. EDIFACT-VAL is equipped with 394 SHACL constraints provided by the experts following the EDIFACT standard. All experiments have been conducted on a machine with an AMD EPYC 9224 24-Core CPU, 578 GB of RAM, and running Ubuntu 24.04.1 LTS.

Dataset. We use 44 different real-world EDIFACT messages from 12 different suppliers and 6 different business cases to evaluate the performance of the EDIFACT-VAL tool. The selection of messages has been made together with the EDI experts from E/D/E to have a range of messages representing the day-to-day business workflow. Since each supplier may include different segments and data elements in the invoices, the selected messages represent a wide range of segments and segment combinations. All 394 SHACL constraints matched at least one triple across the 44 EDIFACT messages used in the evaluation, while also all resulted in at least one validation error. This corresponds to a relevance coverage of 100% and a violation coverage of 100%. These constraints, however, do not cover all data represented in the invoice knowledge graphs. Rather, they focus on the most critical aspects of invoice validation for legal compliance with e-invoice regulations, like mandatory standardized organizational data and required financial calculations.

[3] `https://github.com/DE-TUM/EDIFACT-VAL`

Table 3 Comparison of RDF graph statistics for each message generated by EDIFACT-VAL v.1 and EDIFACT-VAL v.2. Message names are omitted for privacy.

Message	#Triples		#Nodes		#Properties	
	Old	New	Old	New	Old	New
Message 1	96	96	12	12	67	67
Message 2	401	401	27	27	71	71
Message 3	982	989	70	70	78	79
Message 4	130	132	13	13	68	70
Message 5	143	143	12	12	81	81
Message 6	393	398	31	31	63	65
Message 7	36,170	36,165	2,091	2,091	79	79
Message 8	113	113	11	11	66	67
Message 9	122	122	11	11	76	76
Message 10	111	111	11	11	70	71
Message 11	290	282	20	20	75	76
Message 12	3,152	3,135	209	209	91	91
Message 13	921	920	56	56	73	74
Message 14	440	440	23	23	81	82
Message 15	4,634	4,629	307	307	80	80
Message 16	531	532	29	29	72	74

4.2 Equivalence of the Invoice Generated by EDIFACT-VAL v.1 and v.2

Since the completeness and correctness of EDIFACT-VAL v.1 have already been established in [19], it is sufficient to compare the RDF graphs generated by EDIFACT-VAL v.2 to those of EDIFACT-VAL v.1 using a graph isomorphism check with `rdflib` [16].

First, we loaded the RDF graphs produced by both approaches and compared them directly by checking for graph isomorphism. *Rdflib*'s built-in isomorphism function showed that there were some triples present only in the EDIFACT-VAL v.2 and some only in the EDIFACT-VAL v.1. To identify these triples, we compared key statistics of the graphs, including the number of triples, nodes, and distinct properties, as summarized in Table 3. When comparing the key statistics of different EDIFACT messages, we observed notable variation in the number of generated triples. As discussed in Section 2, this variation arises from the complexity of the messages, which is influenced by the number of contained invoices, the length and detail of their segment structures, and the types of segments used.

After discussions with domain experts to investigate why certain triples were missing in one of the versions, we identified the following reasons: The reasons were split into those explaining why elements appear in the RDF graphs of EDIFACT-VAL v.1 and not in EDIFACT-VAL v.2, and vice versa. We begin by discussing why some elements are present in EDIFACT-VAL v.1 but missing in EDIFACT-VAL v.2.

(i) Reduced Triple Count Due to EDIFACT Syntax Errors. One issue arises from the syntax of EDIFACT messages, where certain characters, such as + and ′, are used to indicate the start of the next element or segment. In some of the evaluated messages, these characters were also used as content within data elements, leading to incorrect splitting of the data elements. In EDIFACT-VAL v.1, the input was an XML version of the invoice, and the in-house translation by E/D/E was able to handle these errors. However, because the use of these symbols as content constitutes an error and should be communicated to the customers, we decided to treat them as errors. This issue frequently occurred in the NAD-Segment, which describes the involved organizations, as some companies include the character + in their name. As a result, the syntax of the following data elements, such as postal code or country code, was often missing. This led to a reduction in the number of properties in the RDF graph created for Message 12 by EDIFACT-VAL v.2, as summarized in Table 3.

(ii) Reduced Triple Count from Filtering Empty Data Elements. As outlined in Section 3.2.1, EDIFACT-VAL V.2 omits data elements that have no content in the RDF graph. In contrast, EDIFACT-VAL V.1 did not apply this filtering, resulting in the presence of triples with empty objects (`""`) that are no longer generated in the RDF graph by EDIFACT-VAL V.2. Consequently, this filtering step also reduces the number of properties represented in the RDF graphs created by EDIFACT-VAL V.2, as summarized in Table 3. It is important to note that the magnitude of this reduction varies between messages, since the messages are sourced from different suppliers with differing invoice practices. For instance, some suppliers – such as the supplier of Message 1 – do not include empty data elements, while others – like the supplier of Message 15 – do include them.

(iii) Increased Triple and Property Counts from Improved Segment Allocation. The simpler creation of the XML file in EDIFACT-VAL V.2 has led to a reduction in the number of nested XML elements. As described in Section 3.2.1, this simplification has reduced the number of attributes needed during the pre-processing step to correctly identify and allocate the segments. This is particularly relevant for segments that can occur in all three parts of the message, such as the *RFF-Segment* and the *DTM-Segment*. In EDIFACT-VAL V.1, it was often difficult to correctly allocate these segments because of the complex nested XML structure. Since certain specific combinations of these segments (for example, the valuta date represented by *DTM+209*) only occur in particular parts of the message, precise allocation is crucial and determines whether they are represented in the RDF graph. This improvement in the XML creation process directly affects the completeness of the RDF graph and explains why, in some cases, EDIFACT-VAL V.2 generates more triples and properties than EDIFACT-VAL V.1. This is particularly evident in the results for Messages 3 and 4 in Table 3, where Message 3 has one additional property and Message 4 has two additional properties in the new version of the RDF graph, and therefore also more triples. These findings highlight that the simpler XML creation in EDIFACT-VAL V.2 not only reduces unnecessary data but also ensures that critical segments are correctly represented, contributing to a more complete and accurate RDF graph.

Since the number of triples differs between the RDF graphs generated by EDIFACT-VAL V.1 and EDIFACT-VAL V.2, we cannot claim that the RDF graphs are identical. However, our investigations demonstrate that the RDF graph generated by EDIFACT-VAL V.2 offers an improvement in data representation. Importantly, the number of nodes (entities) remains the same in both graphs, ensuring that all concepts and entities of the invoices are still displayed. At the same time, EDIFACT-VAL V.2 allows us to: (i) omit empty data elements, (ii) identify and exclude errors that were previously undetectable, and (iii) represent information that was previously not included due to the nested structure of the input data. These improvements collectively contribute to a more reliable, accurate, and domain-expert-validated foundation for further processing and analysis of EDIFACT invoices.

4.3 Runtime Performance of EDIFACT-Val

We compare the efficiency of the different versions of EDIFACT-VAL when processing the invoices, i.e., the elapsed time between the tool receiving an EDIFACT message (KG Construction) and producing the validation result (KG Validation). This time includes the generation of the RDF graph and its evaluation using the SHACL constraints. To ensure representative results, we selected one EDIFACT message per supplier, which may contain several invoices with varying numbers of EDIFACT data items. We ran EDIFACT-VAL 10 times on each EDIFACT message using the `hyperfine` [24] command-line benchmarking tool. Figure 3 shows the average runtime per message for both EDIFACT-VAL V.1 (in blue) and EDIFACT-VAL V.2 (in red), with a darker tone representing KG Construction and a lighter tone representing KG Validation. Since the difference

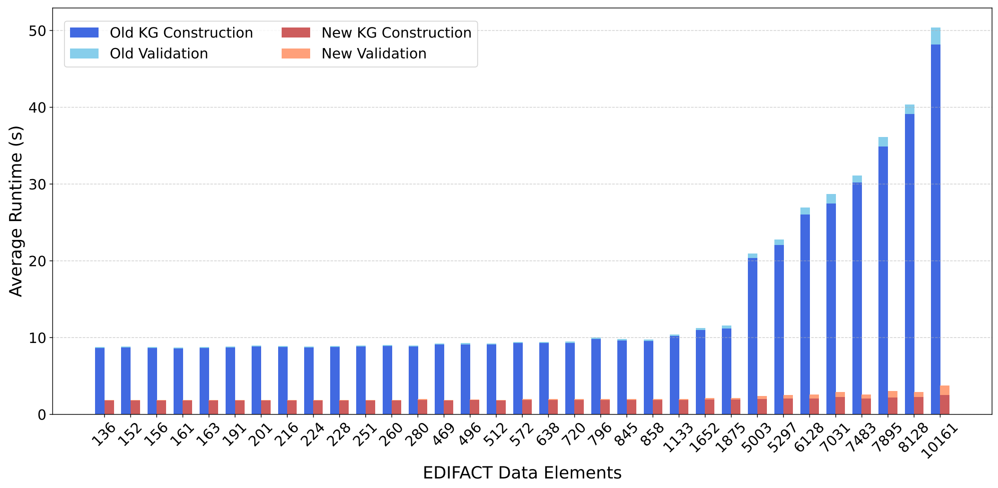


Figure 3 Runtime Comparison: EDIFACT-VAL V.1 versus EDIFACT-VAL V.2.

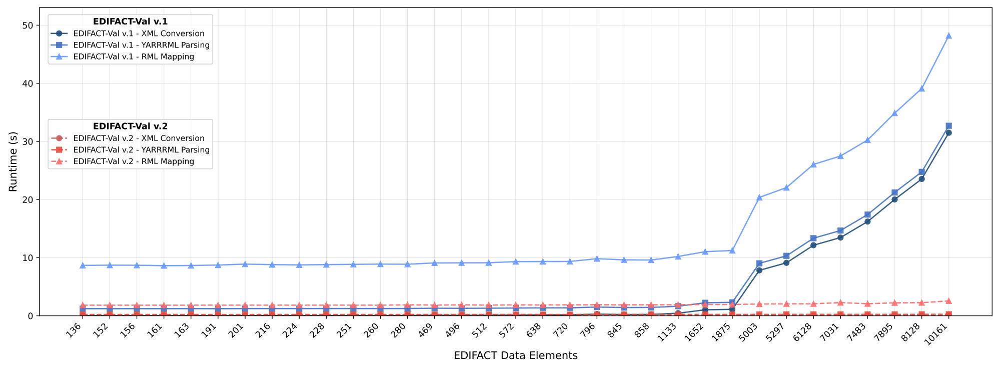


Figure 4 KG Construction Breakdown for EDIFACT-VAL.

between EDIFACT-VAL V.1 and EDIFACT-VAL V.2 lies only in the KG Construction phase, we split the analysis into three parts: (1) general observations, (2) KG construction, and (3) KG validation. This enables us to isolate the performance improvements achieved by EDIFACT-VAL V.2 in the RDF graph generation step.

(1) General Observations. The runtime comparison shown in Figure 3 reveals a significant decrease in the total runtime of EDIFACT-VAL V.2 compared to EDIFACT-VAL V.1. For smaller EDIFACT messages, the runtime with EDIFACT-VAL V.1 was around 8 seconds and is now reduced to about 2 seconds with EDIFACT-VAL V.2. For larger EDIFACT messages, the runtime drops substantially, from 50 seconds with EDIFACT-VAL V.1 to under 4 seconds with EDIFACT-VAL V.2. These observations show that the improvement is consistent across all tested messages and demonstrate that EDIFACT-VAL V.2 effectively handles increasing input sizes. A closer examination of each runtime's composition (Figure 3) shows that this improvement results from the KG construction phase, which is the only part where EDIFACT-VAL V.1 and EDIFACT-VAL V.2 differ.

(2) KG Construction. As noted before, the time spent on KG Construction is where EDIFACT-VAL V.2 is significantly more efficient than EDIFACT-VAL V.1. Looking only at the times for KG Construction, the average percentage of time saved with EDIFACT-VAL V.2 is 82.10%. There are several reasons for this improvement: First, with the tailored pre-processing described in

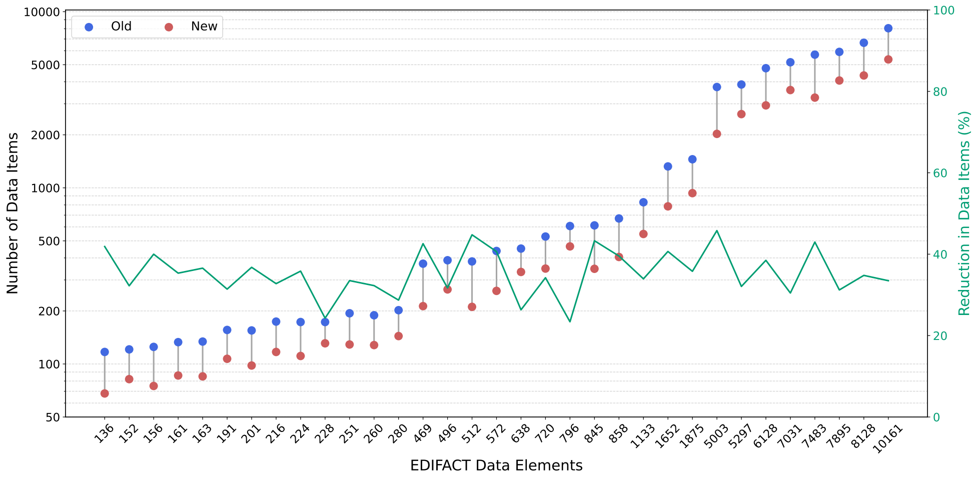


Figure 5 Number of data items and percent Reduction: EDIFACT-VAL V.1 versus EDIFACT-VAL V.2.

Section 3.2.1, the number of created XML elements is reduced by removing unnecessary elements and avoiding nested structures that do not contribute to the final RDF graph. Figure 5 provides an overview of the number of XML elements generated by EDIFACT-VAL V.2 and EDIFACT-VAL V.1 for each EDIFACT message. This figure shows that, on average, the number of XML elements is reduced by 35.39% across all EDIFACT messages. This reduction means that there are fewer XML elements that need to be checked and processed by the RDF mappings in the following step. As a result, the RDF mappings can be executed more quickly and efficiently, which directly speeds up the KG construction process, as shown in Figure 4. A second improvement of the pre-processing that significantly speeds up the KG construction time is that the created XML files no longer need to be parsed and saved multiple times. In EDIFACT-VAL V.1, the XML files were repeatedly parsed, modified, and saved again, leading to redundant file handling operations. By contrast, EDIFACT-VAL V.2 streamlines this process by parsing, modifying, and saving the XML content in a single step within the pre-processing phase. This eliminates unnecessary operations, reduces memory usage, and improves the overall computational efficiency. As a result, the KG construction process becomes more streamlined and significantly faster, contributing to the overall runtime improvement observed in Figure 4. Third, Section 3.2.2 describes the reduction of both YARRRML-Mapping and RML-Mapping in EDIFACT-VAL V.2. The YARRRML-Mapping was reduced by 42.2% due to fewer needed data sources, which led to a reduction of the RML-Mapping by 97.3%. This substantial reduction directly impacts the KG construction runtime by decreasing the parsing and processing overhead for the RML Mapper, simplifying the graph construction phase, and reducing the number of triple patterns to be executed. Overall, the leaner and more direct mapping instructions in EDIFACT-VAL V.2 make the RDF graph generation faster and more efficient, as reflected in the runtime comparison shown in Figure 3.

(3) KG Validation. Since the RDF graphs created by EDIFACT-VAL V.1 and EDIFACT-VAL V.2 are almost identical, as discussed in Section 4.2, the number of RDF triples is also nearly the same (see Table 3). Consequently, the runtime of the RDF validation with SHACL is also similar in both approaches. Across all tested messages, the validation time ranges from as little as 0.05 seconds for smaller EDIFACT messages to a maximum of 1.2 seconds for the largest EDIFACT messages. On average, the runtime reduction in the KG validation phase is 10.8%, which can be attributed to the slight reduction in triples described in Section 4.2.

Overall, the analysis of the runtime shows that EDIFACT-VAL V.2 is significantly more efficient than EDIFACT-VAL V.1, thanks to its optimized pre-processing and RDF mapping steps. The runtime per EDIFACT element is reduced from 28.53 milliseconds to 9.54 milliseconds, resulting in a normalized runtime reduction of 66.56%. This substantial improvement stems from the more efficient KG construction phase in EDIFACT-VAL V.2, which eliminates redundant processing steps and accelerates the generation of the RDF graph.

4.4 Computational Performance of EDIFACT-Val

Since we use a large-scale machine to conduct the runtime experiments, AMD EPYC 9224 24-Core CPU, 578 GB of RAM, we monitored the CPU and memory usage during the experiments with both EDIFACT-VAL V.1 and EDIFACT-VAL V.2. The results of CPU usage (Figures 6 and 7) and memory usage (Figures 8 and 9) provide insights into the efficiency of both approaches. The average CPU usage is calculated as the mean of all monitored CPU utilization across all cores, while the peak CPU usage corresponds to the maximum recorded value. The same definition applies to the memory usage metric.

The CPU measurements show that EDIFACT-VAL V.1 has a lower average overall usage of about 4.1% compared to 6.5% for EDIFACT-VAL V.2. EDIFACT-VAL V.2. However, both approaches utilize a small fraction of the available 24-core CPU, with most cores remaining inactive during validation. The peak overall usage is similar at around 12.6%, but the distribution across cores differs: EDIFACT-VAL V.1 occasionally drives individual cores up to 72%, while EDIFACT-VAL V.2 remains below 50% per core , with an average maximum core utilization of 25.32% for EDIFACT-VAL V.1 and 16.17% for EDIFACT-VAL V.2. These results indicate that in both approaches, the workload is unevenly distributed across cores, with a few cores handling short bursts of higher load while most remain underutilized. Detailed per-core utilization values are available in the project's GitHub repository. An unexpected observation in Figure 6 is that the average CPU utilization is lower for the larger EDIFACT message. This occurs because a larger portion of the processing time is spent in components of the approach that are less CPU-intensive, leading to a lower overall average utilization. This behavior is shown in Figure 7. Consequently, despite the slightly higher average load, EDIFACT-VAL V.2 makes more efficient use of the available hardware resources, as it completes execution significantly faster. Although it utilizes the CPU more intensively at a given moment, it holds the CPU for a much shorter time, resulting in lower total CPU time consumption and thus higher overall efficiency.

The results in Figure 8 show a clear difference in memory consumption between the two approaches. The old approach requires on average between 1.2–1.6 GB, with peak usage rising as the number of EDIFACT data elements increases. In contrast, the new approach consistently stays close to 1.0–1.1 GB for both average and peak usage, showing only minor variations even for large messages. On average, this corresponds to roughly a 30% lower memory footprint compared to the old approach. This means that while EDIFACT-VAL V.1 shows a growing memory demand with message size, EDIFACT-VAL V.2 maintains a stable and minimal footprint regardless of input size. When comparing the memory usage over time for one EDIFACT message (Figure 9), it becomes evident that both approaches show relatively stable memory usage during the initial stages, followed by a clear increase towards the end of processing. The main peak in both plots occurs near the completion phase, where the message validation and result aggregation are performed.

While EDIFACT-VAL V.1 exhibits stronger fluctuations and multiple smaller peaks throughout the execution, EDIFACT-VAL V.2 maintains a smoother progression and a lower overall peak, indicating more consistent and efficient memory utilization. Combined with its shorter runtime, this stability makes EDIFACT-VAL V.2 more predictable and better suited for large-scale or resource-constrained deployments.

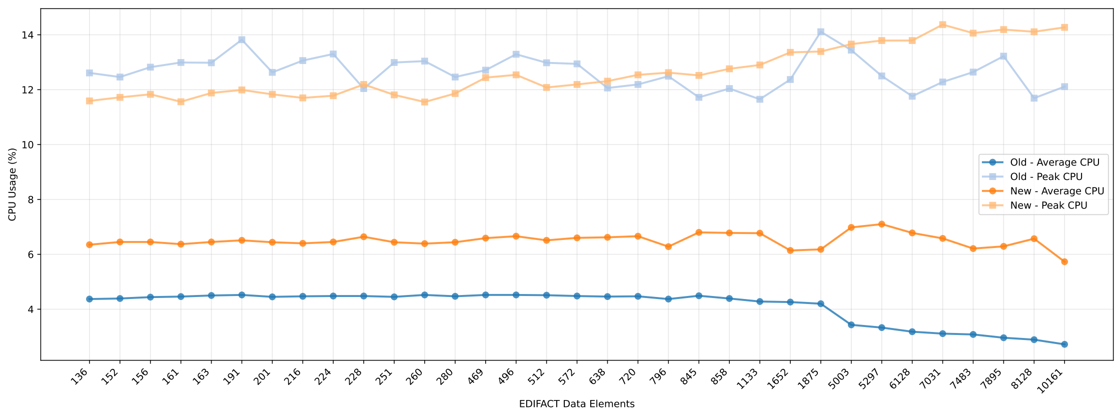


Figure 6 CPU usage comparison for EDIFACT-VAL. Values show the mean utilization across all CPU cores during processing.

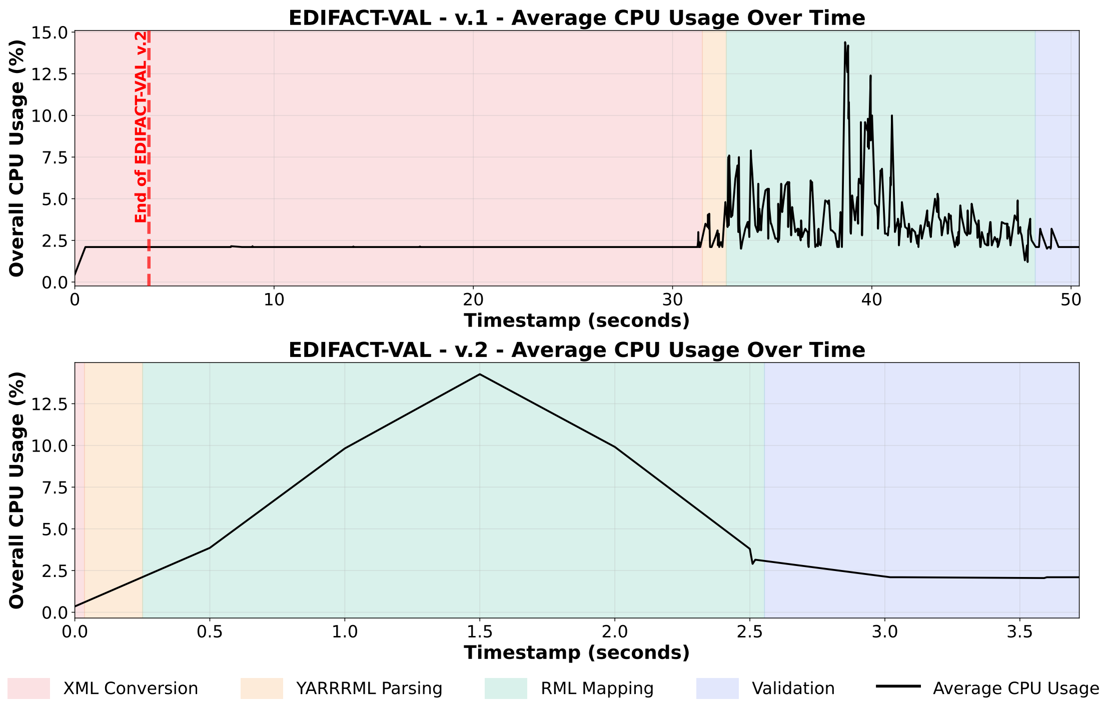


Figure 7 CPU usage timeline for message *10161*.

Overall, the results show that while EDIFACT-VAL V.1 consumes slightly less CPU on average, its longer runtimes and higher memory demand make it far less efficient, whereas EDIFACT-VAL V.2 combines faster execution with lower memory usage, offering a more scalable solution.

4.5 Error Prioritization Evaluation

In our previous work [19], we discussed that not all errors have the same severity. Therefore, classifying the different constraints in SHACL using the *sh:severity* predicate became essential. Section 3.2.3 motivates and describes how this classification was implemented. Incorporating

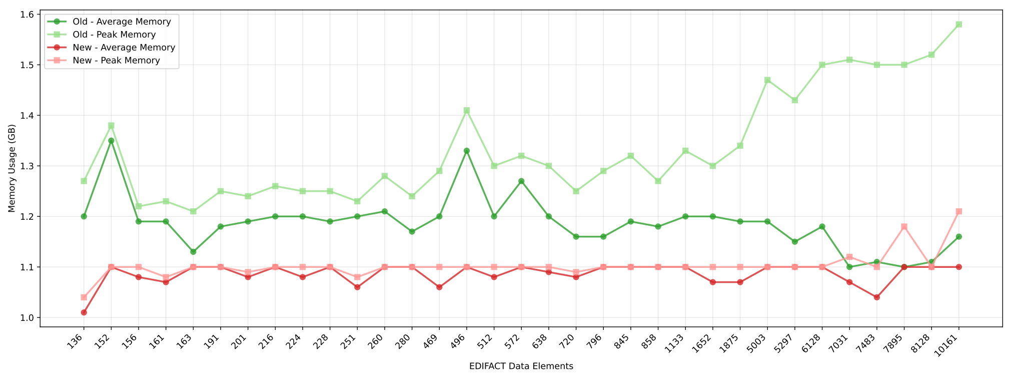


Figure 8 Memory usage comparison for EDIFACT-VAL.

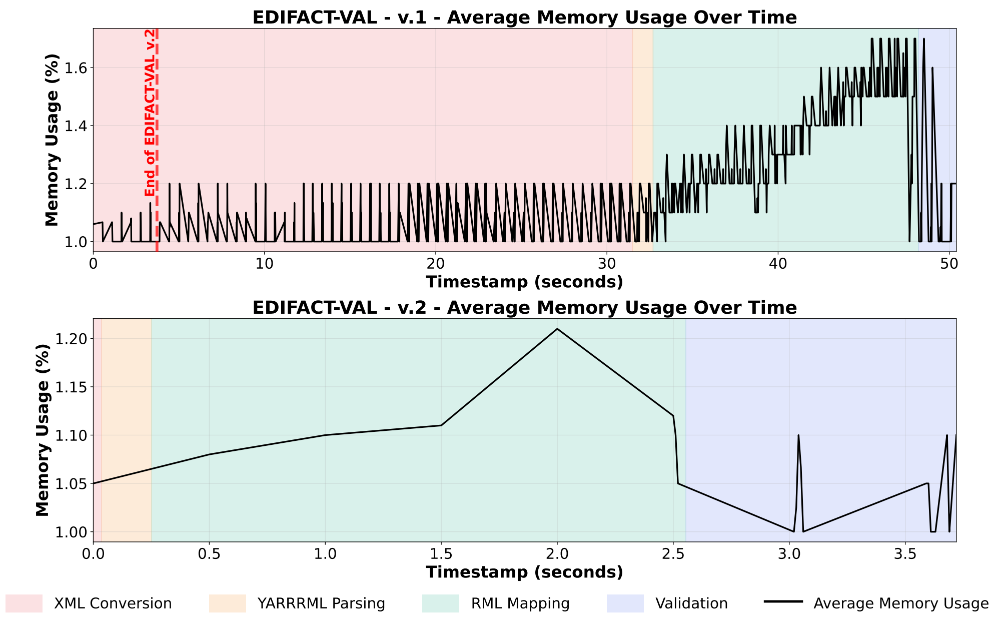


Figure 9 Memory usage timeline for message *10161*.

sh:severity simplifies the analysis of validation reports by prefiltering the errors and highlighting the errors that require the most attention. Table 4 presents the results for an exemplary Business Process from [19]. This refinement led to a 100% reduction in the number of violations categorized as *sh:Violation*. Previously, these messages were incorrectly classified as violations, even though they did not affect further invoice processing. This made it difficult for the experts to identify the errors on which they had to work right away, as all types of errors were handled with the same priority. The refined classification makes it significantly easier for domain experts to identify and address truly impactful issues. As the results from EDIFACT-VAL V.2 now show that the errors in the messages are not crucial for the processing of the invoices, the experts can now work on the messages with errors that are crucial for the processing, and work on other issues after resolving the urgent errors. Importantly, prioritizing constraints in the SHACL shapes did not affect the runtime of the validation process.

Table 4 Comparison of RDF graph statistics with and without error prioritization.

	EDIFACT-VAL V.1	EDIFACT-VAL V.2		
Business Process	**#Violation**	**#Violation**	**#Warning**	**#Info**
BP1	21	15	4	2

Comparison of *sh:severity* options with and without prioritization for an exemplary Business Process.

	EDIFACT-VAL V.1	EDIFACT-VAL V.2		
Message	**#Violation**	**#Violation**	**#Warning**	**#Info**
Message 1	33	0	33	0
Message 2	6	0	6	0
Message 3	3	0	3	0
Message 4	63	0	63	0

Statistics for each message, showing *sh:severity* options with and without prioritization.

4.6 In-Use Evaluation

Since the development was motivated by the group purchasing organization E/D/E, we also evaluated the applicability of our proposed solution approach to the E/D/E invoice processes. We asked the domain experts to validate the EDIFACT messages following the usual procedure, which is done manually. On average, it takes the experts around 30 minutes to validate one EDIFACT message. In comparison to EDIFACT-VAL V.2 (cf. Figure 3), the longest runtime is in the order of 3.65 seconds, which results in a time saving of around 99.98%.

Beyond runtime, we also assessed applicability with E/D/E domain experts. The constraints and validation reports have already been integrated into their internal testing workflows, where they are used to check newly received invoices before processing. While the tool is not yet deployed as a full production service, E/D/E experts confirmed that the validation results align with their manual procedures and that the prioritization mechanism helps them focus on legally critical errors. This pilot use indicates that the approach is practical and paves the way for continued organizational adoption.

5 Related Work

Ontologies for Electronic Invoices. Schulze et al. [27] propose an ontology, the Purchase-To-Pay Ontology (P2P-O), to represent electronic invoices. P2P-O relies on the European Standard EN 16931-1:2017, intending to provide ontology resources for all mandatory information in a purchase-to-pay process, which includes the invoicing process. In contrast, we aim not only to provide mandatory information for processing an electronic invoice, we also aim to validate the completeness and syntactical correctness of the invoices regarding their specific format. Furthermore, the terms used in the EDIFACT messages are not captured in the P2P-O ontology. This is why the creation of a standard-specific ontology, the EDIFACT ontology, is crucial in this case. In the literature, we also found several ontologies and vocabularies that provide invoice-related definitions. The ones we reuse can be found in Table 2. However, they only present small parts of invoices. For example, schema.org [9] provides terms for the amount of money and currencies. Yet, these vocabularies are not specific enough to model all the EDIFACT terms.

Invoice Validation. Several works have tackled the problem of invoice validation from different perspectives. Emmanuel and Thakur [5] present an approach that implements an EDI invoice validation framework; this work focuses on checking the completeness of invoices (defined as the number of fields) by comparing actual values to expected values as given in an XML file. The expected value is defined in a rule set. Similarly, our work validates invoices but uses KG technologies and more expressive SHACL constraints formulated by experts and the EDIFACT standard. Other approaches also perform invoice validation but not in the context of EDI. For example, Sál [25] presents a tool to check whether the invoice fields are provided correctly to a digital system w.r.t. to the original invoice in PDF. Other solutions [1, 3, 10] propose processing and classifying invoices in PDF using Machine Learning (ML) models. Similarly, a comprehensive survey by Saout et al. [26] reviews a wide range of extraction and structure recognition techniques for invoice processing, including NLP and graph-based methods. Another approach by Hamri et al. [11] explores information extraction from invoices by studying document anatomy and training models on synthetic data to handle template variants. Additionally, works by Bisetty et al. [2] and Milosheski et al. [22] explore how ERP systems and automated invoicing systems improve operational efficiency through workflow-level automation. All these works check for the correctness of the translation of the invoices into machine-readable formats, rather than the correctness of the original invoices. These approaches greatly differ from EDIFACT-VAL, as it processes invoices that are already in a machine-readable format and validates the correctness of the original EDIFACT invoice. Lastly, the work by Schulze et al. [27] uses SPARQL queries to perform analytical tasks on the invoices. Examples of these include finding items that are sold in large cities. In contrast, EDIFACT-VAL is tailored to check the conformance of the invoices to the EDIFACT standard and other constraints defined by experts.

6 Lessons Learned

The development and application of EDIFACT-VAL in real-world scenarios provided us valuable insights into the use of Knowledge Graph technologies for invoice validation. In particular, we found that a carefully designed Shapes Graph significantly improves usability, especially when dealing with large and complex data graphs. In the following, we elaborate on these aspects.

1. Importance of a Business Graph for Cross-Checking. A lesson learned from our work is the importance of creating a dedicated Business Graph that integrates all included companies and the items sold. This takeaway emerged because we observed inconsistencies in supplier and product information across different invoices, raising concerns about the overall correctness of the data. Without a ground truth for these entities, one can only check the logical correctness within a single invoice, but this local correctness does not necessarily guarantee global correctness across all invoices. By linking supplier data and product details within the Business Graph, discrepancies and missing links can be automatically identified, which significantly reduces manual validation efforts. Furthermore, this integrated view supports robust data quality assurance, as it provides domain experts with a reliable source of truth for verifying the correctness of invoice data. Developing and evaluating such a comprehensive Business Graph will be a focus of our future work.

2. Lessons for Effective Shape Design. When using SHACL to formulate Invoice Guidelines as constraints, we identified several key takeaways. We present these insights starting from the smallest instance (Shapes) and moving up to the largest instance (Shape Graphs).

(i) Naming Conventions for Shape Graphs A prerequisite for meaningful validation results is the ability to trace each violation back to the shape that caused it. To achieve this, SHACL shapes should follow a consistent naming convention. Without such conventions, the system automatically generates blank nodes, which prevents cross-shape querying and limits the usefulness of the validation report. Establishing a naming convention also increases reusability, since many shapes can be applied across slightly different business processes with minimal adjustments. In the invoicing domain, where processes often differ only in detail, this reusability is particularly valuable. With a well-designed convention in place, shapes can be shared across scenarios, making the validation process both more efficient and more maintainable.

(ii) CQ-Centric Shape Modeling Furthermore, the decision regarding which constraints to include within a single shape is of importance. Our experience has shown that it is more effective to aggregate all constraints that collectively answer a specific competency question within a single shape, rather than fragmenting them based solely on the structural arrangement of the data elements. The shapes presented in Listing 5 and Listing 6 exemplify this approach, as they incorporate all relevant constraints needed to address a particular competency question. Modeling shapes in this manner enables a more systematic and targeted analysis of the validation report, as violations associated with a shape directly indicate the root cause of the issue.

(iii) Domain-Specific Shape Graphs In a company, different departments often have different validation interests. For example, the financial department focuses on the correctness of monetary amounts, while the warehouse department is primarily concerned with the quantities of sold items. In many cases, these interests are disjoint, making it possible to split the shapes accordingly. This separation reduces the runtime of the entire approach, as fewer constraints need to be validated and fewer potential violations need to be processed.

7 Conclusion and Future Work

We presented an updated version of the automatic approach EDIFACT-VAL that assists EDI experts in validating EDIFACT messages using an invoice knowledge graph. In EDIFACT-VAL, EDIFACT messages are transformed into RDF graphs based on the proposed EDIFACT Ontology and RML mappings. These RDF graphs are then validated using SHACL constraints that were developed in collaboration with domain experts. The new version, EDIFACT-VAL V.2, directly processes the original EDIFACT messages as input and employs a tailored pre-processing strategy. This pre-processing not only simplifies the RDF mapping process but also improves the overall data quality and ensures more precise validation of the EDIFACT messages.

Based on our experiments, we observed that the RDF graphs generated by EDIFACT-VAL V.1 and EDIFACT-VAL V.2 are not identical, as the graphs produced by EDIFACT-VAL V.2 represent a more reliable and accurate version of the EDIFACT message. Additionally, EDIFACT-VAL V.2 achieves a runtime reduction of 66.56% compared to EDIFACT-VAL V.1, making it not only more robust but also significantly faster. The evaluation with domain experts further demonstrated that EDIFACT-VAL can substantially reduce manual efforts in the validation process. The collaboration with E/D/E also shows that the approach can be applied beyond synthetic settings. The tool is currently used in pilot workflows, with plans for integration into routine operations. A crucial takeaway from this evaluation is that the effectiveness of automatic approaches like EDIFACT-VAL ultimately depends on the reliability of the validation results. At present, this information is available only in a machine-readable format, making it challenging for human users to analyze and act upon. Future work should therefore focus on tailoring frameworks for automated analysis and a visual presentation of validation results [20] to further support human decision-making.

Future work could extend our solution to support different business documents in various EDIFACT formats, such as purchase order messages (ORDERS) or despatch advice messages (DESADV). Additionally, with the mandatory implementation of electronic invoices in Germany starting in 2025, the adoption of open standards has become increasingly important, particularly for organizations that find the cost or complexity of traditional EDIFACT implementations challenging. In this context, we hope that our work contributes to the broader adoption of electronic invoice processing based on knowledge graph and semantic web technologies.

Supplementary Material Statement

The source code for this work is publicly available on GitHub under the following link `https://github.com/DE-TUM/EDIFACT-VAL`. Due to data privacy and confidentiality obligations, the dataset used in this study cannot be made available.

References

1 Dipali Baviskar, Swati Ahirrao, and Ketan Kotecha. Multi-layout unstructured invoice documents dataset: A dataset for template-free invoice processing and its evaluation using ai approaches. *IEEE Access*, 9:101494–101512, 2021. `doi:10.1109/ACCESS.2021.3096739`.

2 SSSS Bisetty, Aravind Ayyagari, Archit Joshi, Om Goel, Lalit Kumar, and Arpit Jain. Automating invoice verification through erp solutions. *International Journal of Research in Modern Engineering and Emerging Technology*, 12(5):131, 2024.

3 Devanshi Desai, Ansh Jain, Dhaivat Naik, Nishita Panchal, and Dattatray Sawant. Invoice processing using rpa & ai. In *Proceedings of the International Conference on Smart Data Intelligence (ICSMDI 2021)*, 2021.

4 Anastasia Dimou, Miel Vander Sande, Pieter Colpaert, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. Rml: A generic language for integrated rdf mappings of heterogeneous data. *Ldow*, 1184, 2014. URL: `https://ceur-ws.org/Vol-1184/ldow2014_paper_01.pdf`.

5 M Emmanuel and Sucheta Thakur. An approach to develop invoice validation framework. *International Journal of Engineering Research & Technology (IJERT)*, 1, 2012.

6 EN16931-1:2017. Electronic invoicing-part 1 : Semantic data model of the core elements of an electronic invoice. Technical report, European Committee for Standardization, 2017.

7 Herminio García-González, Iovka Boneva, Sławek Staworko, José Emilio Labra-Gayo, and Juan Manuel Cueva Lovelle. Shexml: improving the usability of heterogeneous data mapping languages for first-time users. *PeerJ Computer Science*, 6:e318, 2020. `doi:10.7717/PEERJ-CS.318`.

8 Michael Grüninger and Mark S Fox. The role of competency questions in enterprise engineering. In *Benchmarking—Theory and practice*, pages 22–31. Springer, 1995.

9 Ramanathan V Guha, Dan Brickley, and Steve Macbeth. Schema.org: Evolution of Structured Data on the Web. *Communications of the ACM*, 59(2):44–51, 2016. `doi:10.1145/2844544`.

10 Hiruni Gunaratne and Ingrid Pappel. Enhancement of the e-invoicing systems by increasing the efficiency of workflows via disruptive technologies. In *Electronic Governance and Open Society: Challenges in Eurasia: 7th International Conference, EGOSE 2020, St. Petersburg, Russia, November 18–19, 2020, Proceedings 7*, pages 60–74. Springer, 2020.

11 Mouad Hamri, Maxime Devanne, Jonathan Weber, and Michel Hassenforder. Document information extraction: An analysis of invoice anatomy. *Applied Computational Intelligence and Soft Computing*, 2024(1):7599415, 2024. `doi:10.1155/2024/7599415`.

12 Matthew Horridge, Rafael S Gonçalves, Csongor I Nyulas, Tania Tudorache, and Mark A Musen. Webprotégé: A cloud-based ontology editor. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 686–689, 2019. `doi:10.1145/3308560.3317707`.

13 Christian Huemer, Philipp Liegl, and Marco Zapletal. Electronic data interchange and standardization. *Handbook of e-Tourism*, pages 1–29, 2020.

14 Ana Iglesias-Molina, David Chaves-Fraga, Ioannis Dasoulas, and Anastasia Dimou. Human-Friendly RDF Graph Construction: Which One Do You Chose? In *International Conference on Web Engineering*, pages 262–277. Springer, 2023. `doi:10.1007/978-3-031-34444-2_19`.

15 Holger Knublauch and Dimitris Kontokostas. Shapes constraint language (SHACL). W3C recommendation, W3C, July 2017. URL: `https://www.w3.org/TR/2017/REC-shacl-20170720/`.

16 Daniel Krech, Gunnar Aastrand Grimnes, Graham Higgins, Nicholas Car, Jörn Hees, Iwan Aucamp, Niklas Lindström, Natanael Arndt, Ashley Sommer, Edmond Chuc, Ivan Herman, Alex Nelson, Jamie McCusker, Tom Gillespie, Thomas Kluyver, Florian Ludwig, Pierre-Antoine Champin, Mark Watts, Urs Holzer, Ed Summers, Whit Morriss, Donny Winston, Drew Perttula, Filip Kovacevic, Remi Chateauneu, Harold Solbrig, Benjamin Cogrel, and Veyndan Stuart. RDFLib, January 2025. `doi:10.5281/zenodo.6845245`.

17 Steffen Lohmann, Vincent Link, Eduard Marbach, and Stefan Negru. Webvowl: Web-based visualization of ontologies. In *Knowledge Engineering and Knowledge Management: EKAW 2014 Satellite Events, VISUAL, EKM1, and ARCOE-Logic, Linköping, Sweden, November 24-28, 2014. Revised Selected Papers. 19*, pages 154–158. Springer, 2015. `doi:10.1007/978-3-319-17966-7_21`.

18 Johannes Mäkelburg and Maribel Acosta. EDIFACT-VAL. Software, version 2.0., swhId: `swh:1:dir:d820a10d861cfd9e361208220cf7aad3b03ef1a6` (visited on 2025-12-05). URL:

https://github.com/DE-TUM/EDIFACT-VAL, doi:10.4230/artifacts.25246.

19 Johannes Mäkelburg, Christian John, and Maribel Acosta. Automation of electronic invoice validation using knowledge graph technologies. In *European Semantic Web Conference*, pages 253–269. Springer, 2024. doi:10.1007/978-3-031-60626-7_14.

20 Johannes Mäkelburg, Zenon Zacouris, Jin Ke, and Maribel Acosta. SHACL Dashboard: Analyzing Data Quality Reports over Large-Scale Knowledge Graphs. In *Proceedings of the 24th International Semantic Web Conference (ISWC 2025)*, Lecture Notes in Computer Science. Springer, 2025. to appear. doi:10.1007/978-3-032-09530-5_6.

21 James McKinney and Renato Iannella. vCard Ontology – for describing People and Organizations. W3C note, W3C, May 2014. URL: https://www.w3.org/TR/2014/NOTE-vcard-rdf-20140522/.

22 Filip Milosheski and Aleksandar Karadimce. Improving operational efficiency with an automated invoicing system. *Financial Engineering*, 3:221–230, 2025.

23 Natalya F Noy, Deborah L McGuinness, et al. Ontology development 101: A guide to creating your first ontology, 2001.

24 David Peter. hyperfine, March 2023. URL: https://github.com/sharkdp/hyperfine.

25 Jan Sál. Data mining as tool for invoices validation. *IT for Practice 2018*, page 121, 2018.

26 Thomas Saout, Frédéric Lardeux, and Frédéric Saubion. An overview of data extraction from invoices. *IEEE Access*, 12:19872–19886, 2024. doi:10.1109/ACCESS.2024.3360528.

27 Michael Schulze, Markus Schröder, Christian Jilek, Torsten Albers, Heiko Maus, and Andreas Dengel. P2p-o: A purchase-to-pay ontology for enabling semantic invoices. In *The Semantic Web: 18th International Conference, ESWC 2021, Virtual Event, June 6–10, 2021, Proceedings 18*, pages 647–663. Springer, 2021. doi:10.1007/978-3-030-77385-4_39.

28 Cogan Shimizu, Quinn Hirt, and Pascal Hitzler. Modl: A modular ontology design library. *arXiv preprint arXiv:1904.05405*, 2019. arXiv:1904.05405.

29 David Shotton. FRAPO, the Funding, Research Administration and Projects Ontology, April 2017. URL: http://purl.org/cerif/frapo.

30 Ashley Sommer and Nicholas Car. pySHACL, January 2022. doi:10.5281/zenodo.4750840.

31 Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez, and Mariano Fernández-López. The neon methodology for ontology engineering. In *Ontology engineering in a networked world*, pages 9–34. Springer, 2011. doi:10.1007/978-3-642-24794-1_2.

32 Dylan Van Assche, Thomas Delva, Pieter Heyvaert, Ben De Meester, and Anastasia Dimou. Towards a more human-friendly knowledge graph generation & publication. In *ISWC2021, The International Semantic Web Conference*, volume 2980. CEUR, 2021. URL: https://ceur-ws.org/Vol-2980/paper384.pdf.

Supporting Psychometric Instrument Usage Through the POEM Ontology

Kelsey Rook
Rensselaer Polytechnic Institute, Troy, NY, USA

Henrique Santos
Rensselaer Polytechnic Institute, Troy, NY, USA

Deborah L. McGuinness
Rensselaer Polytechnic Institute, Troy, NY, USA

Manuel S. Sprung
University of California, Los Angeles, CA, USA

Paulo Pinheiro
Instituto Piaget, Lisbon, Portugal

Bruce F. Chorpita
University of California, Los Angeles, CA, USA

Abstract

Psychometrics is the field relating to the measurement of concepts within psychology, particularly the assessment of various social and psychological dimensions in humans. The relationship between psychometric entities is critical to finding an appropriate assessment instrument, especially in the context of clinical psychology and mental healthcare in which providing the best care based on empirical evidence is crucial. We aim to model these entities, which include psychometric questionnaires and their component elements, the subject and respondent, and the latent variables being assessed. The current standard for questionnaire-based assessment relies on text-based distributions of instruments; so, a structured representation is necessary to capture these relationships to enhance accessibility and use of existing measures, encourage reuse of questionnaires and their component elements, and enable sophisticated reasoning over assessment instruments and results by increasing interoperability. We present the design process and architecture of such a domain ontology, the Psychometric Ontology of Experiences and Measures, situating it within the context of related ontologies, and demonstrating its practical utility through evaluation against a series of competency questions concerning the creation, use, and reuse of psychometric questionnaires in clinical, research, and development settings.

2012 ACM Subject Classification Computing methodologies → Ontology engineering; Theory of computation → Semantics and reasoning; Information systems → Ontologies

Keywords and phrases ontology, ontology development, psychometric assessment, psychometric ontology

Digital Object Identifier 10.4230/TGDK.3.3.3

Category Resource

Supplementary Material *Model (Source Code)*: `https://github.com/tetherless-world/POEM` [39]

Funding This work is partially funded by the National Institute of Mental Health "Support for the RCADS Data Collection Measure" project, grant number 75N95022C00018-0-9999-1.

Received 2025-01-21 **Accepted** 2025-10-27 **Published** 2025-12-10

1 Introduction

The field of psychometrics involves the design and evaluation of assessment instruments and associated models to capture complex aspects of mental health and human functioning [20]. A central challenge to the discipline involves the selection and construction of instruments and models that connect unobservable, latent variables, such as mental states and processes, to observable phenomena. While evidence-based medicine has had increasing applications in the improvement of mental health treatment, there is still a lack of attention to what might constitute evidence-based assessment (EBA) [25]. Mental health clinicians are expected to stay informed about accepted standards in psychometric assessment; however, according to Hunsley et al. [26], clinical research and implementation science have focused more on evidence-based intervention, leaving a gap in the area of assessment. While assessment is an integral part of training in graduate psychology programs, relevant guidelines tend to be underdeveloped relative to treatment guidelines; this often results in researchers taking the psychometric properties of their measures for granted [27].

Psychometric instruments and their accompanying handbooks, although informative, lack a standardized representation, hindering the querying, comparison, and reasoning over assessments based on crucial attributes such as length, reliability, validity, and underlying constructs assessed. The selection of appropriate and effective test instruments is pivotal to their successful application in research and service contexts, emphasizing the need for a standardized approach. A vast number of measures exist, but without a knowledge infrastructure to evaluate, organize, and select among them, it is difficult to determine whether a measure is suitable for a specific purpose and context. Many claims in psychometrics research aren't scientifically valid or clinically relevant [45]. More scales are defined each year, and the absence of standardized measurement makes it difficult to combine data. The mental health field is held back by the continuous generation of an inconsistent and underused body of knowledge [17]. So, the ability to harmonize tools for aggregation and comparison would be a powerful tool.

Some existing resources attempt to provide a standard for assessing and distributing testing instruments, such as the Buros Center's Mental Measurements published every three years [8]; however, this resource is expensive to access in PDF or print format. In efforts to make evidence-based assessment more accessible, Becker-Haimes et al. [5] and Beidas et al. [6] have published papers evaluating empirical literature in order to compile lists of assessments that are brief, free, and easily accessible. Although these papers are freely accessible and valuable forms of scholarship, the instruments they review are presented as unstructured data and lack comprehensive information about individual questions, derivations, scales, or scoring methods. Efforts to compile databases for accessibility and comparison, like American Psychological Association (APA) PsycTests, exist, but also fall short. APA PsychTests [42] aims to offer comprehensive information on psychological assessments, but each instrument is primarily characterized within text descriptions lacking standardized structure, which lacks details pertaining to individual items or scales. Clearly, there is an abundance of psychometric resources, but there exists no coordinating infrastructure for their selection, use, or reuse; so, it is very difficult to choose among instruments, and determine whether a specific instrument fits the user's context and aims.

Given the wide variety of instruments and minimal knowledge infrastructure for their use, ontologies emerge as a promising solution to the challenges in this domain. Controlled vocabulary systems and taxonomies for mental health are well-developed within psychiatric nosology, and should be utilized towards this end; however, existing ontologies for assessment lack sufficient support for connecting basic questionnaire elements to these concepts, which they are intended to measure. We propose that ontology can be leveraged to formalize the questionnaire structure, explicate relationships with psychological constructs, track the provenance of questionnaire elements, and assist in aggregating evidence for instrument selection and use. In turn, such an ontology may assist clinical psychologists in the essential task of matching their purpose of testing to the appropriate assessment, increasing understanding of an assessment's construction, administration, scoring, and interpretation.

We introduce the Psychometric Ontology of Experiences and Measures (POEM), which aims to represent crucial aspects of psychometric assessment that may be overlooked by the traditional text-based architecture currently supporting this field. Ontology is a natural progression from the taxonomy and nosology already utilized in the area of mental health, but allows for data management and sophisticated reasoning grounded in prior information. POEM integrates ontological support for psychometric assessment, psychology and mental health domain information, and description of associated entities including questionnaire items, scales, subjects, and respondents. In particular, POEM supports the ability to encode the underlying constructs of measures; that is, information about a questionnaire that is not apparent by examining a plain-text copy of the questionnaire itself. Even more importantly, POEM supports the preservation and propagation of knowledge about the questionnaires as asserted by questionnaire developers. Further, POEM knowledge can be leveraged by future users of questionnaires, specially when questionnaire developers are, for any reason, unavailable to help others on the use of their questionnaires. For example, some questionnaires are used in many countries, and in thousands of service contexts, making human support a large barrier to successfully scaling the application of an otherwise excellent assessment instrument. Similarly, POEM aims to support semantic search of psychometric questionnaires by modeling the underlying constructs and relationships for each entity, moving beyond simple metadata and allowing for complex reasoning.

To the best of our knowledge, POEM is currently the only ontology that tackles the particular issue of modeling questionnaires for clinical assessment, towards the end of deep semantic search, sharing, and reuse at the item, scale, and questionnaire levels. POEM emphasizes the tracking of provenance, enabling and encouraging documentation of the origin and reliability of questionnaire entities that are created from original research, derived from previous entities, or translated from one language to another.

In the following section, we present an overview of related work in the areas of psychological taxonomy and ontology, as well as ontologies that address questionnaires with some consideration of underlying constructs. In Section 3, we introduce the Psychometric Ontology of Experiences and Measures (POEM), including competency questions and the resulting ontology construction process, followed by a description of its architecture in Section 4. In 4.5 we validate the ontology by way of a use case scenario, and conclude with a discussion of our work so far along with plans for future work.

2 Related Work

2.1 Ontologies

Ontologies provide a formally structured way to represent knowledge by defining a shared vocabulary and the relationships between concepts within a particular domain [23] [32]. They enable both humans and machines to consistently interpret and reason over data. Ontologies are often expressed using standardized formats like the Web Ontology Language (OWL) [2], and are commonly used to integrate heterogenous data sources, support interoperability, and enable automated reasoning. Ontologies include definitions of basic concepts in the domain as a finite set of unambiguously identifiable classes and relationships [28]. While taxonomies and ontologies should be distinguished, a very lightweight ontology might also be classified as a taxonomy, and many ontologies contain hierarchies of classes that are based on existing taxonomies of concepts [44].

2.2 Neuropsychological taxonomy and ontology

A number of widely used, comprehensive ontologies and taxonomies exist to give structure to psychological concepts such as diseases, syndromes, signs, symptoms, and assessments; some target the domains of psychology or neuroscience specifically, while others contain neuropsychological concepts within a wider medical context. These ontologies do not represent the internal structure of instruments and questionnaires, but in some cases model the constructs that certain tasks measure. The Systematized Nomenclature of Medicine (SNOMED) is the largest and most widely-used of these; it was created in 1966 as a logic-based healthcare terminology, and has evolved into its current form, SNOMED Clinical Terms (SNOMED CT), which is the most comprehensive healthcare terminology internationally [11]. SNOMED CT contains many psychological concepts, including those subsumed under clinical findings, disorders, and attributes, and acts as an interoperability standard for healthcare information exchange in the United States. Although it is primarily a polyhierarchical taxonomy, SNOMED CT has an ontological foundation and is freely available as an ontology. SNOMED CT can be utilized to handle queries over medical systems in which it is used, due to its support of a simple description logic.

The structure of psychiatric conditions has been formalized for clinical purposes, the most prominent iterations of which are the International Classification of Diseases (ICD), currently on its 11th revision, and the Diagnostic and Statistical Manual of Mental Disorders (DSM), 5th revision [1]. The ICD-11 [33] is a medical taxonomy published by the World Health Organization (WHO); its content encompasses a wide range of health conditions, with a chapter regarding mental, behavioral, and neurodevelopmental disorders. This standard is used internationally, for statistical comparison of morbidity as well as clinical purposes. Currently mappings from ICD-11 codes to the equivalent SNOMED CT concepts are not available, presumably because the ICD-11 is a recent revision and not completely integrated in all settings, but such a map for the ICD-10 is made available by SNOMED International [21]. The DSM-5 [1] is similar in taxonomy to the ICD with key differences: the DSM contains psychiatric diagnoses only, and is based on mental health data and standards from the United States only, limiting its applicability and reach, but it is commonly used in the US for psychiatric diagnoses and treatment recommendations. Although neither is formalized as an ontology, each is an important source of structured mental health information to consider due to their ubiquity in clinical and research settings.

The Cognitive Atlas (CogAt) [36] is a knowledge base for cognitive neuroscience that aims to map mental processes onto brain function, while addressing issues with ambiguous terminology in the field. Additionally, CogAt adopts Wikipedia's approach to collaborative knowledge building, capturing disagreement by allowing public contributions and discussion. CogAt uses basic ontological relations as included in the OBO Relational Ontology, as well as relations between processes, tasks, and "descended from" relations between tasks. The basic classes subsuming all entities in CogAt are Concepts, Tasks, Phenotypes, and specialized Collections of tasks and theories. Although CogAt's hierarchy of Disorders is derived from the Disease Ontology (DO) database [40] and matches most other standard nosologies, CogAt does not aim to be as comprehensive in its inclusion of symptoms.

2.3 Semantic assessment frameworks

There are several existing standards for data exchange in clinical settings. For example, Logical Observation Identifiers Names and Codes (LOINC) [18] provides a common terminology for laboratory and clinical observations for clinical care and management. It has a rich catalog of clinical documents and standardized survey instruments, included psychometric assessments, and is available for public use. Similarly, the Health Level 7 (HL7) clinical document architecture contains many different standards supporting the interoperability of medical data. One such standard is Fast Healthcare Interoperability Resources (FHIR), which provides detailed specifications for document structure including for questionnaires [14].

Several ontologies exist to capture the semantics of measurement instruments; here, we consider those that are close to the goals of POEM either in domain (measurement of constructs in behavioral science, psychology), or in the depth of semantics covered. Beginning in 2010, Batrancourt et al. [3] [4] developed an ontology of Mental State Assessment (ONL-MSA) as an extension of the OntoNeuroLOG ontology [43], a common semantic model of neuroimaging data and tools. ONL-MSA's fundamental contribution is a core ontology providing a general model of mental state assessments alongside a taxonomy of behavioral, neuropsychological, and neuroclinical instrument types. Instrument properties captured include decomposition into sub-instruments, definitions of associated variables, and the domains and qualities measured by instruments. Instruments are associated with a set of measuring actions that link resulting scores to measured variables, and numeric scoring scales are expressed with a set of codes.

In 2012, Cox et al. developed a pair of complementary ontologies to extend the Ontology for General Medical Sciences (OGMS): the Neurological Disease ontology (ND), which aims to be a comprehensive representation of all facets of neuropsychology, and the NeuroPyschological Testing ontology (NPT) [12] [13], a set of classes that represent cognitive functioning assays, including tests, cognitive functions measured, scores, and the associated scale results. NPT and ND also utilize the Mental Disease (MD) and Mental Function (MF) ontologies created by Hastings et al. [24]. Primarily, NPT aims to annotate neuropsychological tests and testing data in order to integrate results over a range of assessments in overlapping domains.

Bensmann et al. have created an ontology with the aims of facilitating data search and reuse for social science survey items [7]. Social surveys, which comprise items usually about attitudes, behaviors, and factual information, are currently findable based on survey-level metadata, but question-level and variable-level searches have previously not been available. Bensmann's ontology tries to remedy this through annotation of question features including the nature of the problem or task given to the respondent, the tone and complexity of language used, and the nature of objects in question, among others.

3 Ontology Development

The initial motivation for POEM stemmed from the need for semi-automated answering of community questions, to relieve the time spent by the administrators of POEM in addressing recurring queries. The development of POEM followed a collaborative, bottom-up approach undertaken by a multidisciplinary team. This team is composed of subject matter experts (SMEs) who are researchers in psychological clinical assessment, led by two doctoral-level researchers working with four additional doctoral-level researchers who specialize in psychological measurement and/or psychometric theory, and four doctoral-level specialists in semantic technologies. This collaborative model allowed for domain knowledge and technical expertise to inform the ontology's design in tandem, ensuring both semantic quality and clinical relevance. POEM was developed iteratively through continuously updated terminology, use case definitions, competency question development, and feedback between SMEs and ontology engineers through weekly meetings. As SMEs gained familiarity with ontology engineering practices, they also contributed refinements to ensure the ontology captured real-world clinical assessment needs.

We conducted a review of previous literature and resources in psychometric representation to create our initial modeling, and identify ontologies that could be reused. This involved a search of key academic databases including Google Scholar and Scopus, and established ontology repositories such as the EMBL-EBI Ontology Lookup Service and NCBO BioPortal. The search strategy utilized a set of keyphrases including but not limited to "mental health ontology", "psychometric ontology", and "mental health data representation". Sources were deemed relevant if they described a formal ontology or data model that could be applied to psychometric assessments, or mental or behavioral health. Next, we established a set of use cases and competency questions, which are elaborated on in Section 3.1. The development process then progressed in three main phases: (1) modeling core questionnaire structure, (2) introducing psychometric constructs and their relations to measurement scales, and (3) incorporating evidence modeling, including provenance tracking to support claims about instrument quality attributes. We followed the Ontology 101 [32] ontology engineering methodology where practical, leveraging its iterative nature and best practices.

Throughout this process, the Revised Child Anxiety and Depression Scale (RCADS) instrument collection served as an anchor for evaluating ontology design and ensuring full use case coverage. Both the original 47-item (RCADS-47) and shortened 25-item (RCADS-25) versions were used. Several contributors to this work are also authors and maintainers of instruments within the RCADS collection, giving increased insight into use-case development and ontology construction. The RCADS instrument family has been widely adopted in the clinical assessment of child and adolescent mental health, and both the RCADS-25 and RCADS-47 have been evaluated in multiple studies. The RCADS-25 is derived from the RCADS-47 using exploratory bifactor analysis to reduce administration time, to be administered at schools or as part of longer test batteries, while maintaining psychometric integrity [16].

Both RCADS versions estimate elevations on multiple clinical dimensions, with scales for depression and anxiety as well as subscales for specific anxiety disorders. Additionally, they are available in multiple versions for child self-report and caregiver report, and have been translated into 31 languages as of the writing of this paper. These factors make the RCADS collection a good candidate for evaluating POEM's ability to support cross-version modeling of questionnaire structure, constructs, and evidence.

3.1 Use Cases and Competency Questions

We have generated a set of use cases, along with competency questions that would be encountered in a broad range of interactions with psychometric assessments, accompanied by example answers based on the full 47-item Revised Child Anxiety and Depression Scale (RCADS-47) [10] and the shortened 25-item version (RCADS-25) [16]. These were generated through regular working sessions that included scenario walkthroughs and expert review, informed by the context of pragmatically supporting RCADS user support service, beginning with questions most frequently received as a starting point. Potential use cases identified are as follows:

- Clinical service: assessment – finding the most appropriate assessment for a specific patient and context based on conditions measured, available languages and norms, provenance, and metrics showing primary utility and exposing strength of evidence; elucidating instrument usage instructions, target subject and respondent, and what an instrument measures
- Clinical service: monitoring – finding the most appropriate assessment for a specific patient and context, which may include reuse of initial questionnaires and scales, or establishing confidence in a shorter assessment for ease of repeated use; elucidating instrument usage instructions, target subject and respondent, and what an instrument measures
- Research: production – determining whether appropriate measures already exist, identifying where reliability and validity may be improved by iterating on an assessment or creating a new one, and supporting the semantic representation of new measures

Table 1 Competency questions per use case.

		Clinical service: assessment	Clinical service: monitoring	Research: production	Research: synthesis	Development: translation
CQ1	What conditions does (questionnaire) measure?	✓	✓	✓	✓	✓
CQ2	How do I score (questionnaire)?	✓	✓	✓		
CQ3	How many/what scales does (questionnaire) have?	✓	✓	✓		
CQ4	What languages are available?	✓	✓	✓		✓
CQ5	Does (questionnaire) require norms?	✓	✓	✓	✓	
CQ6	Does (questionnaire) have relevant norms?	✓	✓		✓	
CQ7	What do the scores actually mean?	✓	✓	✓		
CQ8	Where did the items in (questionnaire) come from?	✓			✓	
CQ9	Who can fill out (questionnaire)?	✓	✓	✓	✓	
CQ10	Are there shorter versions of (questionnaire) available?		✓			
CQ11	Does the short version of (questionnaire) relate to the long version?		✓			
CQ13	How does (questionnaire) relate to other measures in research?			✓		
CQ14	What measures of reliability and validity exist for the RCADS47?			✓		
CQ15	Where can I find out more about the scale and item meanings?					✓

Table 2 Competency questions at Item or Scale level.

CQ16	What concept does (item) represent?
CQ17	If (item) corresponds to a symptom, what syndrome or other corresponding concern does the symptom correspond to?
CQ18	Is the symptom captured by this (item) also captured by other instruments in the world?

- Research: synthesis – supporting the process of combining the results of multiple studies by providing standardized representation of both external and internal questionnaire structure, metadata, and variables
- Development: translation and derivation – determining whether specific language translations of questionnaires exist, as well as their reliability and validity (which may be very different from the original language iteration), determining the minimum design elements needed to create a valid derivative measure, and directing developers towards the provenance and meanings of constructs measured, which should support the accurate translation or adaptation of questionnaires in both linguistic and cultural contexts

Questionnaire-level competency questions can be seen in Table 1, with indication of the particular use cases they are each useful to. Competency questions relevant to specific questionnaire items and scales are relevant across use cases, and can be seen in Table 2.

Based on our competency questions and continual dialogue between the clinical psychology researchers and semantic researchers on our team, we have established a terminology list containing the concepts and definitions that have emerged as important, maintained concurrently with a

Table 3 Ontology prefixes used in POEM.

prefix	ontology	IRI
hasco	Human-Aware Science Ontology	https://hadatac.org/description/ont/hasco
vsto	Virtual Solar Terrestrial Observatory	https://hadatac.org/description/ont/vstoi
sio	Semanticscience Integrated Ontology	https://semanticscience.org/ontology/sio.owl
eco	Evidence and Conclusion Ontology	http://purl.obolibrary.org/obo/eco.owl
stato	The Statistics Ontology	http://purl.obolibrary.org/obo/stato.owl
sco	Study Cohort Ontology	https://purl.org/heals/sco/
poem	Psychometric Ontology of Experiences and Measures	http://purl.org/twc/POEM

UML diagram (Figure 1) representing the structure of POEM as well as its alignment with the foundational ontologies we have chosen to use. Additionally, we developed prototype applications that exposed whether the entities and relationships modeled were sufficient to enable tools serviced by user-relevant data structures, continually iterating between POEM revisions and the design of these applications; particularly, tools associated with the RCADS website for accessing information on the RCADS assessment instruments.

3.2 Implementation

The main goal of POEM is to provide a logical and systematic description of measures in clinical psychology. Based on this goal, as well as terminology and use case documentation, we constructed the POEM ontology focusing first on basic questionnaire structure, then underlying constructs, and finally, evidence modeling, with continuous iteration based on feedback from domain experts. Throughout this process we have used the RCADS-25 and RCADS-47 as examples with which to evaluate our progression, while ensuring that the structure defined by the terms and relations in POEM generalize to other measures in clinical psychology.

POEM is implemented in OWL/RDF [29], with the help of the open-source ontology editor Protégé [30]. We maintain POEM using GitHub at `https://github.com/tetherless-world/POEM`, and documentation including use case deliverables, terminology, a static demo, and publications at `https://tetherless-world.github.io/POEM/`.

4 POEM

The most central classes to POEM can be seen in Table 4, along with their source ontologies, definitions, and parent classes.

4.1 Terminology Reuse

POEM utilizes several well-established ontologies at different domain levels in order to effectively build off of previous work and provide inherent modularity in its usage. Prefixes used for modules of the ontology are shown in Table 3.

4.1.1 Semanticscience Integrated Ontology (SIO)

The POEM ontology is built around the hierarchy provided by the Semanticscience Integrated Ontology (SIO) [15], an upper-level ontology that supports the description of objects, processes, and attributes needed to facilitate biomedical data discovery. SIO defines objects (sio:Object) as entities that have spatial components and identifiably persistent characteristics, while processes (sio:Process) are entities with a temporal element. Attributes (sio:Attribute) are qualities, capabilities, or roles that can describe some other entity. On top of this simple structure there is a large hierarchy with particular focus in the biomedical domain, and relations that can describe entities in terms of spatial organization, process flow, and referential relations. SIO offers a practical applied vocabulary that facilitates the integration of scientific data, with simple design patterns that suit the needs of POEM for the descripion of informational and biomedical entities.

4.1.2 Human-Aware Science Ontology (HAScO)

The Human-Aware Science Ontology (HAScO) [34] is designed to describe scientific data, supporting its related activities, such as data acquisition and scientific studies, in a way that applies to a diverse range of domains. HAScO encompasses three particular areas: scientific activities for data acquisition, data schema, and instruments, with reliance on VSTO-I (4.1.3). Through the use of Semantic variables (hasco:SemanticVariable) [35], POEM aligns with the capability of semantically describing scientific variables, enabling data annotation of datasets that contain data acquired through the application of questionnaires.

4.1.3 Virtual Solar Terrestrial Observatory– Instruments (VSTO-I)

The Virtual Solar-Terrestrial Observatory (VSTO) [19] is a semantic data framework requiring formal representations of physical quantities and their underlying representations. Originally intended to support observatory projects across various physics subfields, VSTO has been expanded to more generically support scientific instruments (vstoi:Instrument) with VSTO-I (the instrumentation portion of VSTO). VSTO-I has support for questionnaires (vstoi:Questionnaire) and items (vstoi:Item) on top of some of its most basic components:

- vstoi:Detector: A device that detects measurements; *items* (vstoi:Item) are *detectors* of signals representing constructs through the recording of human response to some prompt
- vstoi:Instrument: A device that receives attribute measurements from detectors, processing them into a useful value/s; *questionnaires* (vstoi:Questionnaire) are *instruments* that translate a set of responses to *items* into one or more scores

4.1.4 Evidence and Conclusion Ontology (ECO)

The Evidence and Conclusion Ontology (ECO) [31] is intended to capture annotations and evidence to support biomedical assertions. The main root class, Evidence (eco:evidence), is defined as being the output of some planned evidence-gathering process, or assay, as well as producing some conclusion based on data. ECO also defines automatic and manual assertion methods, which, alongside evidence classes, allow complex statements about the evidence gathering process. ECO was created by the founders of the Gene Ontology (GO) and so contains precise evidence types of molecular, cellular, and biological natures; however, the comprehensive hierarchy of evidence types, including computational, experimental, inferential, and similarity evidences, easily generalizes to psychometric usage.

4.1.5 Statistics Ontology (STATO)

The Statistics Ontology (STATO) [22] is a general purpose ontology based on the Basic Formal Ontology (BFO) [41], an upper-level ontology for scientific research. STATO covers processes involved in statistical analysis, including statistical tests, the input and output information from these tests, and aspects of experimental design; additionally, these statistical analysis concepts are related to different study designs. STATO supports the application of statistical tests, generation of results and reports, and communication of scientific results. Most notably to its usage in POEM, STATO defines study group populations and cohorts, as well as statistical data items that are commonly used to assess the reliability and validity of psychometric tools.

4.1.6 Study Cohort Ontology (SCO)

The Study Cohort Ontology [9] addresses challenges faced when matching patient populations to study cohort characteristics during the process of generating treatment recommendations within Clinical Practice Guidelines (CPGs). The primary focus of SCO is clinical trials that have study and control arms, but can be generalized to other types of cohort studies, including observational studies used to evaluate psychometric assessments. Primarily, SCO encodes the vocabulary needed to describe study populations, including study subjects, subject characteristics, and accompanying statistical measures. SCO supports a workflow that allows practitioners to perform population analysis, visualize cohort similarities, and derive clinically relevant inferences.

Table 4 Primary POEM concepts.

Source	Class	Definition	Parent
POEM	Construct	A hypothetical or theoretical entity that can not be directly observed	`sio:Entity`
	Psychometric Questionnaire	a questionnaire used to measure an individual's mental capabilities, behaviors, or psychological traits	`vstoi:Questionnaire`
	Questionnaire Scale	a collection of indicators designed to be related to a shared construct	`sio:Object`
	Composite Scale	a scale containing two or more subscales	`poem:QuestionnaireScale`
	Experience	An informant's observation or encounter in a situation or event	`sio:Quality`
	Item Stem Concept	The experience or construct that the question or statement used to prompt an individual intends to capture	`sio:Object`
	Instrument Family	A set of measurement instruments that have been derived from each other, are published by the same organization, or have some other reason for close association	sio:Collection
HASCO	Semantic Variable	A variable specification that includes the target entities and attributes, but not the population property	`owl:Thing`
VSTO-I	Detector	A device which detects measurements	`sio:Device`

Instrument	A device or mechanism that is used to acquire attribute values of entities of interest	`sio:Device`
Questionnaire	An instrument used to acquire data reported from human subjects	`vstoi:Instrument`
Item	An item stem and its response option within an assessment	`vstoi:Detector`
Item Stem	A question or statement used to prompt the individual to provide information regarding a latent variable	`sio:Object`
Scale	A collection of indicators designed to be related to a shared construct	`vstoi:Instrument`
Codebook	A document used to outline the content, format, and coding scheme of a dataset	`sio:Entity`
Response Option	A possible answer choice provided for a question or statement	`sio:Object`
Informant	An individual or respondent who provides information based on an observation about themselves or someone else	`sio:Person`

We now describe the structure of the POEM ontology, describing in more detail how elements of the ontologies summarized in section 4.1 are utilized, with references to classes in the POEM ontology italicized for clarity.

4.2 Questionnaire Structure

The core concept of POEM is the *psychometric questionnaire*, which inherits the majority of its structural concepts from *questionnaire* (vstoi:Questionnaire). The *psychometric questionnaire* concept inherits provenance attributes such as authorship, licensing, intended subject and respondent, derivation, and structural attributes such as instructions. Additionally, the questionnaire *item* (vstoi:Item) encapsulates several entities and other pieces of information included in an assessment question: an *item stem*, which is the text presented in the context of prompting a response regarding some latent construct; the *item stem concept*, which is the specific phenomenon an *item* is intended to capture; and a *codebook* (vstoi:Codebook) representing a range of possible respondent *experiences*. *Items* have instrument membership attributes inherited from HAScO specification, giving *items* membership and position within any number of *questionnaires*.

4.3 Underlying Constructs

The primary focus of POEM lies in the representation of the underlying semantics of psychometric questionnaires. This modeling extends from the foundational structure established in VSTO-I, representing the *constructs* (poem:Construct) that questionnaires assess. A psychometric construct is a phenomenon whose signal is meant to be detected by an *item* or *questionnaire scale*.

An *item* is an indicator that targets a signal representing a particular *construct*, using the recorded human response. More precisely, an *item stem* is associated with an *item stem concept*, which is an expression of the specific construct or experience intended to be captured. Further, *questionnaire scales* (poem:QuestionnaireScale) comprise a collection of one or more *item stem concepts* designed to measure a shared *construct.* The set of *item stem concepts* in a *scale* that has been shown as reliable and valid is said to accurately estimate the presence of a *construct.*

To support a consistent view of variables for data unification, HAScO uses the notion of a *semantic variable*[35]. Formalized variable specifications support the alignment and combination of variables in processes that occur across multiple studies. In particular, the *semantic variable* includes *entity* and *attribute* properties, but no population property. Accordingly, two variables share a common *semantic variable* when the only distinction between them is their population properties. HAScO specifies that *semantic variables* are measured by *detectors*, and are also associated with attributes such as temporal and spatial information, and measurement unit. In POEM, the *semantic variable* detected by an *item* has, for its attribute, some *symptom*, and for its entity, a human of some demographic. In this way, questionnaire data can be aligned within and across cohorts.

POEM introduces the *experience* class. Each *codebook* corresponds with an *experience* related to the subject's personal degree to which they experience a *construct* measured by a questionnaire *item*; for example, frequency or intensity.

We maintain that POEM should remain neutral with respect to frameworks such as the ICD or DSM, and recognize that these and related frameworks are not uncontroversial. Further, while POEM provides the scaffolding for linking questionnaire scales to clinical constructs, it does not limit these constructs to those in a particular nosology. The axiom <poem:QuestionnaireScale *isAbout* poem:Construct> is intentionally generalizable, allowing integration with any framework, provided that any object of this axiom is reasonable subsumed by the *construct* concept. In our evaluation, we use SNOMED entities to represent the concepts measured, because of its wide use and coverage, and mappings to other terminologies.

4.4 Evidence Modeling

Determining if psychometric tests are adequate to assess psychological constructs is a process that is crucial to successful clinical assessment. POEM supports these processes by modeling population-based *studies* (SIO:001041) that are conducted by researchers to determine whether features of questionnaires measure the *construct* they are intended to, including metrics such as reliability and validity. The study group population that meets some criteria for inclusion in a study is referred to as a *cohort* (hasco:cohort). Knowing the context in which a measure is applied can be important to evaluating its importance. We use SCO, which connects the *cohort* class to demographic information, effectively connecting studies to the context in which they were conducted.

Observational studies involve the administration of created measures to a *cohort* one or more times depending on the target metric; for example, measuring test-retest reliability requires comparison between scores within the same cohort over time. Questionnaires such as the Caregiver version of the RCADS-47 require the inclusion of two *cohorts*, since the respondent (caregiver) and subject (youth) are different people. Each metric produced as the result of a *study* constitutes a piece of *evidence* (ECO:Evidence), which supports some assertion about an item or scale of the instrument in question. For example, in an initial study of the RCADS-47 [10], the value of Cronbach's Alpha (α) calculated for the Social Phobia scale was $\alpha = 0.82$. This value of α is generally considered to be good; so, the Cronbach's Alpha coefficient for the Social Phobia scale is a piece of observational study evidence supporting the assertion that the Social Phobia Scale

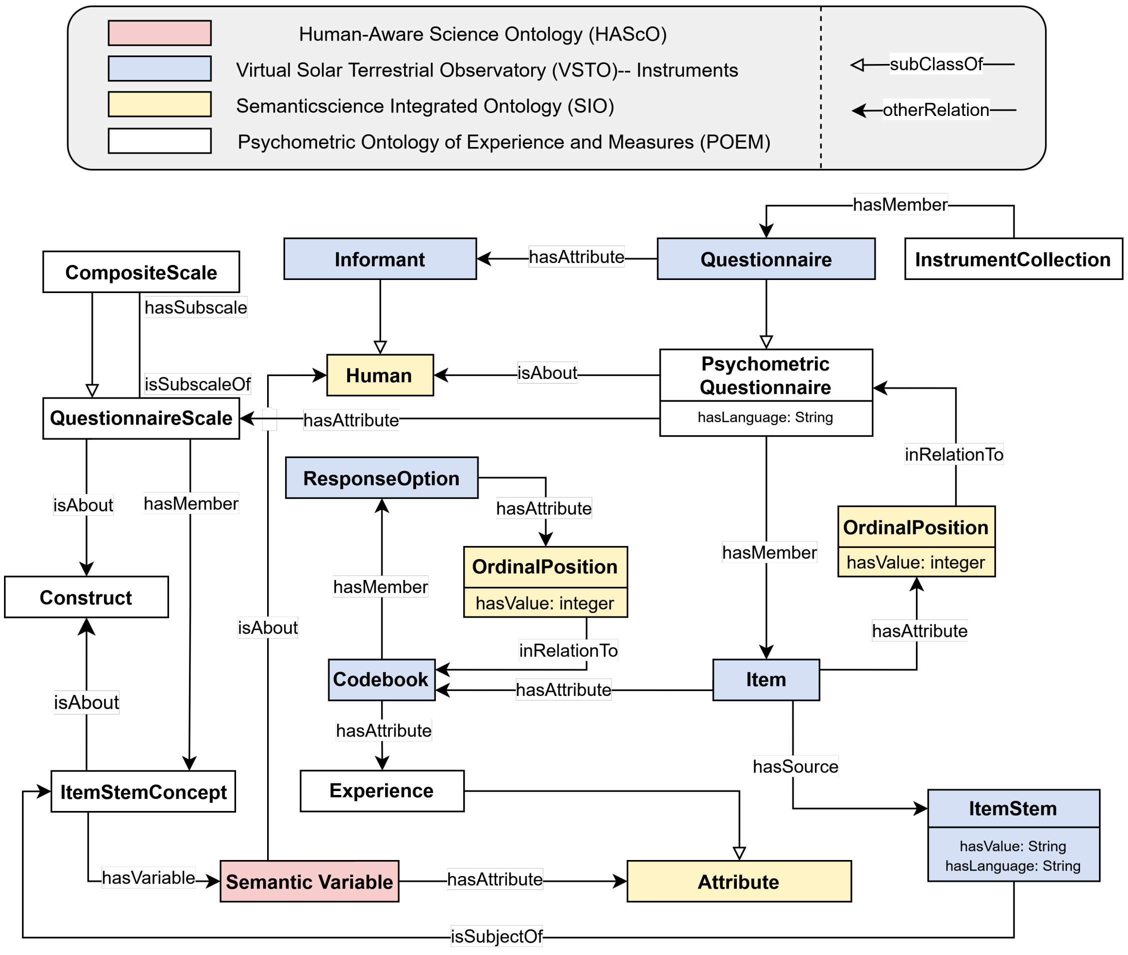


Figure 1 Conceptual diagram of the POEM ontology.

has good internal consistency for the cohort represented. The modeling described can be seen in Figure 2. This formalization of studies and evidence supports the aggregation of metrics about a questionnaire and its features, allowing clinicians to choose questionnaires that adequately fit their use case.

We avoid making claims about what standards constitute "good evidence"; for example, while a value of $\alpha = 0.82$ is generally considered good, there is no consistent criteria for determining assessment quality, and there are certain ways in which numerical indicators can be misleading [38].

4.5 Use Case Scenario

We demonstrate POEM's utility in psychometrics via the competency questions posed early in the development process (table 1, table 2), using the RCADS-47 and RCADS-25 as subjects. We revisit the use cases outlined in Section 3.1 along with applicable competency questions, based on knowledge graphs representing the 47- and 25-item versions of the RCADS. For each competency question, we deploy SPARQL to query the complete knowledge graphs.

Table 5 revisits the competency questions posed in tables 1 and 2, along with specific answers based on the RCADS-47 and RCADS-25. Table 5 also shows how the current version of POEM can support the answering of a subset of our original competency questions through SPARQL queries, generating results that align with SME-generated answers. The SMEs reviewed the

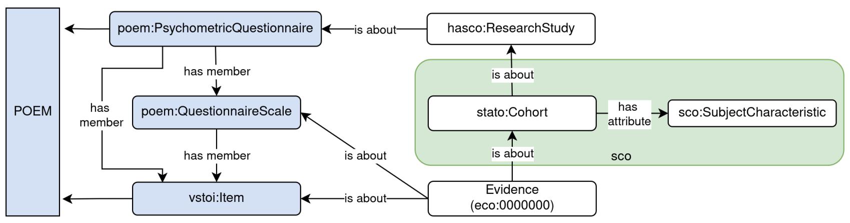


Figure 2 Evidence Modeling in POEM.

ontology's structure and its ability to provide meaningful responses, through demonstration of queries that could sufficiently answer the generated competency questions dynamically, provided by the ontology experts. Overall, SMEs expressed a high degree of satisfaction with the ontology's coverage and utility.

5 Future Work

The primary focus of our work has been to formalize assessment concepts to support measurement of disorders in a clinical setting. Ongoing work focuses on enhancing POEM's utility through generalization to broader types of psychometric questionnaires. Key extensions of POEM include the modeling of scoring instructions, and continued work on evidence modeling. This includes the formal representation of norms, which are the sets of answers from test-takers within specific groups. They may be used during the scoring process to determine the relative standing of a subject within a group sharing their attributes, generating a percentile or other normative score. The ability to integrate different norms for the same test across various demographics can support various applications such as an automatic scoring tool that leverages ontology linkages, which is currently at the advanced prototyping stage. We also plan to model additional details of provenance, such as linking of questionnaires and components to publications for further details and evidentiary support, and providing authorship details.

POEM is being used to support the Semantic Instrument Repository (SIR), a software infrastructure for the management and distribution of knowledge graphs about data acquisition instruments, particularly for mental health screening. SIR is an ongoing project and collaborative effort that is open source and freely available to the public, with a Drupal module client component and a Java API server component with access to a FUSEKI triple-store repository. The usage of POEM enables powerful semantic search and retrieval of published instruments and elements, as well as tracking the evolution and reuse of questionnaires. Users at the appropriate level of authorization will be able to draft, publish, edit, and deprecate questionnaires.

Additionally, SIR uses POEM to enable instrument rendering and sharing in formats such as XML/OWL, Turtle, and JSON, and uses POEM knowledge graphs to map from canonical descriptions into rendering tools like REDCap and FHIR. SIR also utilizes semantic data dictionaries [37], a standard for semantic representation of data, to formally represent acquired data. SIR also supports the rendering of POEM questionnaires into human-readable formats such as PDF. SIR is currently under development and not yet deployed, but future work is planned.

6 Conclusion

To address the lack of knowledge infrastructure currently supporting psychometric assessment, we have designed the POEM ontology, aligning it with SIO and other high-level ontologies. We have described how POEM formalizes the structure of questionnaires and the constructs they measure, aligning with terminology used in psychology and psychometrics, and shown how, given a knowledge graph of an assessment, POEM can answer a range of queries about content and provenance within clinical and research use cases. Following discussion of difficulties faced due to lack of formal guidance in proper selection and usage of instruments, we see value in continued development of POEM as research drives an increasingly diverse proliferation of assessment instruments.

References

1 DSMTF American Psychiatric Association, American Psychiatric Association, et al. *Diagnostic and statistical manual of mental disorders: DSM-5*, volume 5. American psychiatric association Washington, DC, 2013. doi:10.1176/appi.books.9780890425596.

2 Grigoris Antoniou and Frank van Harmelen. Web ontology language: Owl. *Handbook on ontologies*, pages 91–110, 2009. doi:10.1007/978-3-540-92673-3_4.

3 Bénédicte Batrancourt, Michel Dojat, Bernard Gibaud, and Gilles Kassel. A core ontology of instruments used for neurological, behavioral and cognitive assessments. In *FOIS*, pages 185–198, 2010. doi:10.3233/978-1-60750-535-8-185.

4 Bénédicte Batrancourt, Michel Dojat, Bernard Gibaud, and Gilles Kassel. A multilayer ontology of instruments for neurological, behavioral and cognitive assessments. *Neuroinformatics*, 13:93–110, 2015. doi:10.1007/s12021-014-9244-3.

5 Emily M Becker-Haimes, Alexandra R Tabachnick, Briana S Last, Rebecca E Stewart, Anisa Hasan-Granier, and Rinad S Beidas. Evidence base update for brief, free, and accessible youth mental health measures. *Journal of Clinical Child & Adolescent Psychology*, 49(1):1–17, 2020.

6 Rinad S Beidas, Rebecca E Stewart, Lucia Walsh, Steven Lucas, Margaret Mary Downey, Kamilah Jackson, Tara Fernandez, and David S Mandell. Free, brief, and validated: Standardized instruments for low-resource mental health settings. *Cognitive and behavioral practice*, 22(1):5–19, 2015. doi:10.1016/j.cbpra.2014.02.002.

7 Felix Bensmann, Andrea Papenmeier, Dagmar Kern, Benjamin Zapilko, and Stefan Dietze. Semantic annotation, representation and linking of survey data. In *Semantic Systems. In the Era of Knowledge Graphs: 16th International Conference on Semantic Systems, SEMANTiCS 2020, Amsterdam, The Netherlands, September 7–10, 2020, Proceedings 16*, pages 53–69. Springer International Publishing, 2020. doi:10.1007/978-3-030-59833-4_4.

8 Janet F Carlson, Kurt F Geisinger, Jessica L Jonson, and Nancy A Anderson. The twenty-first mental measurements yearbook. *(No Title)*, 2021.

9 Shruthi Chari, Miao Qi, Nkechinyere N Agu, Oshani Seneviratne, Jamie P McCusker, Kristin P Bennett, Amar K Das, and Deborah L McGuinness. Making study populations visible through knowledge graphs. In *International Semantic Web Conference*, pages 53–68. Springer, 2019. doi:10.1007/978-3-030-30796-7_4.

10 Bruce F Chorpita, Letitia Yim, Catherine Moffitt, Lori A Umemoto, and Sarah E Francis. Assessment of symptoms of dsm-iv anxiety and depression in children: A revised child anxiety and depression scale. *Behaviour research and therapy*, 38(8):835–855, 2000. doi:10.1016/S0005-7967(99)00130-8.

11 Ronald Cornet and Nicolette de Keizer. Forty years of snomed: a literature review. *BMC medical informatics and decision making*, 8(1):1–6, 2008. doi:10.1186/1472-6947-8-S1-S2.

12 Alexander P Cox, Mark Jensen, William Duncan, Bianca Weinstock-Guttman, Kinga Szigeti, Alan Ruttenberg, Barry Smith, and Alexander D Diehl. Ontologies for the study of neurological disease. *Third International Conference on Biomedical Ontology*, 2012.

13 Alexander P Cox, Mark Jensen, Alan Ruttenberg, Kinga Szigeti, and Alexander D Diehl. Measuring cognitive functions: Hurdles in the development of the neuropsychological testing ontology. In *ICBO*, pages 78–83. Citeseer, 2013. URL: https://ceur-ws.org/Vol-1060/icbo2013_submission_46.pdf.

14 Robert H Dolin, Liora Alschuler, Sandy Boyer, Calvin Beebe, Fred M Behlen, Paul V Biron, and Amnon Shabo. Hl7 clinical document architecture, release 2. *Journal of the American Medical Informatics Association*, 13(1):30–39, 2006. doi:10.1037/1040-3590.17.3.251.

15 Michel Dumontier, Christopher JO Baker, Joachim Baran, Alison Callahan, Leonid Chepelev, José Cruz-Toledo, Nicholas R Del Rio, Geraint Duck, Laura I Furlong, Nichealla Keath, et al. The semanticscience integrated ontology (sio) for biomedical research and knowledge discovery. *Journal of biomedical semantics*, 5:1–11, 2014. doi:10.1186/2041-1480-5-14.

16 Chad Ebesutani, Steven P Reise, Bruce F Chorpita, Chelsea Ale, Jennifer Regan, John Young, Charmaine Higa-McMillan, and John R Weisz. The revised child anxiety and depression scale-short version: scale reduction via exploratory bifactor modeling of the broad anxiety factor. *Psychological assessment*, 24(4):833, 2012.

17 Gregory K Farber, Suzanne Gage, Danielle Kemmer, and Rory White. Common measures in mental health: a joint initiative by funders and journals. *The Lancet Psychiatry*, 10(6):465–470, 2023. doi:10.1016/S2215-0366(23)00139-6.

18 Arden W Forrey, Clement J Mcdonald, Georges DeMoor, Stanley M Huff, Dennis Leavelle, Diane Leland, Tom Fiers, Linda Charles, Brian Griffin, Frank Stalling, et al. Logical observation identifier names and codes (loinc) database: a public use set of codes and names for electronic reporting of clinical laboratory test results. *Clinical chemistry*, 42(1):81–90, 1996. doi:10.1093/clinchem/42.1.81.

19 Peter Fox, Deborah L McGuinness, Luca Cinquini, Patrick West, Jose Garcia, James L Benedict, and Don Middleton. Ontology-supported scientific data frameworks: The virtual solar-terrestrial observatory experience. *Computers & Geosciences*, 35(4):724–738, 2009. doi:10.1016/j.cageo.2007.12.019.

20 R Michael Furr. *Psychometrics: an introduction.* SAGE publications, 2021.

21 Kathy L Giannangelo and Jane Millar. Mapping snomed ct to icd-10. In *MIE*, pages 83–87, 2012. doi:10.3233/978-1-61499-101-4-83.

22 Alejandra Gonzalez-Beltran. Statistics ontology. URL: http://purl.obolibrary.org/obo/stato.owl.

23 Thomas R Gruber. A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2):199–220, 1993. doi:10.1006/knac.1993.1008.

24 Janna Hastings, Werner Ceusters, Mark Jensen, Kevin Mulligan, and Barry Smith. Representing mental functioning: Ontologies for mental health and disease. *Third International Conference on Biomedical Ontology*, 2012.

25 John Hunsley and Eric J Mash. Introduction to the special section on developing guidelines for the evidence-based assessment (eba) of adult disorders. *Psychological assessment*, 17(3):251, 2005.

26 John Hunsley and Eric J Mash. Evidence-based assessment. *Annu. Rev. Clin. Psychol.*, 3:29–51, 2007. doi:10.1093/oxfordhb/9780199328710.013.019.

27 Scott O Lilienfeld and Adele N Strother. Psychological measurement and the replication crisis: Four sacred cows. *Canadian Psychology/Psychologie Canadienne*, 61(4):281, 2020. doi:10.1037/cap0000236.

28 Deborah L McGuinness. Ontologies come of age. In *Spinning the semantic web*, pages 171–194, 2003. doi:10.7551/mitpress/6412.003.0008.

29 Deborah L McGuinness, Frank Van Harmelen, et al. Owl web ontology language overview. *W3C recommendation*, 10(10):2004, 2004.

30 Mark A Musen. The protégé project: a look back and a look forward. *AI matters*, 1(4):4–12, 2015. doi:10.1145/2757001.2757003.

31 Suvarna Nadendla, Rebecca Jackson, James Munro, Federica Quaglia, Bálint Mészáros, Dustin Olley, Elizabeth T Hobbs, Stephen M Goralski, Marcus Chibucos, Christopher John Mungall, et al. Eco: the evidence and conclusion ontology, an update for 2022. *Nucleic acids research*, 50(D1):D1515–D1521, 2022. doi:10.1093/nar/gkab1025.

32 Natalya F Noy, Deborah L McGuinness, et al. Ontology development 101: A guide to creating your first ontology, 2001.

33 World Health Organization. *International statistical classification of diseases and related health problems*, volume 11. World Health Organization, 2019.

34 Paulo Pinheiro, Marcello Peixoto Bax, Henrique Santos, Sabbir Rashid, Zhicheng Liang, Yue Liu, James Mccusker, Deborah Mcguinness, and Yarden Ne'eman. Annotating diverse scientific data with hasco. In *Seminar on Ontology Research in Brazil.* Universidade Federal de Minas Gerais, 2018.

35 Paulo Pinheiro, Henrique Santos, Miao Qi, Kristin P Bennett, and Deborah L McGuinness. Towards machine-assisted biomedical data preparation: A use case on disparity in access to health care. *6th International Workshop on Semantic Web solutions for large-scale biomedical data analytics*, 2023.

36 Russell A Poldrack, Aniket Kittur, Donald Kalar, Eric Miller, Christian Seppa, Yolanda Gil, D Stott Parker, Fred W Sabb, and Robert M Bilder. The cognitive atlas: toward a knowledge foundation for cognitive neuroscience. *Frontiers in neuroinformatics*, 5:17, 2011. doi:10.3389/fninf.2011.00017.

37 Sabbir M Rashid, James P McCusker, Paulo Pinheiro, Marcello P Bax, Henrique O Santos, Jeanette A Stingone, Amar K Das, and Deborah L McGuinness. The semantic data dictionary–an approach for describing and annotating data. *Data intelligence*, 2(4):443–486, 2020. doi:10.1162/dint_a_00058.

38 Andres De Los Reyes and David A Langer. Assessment and the journal of clinical child and adolescent psychology's evidence base updates series: Evaluating the tools for gathering evidence. *Journal of Clinical Child & Adolescent Psychology*, 47(3):357–365, 2018. doi:10.1080/15374416.2018.1458314.

39 Kelsey Rook, Henrique Santos, Deborah L. McGuinness, Manuel S. Sprung, Paulo Pinheiro, and Bruce F. Chorpita. POEM Ontology. Model (visited on 2025-12-08). URL: https://github.com/tetherless-world/POEM, doi:10.4230/artifacts.25228.

40 Lynn M Schriml, Elvira Mitraka, James Munro, Becky Tauber, Mike Schor, Lance Nickle, Victor Felix, Linda Jeng, Cynthia Bearer, Richard Lichenstein, et al. Human disease ontology 2018 update: classification, content and workflow expansion. *Nucleic acids research*, 47(D1):D955–D962, 2019. doi:10.1093/nar/gky1032.

41 Barry Smith, Anand Kumar, and Thomas Bittner. Basic formal ontology for bioinformatics. *IFOMIS reports*, 2005.

42 Susan E Swogger. Psyctests. *Journal of the Medical Library Association: JMLA*, 101(3):234, 2013. doi:10.3163/1536-5050.101.3.021.

43 Lynda Temal, Michel Dojat, Gilles Kassel, and Bernard Gibaud. Towards an ontology for sharing medical images and regions of interest in neuroimaging. *Journal of Biomedical Informatics*, 41(5):766–778, 2008. doi:10.1016/j.jbi.2008.03.002.

44 Reinout Van Rees. Clarity in the usage of the terms ontology, taxonomy and classification. *Cib Report*, 284(432):1–8, 2003.

45 Eric A Youngstrom, Sophia Choukas-Bradley, Casey D Calhoun, and Amanda Jensen-Doss. Clinical guide to the evidence-based assessment approach to diagnosis and treatment. *Cognitive and Behavioral Practice*, 22(1):20–35, 2015. doi:10.1016/j.cbpra.2013.12.005.

A Competency Questions and SPARQL Queries

Ontology prefixes:

```
PREFIX poem: <http://purl.org/poem#>
PREFIX rcads: <http://purl.org/poem/individuals#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dc: <http://purl.org/dc/terms/>
```

Table 5 Competency Questions and Corresponding SPARQL Queries.

#	Question	Answer
CQ1	What conditions does the RCADS-47 measure?	separation anxiety disorder, social phobia, generalized anxiety disorder, panic disorder, obsessive compulsive disorder, major depressive disorder

	`SELECT ?condition (STR(?lab) AS ?label) WHERE {`
	`  rcads:RCADS47Questionnaire sio:hasMember ?scale .`
	`  ?scale rdf:type poem:QuestionnaireScale .`
	`  ?scale sio:isAbout ?construct .`
	`  ?construct rdfs:label ?label .`
	`}`	

CQ3	How many/what scales does the RCADS-47 have?	8 scales; separation anxiety disorder, social phobia, generalized anxiety disorder, panic disorder, obsessive compulsive disorder, major depressive disorder, total anxiety, total anxiety and depression

	`SELECT ?subscaleLabel`
	`WHERE {`
	`  rcads:RCADS47Questionnaire sio:hasMember ?scale .`
	`  ?scale rdf:type poem:QuestionnaireScale .`
	`}`	

CQ4	What languages are available for the RCADS-47?	US English, Chinese, Danish, Dutch, Finnish, French, German, Greek, Icelandic, Japanese, Korean, Norwegian, Persian, Polish, Slovene, Portuguese, Spanish, Swedish, Turkish, Urdu

	`SELECT ?languageCode`
	`WHERE {`
	`  rcads:RCADS47Questionnaire sio:hasMember ?item .`
	`  ?item rdf:type vstoi:Item .`
	`  ?item sio:hasSource ?itemStem .`
	`  ?itemStem dc:language ?languageCode .`
	`}`	

CQ9	Who can fill out the RCADS-47?	Youth (8-18 years), caregiver of youth (8-18 years)

	`SELECT ?informantLabel`
	`WHERE {`
	`  rcads:RCADS47Questionnaire sio:hasAttribute ?informant .`
	`  ?informant rdf:type poem:Informant .`
	`  ?informant rdfs:label ?informantLabel .`
	`}`	

CQ10	Are there shorter versions of the RCADS-47 available?	Yes; the RCADS-25

```
SELECT ?questionnaireLabel ?itemCount
WHERE {
  {
    SELECT ?questionnaire (COUNT(?item) AS ?itemCount)
    WHERE {
      ?questionnaire sio:hasMember ?item .
    }
    GROUP BY ?questionnaire
  }
  {
    SELECT (COUNT(?rcads47item) AS ?rcads47itemCount)
    WHERE {
      rcads:RCADS47Questionnaire sio:hasMember ?rcads47item .
    }
  }
  FILTER (?itemCount < ?rcads47itemCount)
} ORDER BY ?itemCount
```

CQ11	Does the RCADS-25 relate to the RCADS-47?	Yes; the RCADS-25 is an abbreviated version of the RCADS-47, and contains 25 items also in the RCADS-47.

```
SELECT (COUNT(?item) AS ?overlapCount)
WHERE {
  rcads:RCADS47questionnaire sio:hasMember ?item .
  rcads:RCADS25questionnaire sio:hasMember ?item .
}
```

CQ14	What measures of reliability and validity exist for the RCADS47?	Several studies have found the RCADS-47 to be a reliable and valid measure of children's anxiety and depression (for example: Chorpita 2000, Ebesutani 2012.)

```
SELECT ?scale ?reliabilityMeasure ?reliabilityValue \\
                ?validityMeasure ?validityValue
WHERE {
  rcads:RCADS47Questionnaire sio:hasMember ?scale .
  ?scale a poem:questionnaireScale .

  ?evidence a eco:0000000 .  # eco:0000000 = Evidence
  ?evidence sio:isAbout ?scale .

OPTIONAL {
  ?evidence a poem:reliabilityFinding .
}
OPTIONAL {
  ?evidence a poem:validityFinding .
}
FILTER(BOUND(?reliabilityFinding) || BOUND(?validityFinding))
}
```

CQ16	What concept does Item 1 ("I worry about things") of the RCADS-47 represent?	worry (generalized)

	`SELECT ?conceptLabel WHERE {`
	`rcads:item/1 sio:hasSource ?itemStem .`
	`?itemStem sio:isAbout ?itemStemConcept .`
	`?itemStemConcepts sio:isAbout ?construct .`
	`?construct a poem:construct .`
	`}`	

CQ17	If RCADS-47 Item 1 corresponds to a symptom, what condition does the symptom correspond to?	generalized anxiety disorder

	`SELECT ?condition WHERE {`
	`rcads:itemStemConcept/1 sio:isAbout ?symptom .`
	`?symptom a poem:construct .`
	`?scale sio:hasMember rcads:itemStemConcept/1 .`
	`?scale a poem:questionnaireScale .`
	`?scale sio:isAbout ?condition .`
	`}`	

Mining Inter-Document Argument Structures in Scientific Papers for an Argument Web

Florian Ruosch
University of Zurich, Switzerland

Cristina Sarasua
University of Zurich, Switzerland

Abraham Bernstein
University of Zurich, Switzerland

Abstract

In Argument Mining, predicting argumentative relations between texts (or spans) remains one of the most challenging aspects, even more so in the cross-document setting. This paper makes three key contributions to advance research in this domain. We first extend an existing dataset, the Sci-Arg corpus, by annotating it with explicit inter-document argumentative relations, thereby allowing arguments to be distributed over several documents forming an Argument Web; these new annotations are published using Semantic Web technologies (RDF, OWL). Second, we explore and evaluate three automated approaches for predicting these inter-document argumentative relations, establishing critical baselines on the new dataset. We find that a simple classifier based on discourse indicators with access to context outperforms neural methods. Third, we conduct a comparative analysis of these approaches for both intra- and inter-document settings, identifying statistically significant differences in results that indicate the necessity of distinguishing between these two scenarios. Our findings highlight significant challenges in this complex domain and open crucial avenues for future research on the Argument Web of Science, particularly for those interested in leveraging Semantic Web technologies and knowledge graphs to understand scholarly discourse. With this, we provide the first stepping stones in the form of a benchmark dataset, three baseline methods, and an initial analysis for a systematic exploration of this field relevant to the Web of Data and Science.

2012 ACM Subject Classification Computing methodologies → Information extraction; Computing methodologies → Language resources; Computing methodologies → Semantic networks; Information systems → Graph-based database models

Keywords and phrases Argument Mining, Large Language Models, Knowledge Graphs, Link Prediction

Digital Object Identifier 10.4230/TGDK.3.3.4

Category Research

Supplementary Material
Software (System): `https://gitlab.ifi.uzh.ch/DDIS-Public/midas/-/tree/main/bam`
archived at `swh:1:dir:b09e3e98962964553b846eda4014a48333f989b1`
Dataset: `https://gitlab.ifi.uzh.ch/DDIS-Public/midas/-/tree/main/data`
archived at `swh:1:dir:f8a081776dcaea3fd18c44d3dd0e8b9b4cd83800`
Software (Evaluation): `https://gitlab.ifi.uzh.ch/DDIS-Public/midas/-/tree/main/evaluation`
archived at `swh:1:dir:c808e145b2558a7353eb46ff0c34aede83ec2aac`

Funding This research was partially funded by the Swiss National Science Foundation (SNSF) through projects “CrowdAlytics” (contract 184994) and “Digital Deliberative Democracy” (contract 205975).

Acknowledgements The authors would like to thank the people at ARG-tech, particularly Chris Reed, for their helpful insights and the anonymous reviewers for their constructive feedback.

Received 2025-03-31 **Accepted** 2025-11-11 **Published** 2025-12-10

1 Introduction

The Argument Web [39] was postulated more than 15 years ago. Specified as a knowledge graph describing arguments in scientific publications, the Argument Web may serve several purposes for downstream tasks, such as fact-checking and misinformation detection [9] or text summarization [37]. As such, the Argument Web of Science [45] could be part of the Web of Data modeling relevant knowledge about the scientific process and its results. While there has been significant progress in the argument mining field, we still lack the necessary tools to generate the complete graph of linked arguments, as work on cross-document argumentative structure prediction has been limited, evidenced by a scarcity of corpora with such annotations [3]. Specifically, related work has largely focused on annotating intra-document content, neglecting the intricate and highly relevant relationships between scientific publications.

This paper addresses this crucial gap, primarily by introducing a novel dataset that comprehensively annotates inter-document argumentative relations in scientific publications. To construct this dataset, we extend the state-of-the-art Sci-Arg corpus [22], which describes arguments present in 40 computer graphics scientific publications. We identify and explicitly annotate inter-document argumentative relations, where one argument component from a paper attacks or supports a claim in another paper. Unlike intra-document relations, which describe argumentative links within a single paper, inter-document relations capture how argumentative components in one scientific publication explicitly support or attack those in another. These cross-document connections are crucial for understanding the broader argumentative discourse across the scientific literature and are inferred from citation data.

As a result, we produce a dataset that adds around 800 papers to the existing 40 annotated in Sci-Arg. To represent these links, we use the simplified *claim/premise model* [54] that has found widespread use [27]. These components are connected with relations such as *attack* or *support* to form an Argument Web. We also add and distinguish the notion of inter- (i.e., across document boundaries) and intra-document (i.e., within a single document) relationships. This dataset can serve as the stepping stone to larger datasets necessary for advancing the state of the art.

Furthermore, to demonstrate the utility and establish initial benchmarks for the new dataset, we investigate different methods to generate argumentative links: first, a rule-based model using the presence of discourse indicators to predict argumentative relations; second, a state-of-the-art argument miner [31] trained on the dataset; and third, Mistral [18], a pre-trained Large Language Model, used in two different ways: zero-shot classification of relations and few-shot with similar examples [28].

Then, we *compare the results of the methods for two different settings*: intra- and inter-document argumentative relation prediction. The former is to identify argumentative structures within a single document, while the latter also considers relations across document boundaries. To this end, we split the dataset into two disjoint subsets of intra- and inter-document relations. This allows for insights into how well the methods generalize across these two scenarios.

In this paper, we, hence, present three main contributions.

1. A novel, extended dataset of 836 papers based on the Sci-Arg corpus [22]. In addition to the original annotated arguments, we explicitly include and annotate inter-document relations at a document level. We publish these annotations using Semantic Web technologies (RDF, OWL), given their foundational role in the Argument Web vision.[1]
2. An evaluation of three diverse approaches for predicting inter-document argumentative relations. We find that a simple classifier based on the presence of discourse indicators outperforms modern neural methods for the given dataset. These results serve as a critical baseline and reference for future evaluations on the newly constructed dataset.

[1] An endpoint for SPARQL queries is available at `https://sciarg.ifi.uzh.ch/#/dataset/Sci-Arg/query`

3. A comparative analysis of these approaches in both inter- and intra-document settings. We identify notable differences in accuracy across systems, indicating varying performance capabilities depending on the relation scope (within vs. across documents). This analysis can serve as a template for future comparisons.

As such, this paper can act as a stepping stone for future studies in inter-document argument mining. Furthermore, our resulting annotations serve as a rich source of detailed, structured knowledge that can extend other Semantic Web data sources in the scientific publications domain. By publishing them in RDF, we facilitate a seamless integration with other Semantic Web sources.

The remainder of this paper is structured as follows. Section 2 presents the related work, and Section 3 introduces our methodology. In the ensuing Section 4, we describe our experiments and their results. Section 5 outlines possible use cases and applications. Then, Section 6 discusses limitations and future work. Finally, we conclude with Section 7.

2 Related Work

In this section, we present relevant related works. First, we cover existing argument mining corpora in the scientific domain. Then, we describe works relating to inter-document argumentative relations. Finally, we explore approaches to using the Semantic Web to support the scientific process.

2.1 Argument Mining Corpora in the Scientific Domain

Argument Mining for scientific papers has its roots in Argumentative Zoning [49]: classifying the relevant sentences from scientific articles into one of four categories (*Claim*, *Method*, *Result*, *Conclusion*). To the best of our knowledge, Sci-Arg [22] is the only dataset for Argument Mining in the scientific domain composed of fully annotated papers in English. This is unsurprising, given that annotating gold standard data is a resource-demanding task. The challenge is even exacerbated in the case of scholarly texts because they demand considerable domain expertise of the annotators [2].

For one of the few datasets [1] available, the authors annotated 60 abstracts based on a subset of the SciDTB corpus [57]. They add an argumentation layer containing 352 argumentative components, which are connected by 292 argumentative relations, resulting in partly annotated papers from computational linguistics. Since abstracts generally do not contain references to other documents, they cannot be used for inter-document relation prediction.

Another dataset consisting of 24 papers is presented in [20]. The articles from the domain of educational research have their introduction and discussion sections argument-annotated, notably including relations. However, this corpus has the reservation that it is comprised of publications in German, for which Argument Mining approaches are very limited. Instead, our work focuses on English data as it allows us to aim for a broader audience.

2.2 Inter-Document Argumentative Relations

While not intended to be purely argumentative, the Citation Typing Ontology (CiTO) [46] allows for the classification of the nature of in-text citations as factual or rhetorical relationships. The latter has more fine-grained types such as *supports* for positive and *refutes* for negative, which directly correspond to their argumentative equivalents. Since CiTO assigns types to citations, these relationships inherently involve more than one document, making them inter-document. However, [46] presents no method for any (automated) annotations but solely an ontology. Furthermore, it also lacks a way to annotate intra-document relations since the entirety of CiTO targets citations.

The authors of [11] follow the relation-based Argument Mining paradigm [10]. Its goal is to predict the type of relations between text spans, such as attack, support, or neither. Using LSTMs [17] and GloVe embeddings [36], their approach is explicitly capable of classifying links between "any two texts," which also implies inter-document settings. The results indicate that their neural network methods improve over traditional classifiers.

The notion of "intertextual correspondence" [53] introduces another idea to connect annotated corpora (i.e., their documents) by exploiting relations, for example, of topical or temporal nature. This can also lead to multi-modal datasets for Argument Mining purposes. However, such links need to be identified manually first. The effectiveness of the approach is demonstrated by fusing a corpus of the debates in the US election of 2016 with commentary and reactions on the online platform Reddit. Such techniques could also be applied to the Sci-Arg dataset with other media or documents, since this inherently would result in inter-document relations.

In [35], the authors investigate the impact of content and context on analyzing argumentative relations. They find that systems that focus too much on the text of the argument components for relation prediction may easily be deceived and make wrong predictions. For example, they may rely on discourse indicators contained in the two text fragments to be analyzed, even though they are not adjacent, and, thus, the discourse indicator does not apply. Therefore, the importance of the context of argumentative units is asserted [35]: the position in the text and the surroundings (pre- and succeeding tokens). They show that systems only relying on the context instead of the content (i.e., component text span) may improve accuracy. Finally, the authors argue that systems dissecting content and context should also be more adequate for handling inter-document relation prediction.

2.3 Semantic Web and the Scientific Process

This work builds upon the vision of the Argument Web of Science [45]. While the notion of a scholarly knowledge graph is not novel, building one out of arguments extracted from scientific documents is. Given the prevalent role of argumentation in scientific communication [55], arguments are a suitable vehicle to represent scholarly information. Hence, we follow the *Argument Web* [39], which is based on Semantic Web knowledge representation technologies [8, 13].

The most prominent approach to a Knowledge Graph for Science has been described in [4]. Moving from document-centrism to a knowledge-based perspective would allow for a systematic organization of scholarly information. These efforts culminate in the Open Research Knowledge Graph (ORKG) [48, 5]: a representation that describes scientific papers in a structured way, facilitating question answering [33], paper comparisons, and visualizations. ORKG distinguishes itself from other large-scale efforts to represent scientific paper content, such as Microsoft Academic Graph (MAG) [56] (which is deprecated and succeeded by OpenAlex [38]), primarily in its granularity and explicit focus on structured, semantic content within papers. While MAG provided a vast bibliographic graph of papers, authors, and citations, ORKG aims to extract and structure the specific findings, methods, and results of scientific contributions, making the scientific content itself machine-readable and comparable. This deep semantic representation supports fine-grained analyses and direct comparisons of research outputs, unlike the primarily metadata-level focus of general bibliographic databases. However, ORKG uses a complex science ontology to represent the knowledge. While we share the goal of representing scientific information as a knowledge graph, our focus is different: we model arguments by means of their components (claims, premises) and two relationships connecting them (attacks, supports).

The idea behind Research Objects [6] contributes to Semantic Web efforts by defining a framework to preserve scientific workflows. It extends traditional workflow ontologies with metadata, annotations, provenance traces, and execution environments to improve reproducibility, reusability, and long-term preservation. A suite of ontologies provides structured descriptions of workflows and their evolution and aligns with other Semantic Web initiatives to structure scientific knowledge for better discovery and reuse. However, the focus is on Research Objects and their workflows, while this paper uses arguments as the core element of scientific knowledge representation.

Nanopublications [15, 21] aim to represent atomic units of knowledge with structured metadata and provenance information. This is to ensure trust, reproducibility, and interoperability. Hence, Nanopublications contribute to the Semantic Web's support for the scientific process by providing a provenance-centric format for publishing scholarly assertions, enhancing machine-readable scientific communication. However, Nanopublications focus on self-contained statements with metadata such as provenance and publication information. Our approach, in contrast, models arguments explicitly by capturing claims along with their attacking or supporting premises. This makes the logical relationship between statements a central feature.

In the context of knowledge representation and the Semantic Web, the Provenance Ontology (PROV-O) [26] primarily focuses on describing the origin, history, and derivation of entities and activities. While argument relations certainly possess provenance aspects (e.g., who asserted a claim, when, and based on what evidence), PROV-O's scope is distinct from the explicit modeling of types of argumentative connections (e.g., "supports", "attacks") between content units across documents. This is where the necessity of an Argument Web arises. Unlike general knowledge graphs that might link papers by co-authorship or topics, the Argument Web specifically aims to map the persuasive and confrontational dynamics across research. It addresses the limitations of single-document or citation-based views, particularly by capturing the complex inter-document support and attack relations that define scientific discourse. Our framework could be extended to integrate PROV-O for richer provenance tracking of the argumentative links themselves, but its core purpose lies in explicitly mapping the argumentative structure across documents, a functionality that goes beyond the scope of a pure provenance ontology. Our contribution lies in establishing the argumentative structure itself, which is a specialized form of inter-resource relationship not directly captured by general provenance models.

Lastly, SciHyp [52] introduces a fine-grained dataset representing hypotheses explicitly mentioned in scientific publications. It captures and structures scientific hypotheses from 479 computer science papers, categorized into relation-finding and comparative hypotheses. The dataset was created using a hybrid human-AI pipeline, combining experts, LLMs, and crowd refinements. Furthermore, SciHyp also extends existing hypothesis ontologies to better model their components. Through extensive evaluation, the authors demonstrate that LLMs can assist in hypothesis detection and component extraction, while also showing that human intervention remains crucial for precision. While SciHyp focuses on hypothesis-driven research, this paper aims to enhance the scientific process with a data-centered approach based on scientific arguments.

3 Methodology

This section explains the methodology behind our contributions. First, we lay the theoretical foundations of argumentative relations. Second, we describe the process of creating the new corpus by extending the Sci-Arg dataset [22] and list threats to the validity of this process and how we mitigated them. Finally, we lay out the details of the three approaches involved in generating the baseline for the new dataset, as well as how we evaluate them.

Table 1 Three examples of annotated relations taken from the Sci-Arg dataset [22].

Example	Component A	Relation	Component B
Intra-Doc	they tend to appear overly smooth and at times robotic	—attacks→	these methods do satisfy physical laws
Inter-Doc	Chadwick et al. 1989	—supports→	are computationally more expensive
Negative	its ease of implementation	—none→	SSD is a 3D transformation

3.1 Argumentative Relation Prediction

Predicting relations between argument components is the most complex and challenging part of the Argument Mining pipeline [27, 3]. The goal is to identify related pairs of argumentative components and to classify the nature of this link.

We follow Opitz and Frank [35] and define argumentative relation prediction to be an irreflexive function as follows:

$$f : C \times C \to R \quad \text{where } c_1 \neq c_2 \text{ for } (c_1, c_2) \in C \times C \tag{1}$$

C is the set of argumentative components. Each component $c \in C$ is associated with a specific document $doc(c)$ from which it originates. Components can be either a *text span* from within $doc(c)$ or a r*eference component* to an external document. For a reference component c_{ref}, $doc(c_{ref})$ refers to the document containing the citation. The range R of the function refers to the directed argumentative relation between the components. In this work, R is the set {attack, support, no-relation}, where "attacks" indicates a clash, "supports" indicates backing, and "no relation" signifies the absence of an argumentative link between the given components. Applying this function creates (typed) directed edges between argument component pairs, thus allowing us to conceive an argument graph. Table 1 provides examples of component pairs with the relation that connects them.

There are multiple ways to represent an argument, which influences the sets of C and R. In the context of this paper, we consider the following representation of an argument: the *claim/premise model* [54] with *attack* and *support* relations. In the context of this paper, the argumentative *roles* of components within C are further categorized as *claim* or *premise* (also called data, evidence, or reason in the literature [27]). A claim is a proposition about a point the arguer tries to make, and a premise is a statement about the validity of a claim. In the context of $f(\cdot)$, c_1 is typically the premise and c_2 the claim, especially for the support relation. However, there is no formal restriction on the type of components, as a claim may also attack another claim. This follows the conventions set by BAM, our Benchmark for Argument Mining [44], and is grounded in the fact that many other (more detailed) argument representations can be simplified to this model and, thus, allows for a unified evaluation (even if this incurs some information loss).

Building upon this general definition of argumentative relation prediction, we further categorize relations based on the origin and nature of their components, distinguishing between intra-document and inter-document argumentative relations. We use the following definitions:

▶ **Definition 1.** *An argumentative relation $f(c_1, c_2) \in R$ is* ***intra-document*** *if both components $c_1, c_2 \in C$ originate from the same document ($doc(c_1) = doc(c_2)$) and neither c_1 nor c_2 is a component representing a reference to an external document.*

▶ **Definition 2.** *An argumentative relation $f(c_1, c_2) \in R$ is* ***inter-document*** *if the two components $c_1, c_2 \in C$ do not originate from the same document ($doc(c_1) \neq doc(c_2)$)* ***OR*** *at least one of the components is a component representing a reference to an external document.*

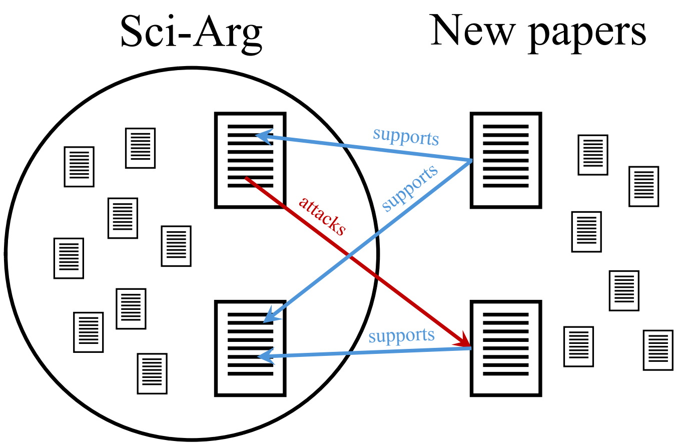


Figure 1 A visualization of the extension of the Sci-Arg corpus [22] with inter-document relations represented by the directed edges.

With the concept described in Definition 2, we can build an Argument Web as shown in Figure 1 since arguments can now be linked across multiple documents. Furthermore, using these definitions, we can form two (disjoint) subsets of relations for any dataset. It is important to clarify that this disjointness applies to the classification of argumentative relations based on the defined criteria of component origin and type (text span vs. reference to another document). This does not prevent the possibility of shared or overlapping content existing between documents that are involved in inter-document argumentative relations. Indeed, such shared content often provides the very basis for one paper to support or attack another. Our definitions categorize the type of argumentative relation established between components, rather than asserting that the underlying documents themselves are semantically or topically distinct. This ensures a clear distinction in how different types of argumentative interactions, within a document versus across documents, are formally represented and evaluated. Annotation examples from the Sci-Arg dataset [22] can be seen in Table 1 for the relation types to predict, as well as the two settings we distinguish.

3.2 Corpus Creation

We based the extended dataset on the freely available Sci-Arg corpus [22], which consists of 40 computer graphics papers fully argument annotated, including components and relations. Sci-Arg was annotated by one expert and three non-experts in five iterations. The reported Inter Annotator Agreement (IAA) in terms of the F_1-measure was reported with two criteria. For the strict version, span and type have to match exactly for components, and for relations, the direction also has to be correct. In the relaxed version, components only have to correspond in type and match half the span. This results in a higher IAA for the relaxed criteria: 72% for the components and 49% for relations. Meanwhile, the strictly measured IAA is 60% for the components and 35% for relations. The new annotations and the code for the automated processes are available in the online repository.[2]

Figure 1 visualizes the concept behind the extension of the dataset, which is represented by the circle, adding new documents outside of it. From the Sci-Arg annotations, we identified relations that involved components, which are references to other documents. That is, we determined triples consisting of an argumentative unit annotated in the dataset, a relation (type), and an

[2] `https://gitlab.ifi.uzh.ch/DDIS-Public/midas`

external paper acting as the respective endpoint. Crucially, for these inter-document relations, our annotation captures a document-level relationship, indicating that the referencing document supports or attacks the cited document as a whole, rather than specific claims or spans within it. Hence, we did not identify or annotate specific argumentative spans within the target (cited) papers. Instead, the argumentative relation type (supports or attacks) for these inter-document links was directly inherited from the original intra-document annotation in Sci-Arg. Specifically, if an argumentative component within one of the original 40 Sci-Arg papers was annotated with a certain relation (e.g., "supports") to another component, and that second component was identified as a citation to an external document, then that existing "supports" label was applied as the relation between the citing component and the cited document as a whole. Thus, the labels for inter-document relations originate solely from the high-quality, human-derived annotations within the original Sci-Arg corpus, ensuring their trustworthiness without requiring fine-grained annotation of the newly added documents. This process yielded 796 unique external document references from the initial 40 Sci-Arg papers that were part of annotated argumentative components. These references formed the basis for expanding our network of inter-document relations.

We assumed that if an in-text citation is annotated as an argumentative component, the original paper the citation is in reference to is meant to be used as the component. Therefore, we extracted all these components that contain a citation based on pattern matching (e.g., "Smith et al., 2000" or "[1]").[3] We then manually verified that all the identified components were citations in their context by looking through the 40 documents in the Sci-Arg corpus. This involved confirming the presence of standard citation formats (e.g., author-year mentions or numerical indices) that directly corresponded to entries in the papers' bibliographies, ensuring the component's accurate role as a citation to an external source.

Next, we matched the citations to their reference in the bibliographies in the papers. This was automated wherever possible. For the cases of ambiguity, occurring in approximately 10% of the instances, we determined the references by hand. These typically arose from multiple bibliography entries matching a single citation pattern, from unclear author-year combinations, or special characters not handled correctly during the content extraction. The issues were resolved by careful cross-referencing with the bibliography, the citing document's content, and, when necessary, external academic search engines, following a predefined set of internal guidelines to ensure consistency.[4] This way, we could assign the new documents to the annotated argument components.

Then, we resolved the references to their Digital Object Identifiers (DOI) using Crossref's API.[5] These API calls were successful for approximately 90% of the identified references. For the remaining cases without a direct DOI, we followed the guidelines detailed in the online repository to derive a DOI or, when none was available, another unique, persistent identifier (e.g., arXiv ID, official publisher URL) that ensured the consistent and unambiguous identification of the target paper. With these, we could now obtain the content of papers as PDFs, if accessible, from which we extracted information about their content and structure using Papermage [29]. We successfully obtained PDF content for 649 of the 796 new papers (approximately 81.5% of the newly added documents). From these PDFs, Papermage extracted key structural elements such as paragraphs, sentences, and headings, making their content programmatically accessible for potential future use, even though fine-grained annotation of these new documents was not within the scope of this initial extension.

[3] The full list of patterns is available in Appendix A.
[4] `https://gitlab.ifi.uzh.ch/ruosch/wp3/-/blob/main/data/guidelines.md`
[5] `https://api.crossref.org`

It is crucial to clarify that for the current scope of this dataset extension, the detailed full-text content extracted by Papermage from these additional 649 papers was not directly used to determine the argumentative relation types (supports/attacks) between documents. These relation types were derived by performing lookups in the original, human-annotated Sci-Arg data, where the citation acted as an argumentative component. Therefore, the content acquisition and extraction primarily served to identify and confirm the existence and structure of the cited documents and to build a resource for future, more granular analyses. While this approach might be considered "shallow retrieval" in terms of not using the full text for the initial relation labeling, it was a deliberate and necessary choice for the initial, broad-scale expansion of the Argument Web. It allowed us to efficiently establish a foundational network of document-level argumentative links. The availability of this extracted full text for a large portion of the new documents will be crucial for future work aimed at developing more sophisticated, content-based methods for fine-grained inter-document argument mining and automatic relation prediction, moving beyond simple citation-based links.

To finalize the new annotations for the intra- and inter-document argumentative relations, we needed to complete the triples with their predicate (relation type) and assign them to either of the two disjoint subsets. For this, we performed lookups in the original Sci-Arg data using the two component identifiers to get the relationship label connecting the two. Then, we added these triples to their corresponding subset based on Definitions 1 and 2: if both components were from the same document, they were classified as *intra*, and *inter*, otherwise.

As a result, we had a new dataset with two additional aspects. First, we added the distinctive notion of intra- and inter-document argumentative relations, producing annotations of these two disjoint subsets. This allows us to evaluate relation prediction approaches in the two settings independently, potentially leading to more in-depth analyses of how different methods work for them. Second, and more importantly, we augmented the original 40 papers by including an additional 796 new papers, extending the dataset to 836 total papers.

It is crucial to clarify that this expansion primarily involved identifying external papers cited by the original 40 and determining document-level relationships to them. Thus, while this represents a twentyfold increase in the number of documents, the detailed, claim-level annotations present in the original 40 papers were not replicated for the additional 796 documents. This targeted approach required substantially less effort per additional document, allowing us to broaden the scope of inter-document relations efficiently. Even though we do not currently make use of the full text of the additional papers for detailed internal annotations, their inclusion is vital. These 796 documents serve as essential nodes within the expanded inter-document argumentative graph, providing a richer, more representative context for studying cross-document argumentation. They enable the analysis and prediction of how arguments originating from the original 40 papers relate to a much wider body of scientific literature, making them valuable for understanding the broader Argument Web.

Additional statistics about the original and the extended dataset can be found in Table 2 (the class distributions are shown in Appendix F). The number of total relations remains unchanged in the extension because these 1996 relations represent all argumentative links identified within and originating from the original 40 Sci-Arg papers. Our extension primarily involved: 1) identifying which of these existing relations were inter-document (i.e., involved a reference to an external paper), and 2) adding the external documents themselves as nodes to the Argument Web to serve as endpoints for these newly categorized inter-document relations. We did not perform new fine-grained argumentative annotations within the 796 newly added papers. Instead, the total number of relations reflects the comprehensive set of argumentative links present in the original Sci-Arg corpus, now re-categorized and contextualized within a larger network of documents.

Table 2 The statistics of the original Sci-Arg [22] and its extension.

Number of	Sci-Arg	Extension
Total Relations	1996	1996
Intra-Doc Relations	n/a	1428
Inter-Doc Relations	n/a	568
Doc-Level Relations	n/a	1109
Documents	40	836

We determined 568 relations that go across document boundaries and 1428 within documents. If we only allow one relation per direction and type for each pair by consolidating relations where one paper is used to support or attack at multiple locations in another, we reduce the annotations to the document level. This way, we maintain a graph with 836 nodes and 1109 edges (i.e., the number of doc-level relations).

We publish these document-level annotations as Resource Description Framework (RDF) data since Semantic Web technologies provide the means to represent machine-readable data that is easy to integrate with other data despite heterogeneity.[1] This is particularly useful in this context, as there exists a variety of argument representation models with different levels of granularity [27]. To model the data, we used the CrowdAlytics ontology, which extends other ontologies and integrates ontological components to describe scientific hypotheses and arguments present in scientific publications [51]. Even though we produced an Argument Web, the inter-document relations are indeed only on the document level.

This document-level granularity for inter-document relations was a deliberate design choice, allowing us to scalably extend the dataset's scope to hundreds of external documents. Unlike the fine-grained, claim-level annotations within documents, identifying specific span-level argumentative connections across disparate documents proved prohibitively complex and resource-intensive for manual annotation, so we only resolved the references to the documents and did not determine the more fine-grained details of which exact proposition the reference is to. This is visualized in Figure 1 by the arrows representing the relations only pointing to the documents and lacking detail outside the circle representing the original corpus. The implication of this design is that our dataset provides a foundational macro-level view of argumentative connections between papers, capturing their overall support or attack roles. While this work focuses on these document-level interactions, future research can leverage this foundation to develop automated methods for identifying more granular, span-level inter-document argumentative links.

For consistency, we thus make this difference in the relations evident by altering their labels slightly. *Supports* becomes *is used as support in*, and *attacks* turns into *is used as attack in*. This emphasizes the difference between argument component pairs where both are well-specified propositions as opposed to those consisting of one text span and one reference to a different document. Semantic Web technologies facilitate the alignment of the two different granularities of annotations in the same dataset.

Threats to Validity

Several threats to validity impact our corpus creation. First, manual processes were crucial for verifying identified citation components within their context and for resolving ambiguous references in bibliographies. However, such manual intervention inherently introduces the risk of human error, misinterpretation, or inconsistency, which could lead to incorrect associations between argumentative components and external papers, thereby compromising the accuracy of

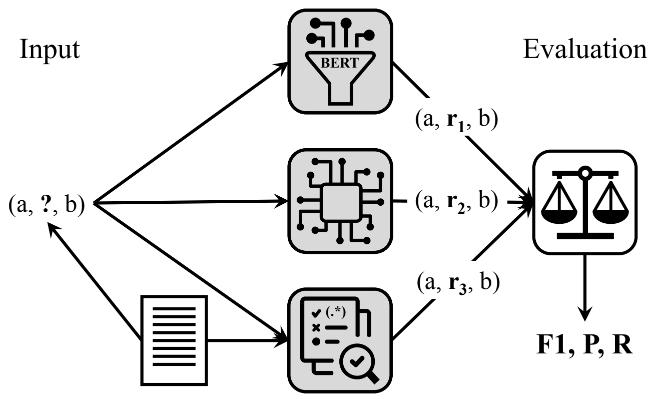


Figure 2 Overview of the three approaches for predicting the argumentative relation for two given argumentative units.

the inter-document relations. To address this, we systematically went through all 40 documents of the Sci-Arg corpus for manual verification and resolved ambiguities by hand, aiming for precision in these critical steps. For consistency and reproducibility, these manual steps are detailed in the internal annotation guidelines.[4]

Second, our reliance on pattern matching for extracting components containing citations, while efficient, carries the threat of incompleteness if non-standard citation formats were missed, or inaccuracy if text was incorrectly identified as a citation, potentially leading to an incomplete or flawed set of inter-document relations. We sought to mitigate this by providing a comprehensive list of patterns in the Appendix, allowing for transparency and future refinement.

Third, challenges encountered during the Digital Object Identifier (DOI) resolution process using Crossref's API, and the subsequent efforts to manually derive DOIs or obtain paper content, meant that not all referenced papers were accessible or fully processable. This limitation means the dataset may not encompass all intended inter-document relations, or the full text content crucial for future work leveraging Papermage was not uniformly obtained across all added documents. We addressed this by following specific guidelines detailed in our online repository for deriving DOIs or other unique identifiers when automated methods failed. Crucially, our underlying assumption that an in-text citation annotated as an argumentative component refers to the entire original paper simplifies potentially more nuanced argumentative connections. This design choice directly impacts the granularity of our inter-document relations, which are modeled only at the document level. Consequently, while we clarified this distinction in the annotations by altering relation labels (e.g., "supports" to "is used as support in"), the dataset does not capture the precise argumentative unit (e.g., a specific proposition) within the external document that is being supported or attacked, thereby limiting the depth of fine-grained argumentative analysis possible across documents. We explicitly acknowledged this limitation by making the difference in relations evident through altered labels and noting that Semantic Web technologies facilitate the alignment of these different granularities within the same dataset.

3.3 Automated Relation Prediction

This subsection describes the three approaches used to create the baseline for the new dataset. Furthermore, we also explain the evaluation methods that we apply. An overview is shown in Figure 2: the inputs on the left-hand side, the three approaches in the middle, and the evaluation on the right. The first two approaches only receive the two components to predict the relation, while the bottom one also has access to the full text. They are all evaluated by the same methodology to produce comparable results in the F1-score.

3.3.1 Rule-Based Approach

The first approach for predicting argumentative relations is rule-based on the presence of discourse indicators. It is depicted as the bottom of the three gray center squares in Figure 2. Discourse indicators have already been successfully employed as an Argument Mining technique [47] and have been shown to be a dependable method for argument relation prediction [25]. To keep this method simple, we take the 24 expressions from [24, p. 128] (cf. Appendix B) as the set of discourse indicators.

In order to predict the relation for a given pair of argumentative components (text spans), we first check if they occur in the same sentence in the original text. For this, we have to give the system access to the document and, thus, the context of the argumentative units, since the discourse indicator may occur outside of their boundaries. If we fail one of the two checks, we assume the pair to be not related and predict *no relation*. Otherwise, we construct a triple of the two argument components and derive the relation type from the nature of the discourse indicator. Some signify *support* (e.g., “because” or “since”) and others *attack* (e.g., “but” or “despite”).

It is important to note that the annotation guidelines for Sci-Arg [23] only mention discourse indicators as cues for annotating components but not for relations. Hence, no correlation is expected ex-ante for the relation prediction.

3.3.2 Argument Miner

As the argument miner, we use the system described and implemented in [31]. Depicted as the top of the three gray squares in Figure 2, it combines pre-trained transformers (variants of BERT [12]) with recurrent architectures (GRU, CRF, LSTM) for mining arguments and their structures from natural language texts.

The prediction of argumentative relations between components (text spans) is modeled as a sequence classification problem. For the pairs of components, one of the following three classes is to be chosen: *Support*, *Attack*, *NoRelation*. Furthermore, one component can be related to multiple other components, since each combination is classified independently. Given its top-performing results in [31], we opted for the uncased SciBERT [7] as the transformer to compute the embeddings. The pooled representation of the sequence to be classified is fed into a linear layer with a softmax that produces a distribution over the target classes. These results can be used to predict the argumentative relationship type between the two given components. The details regarding fine-tuning, exact parameters, and data splits can be found in Appendix C.

3.3.3 Large Language Model

We use a quantized version of Mistral-7B [18] as the large language model in two different ways, shown as the middle of the three gray squares representing the approaches in Figure 2. Our choice of Mistral-7B was driven by the constraint of running the model on a single GPU. This approach improves the reproducibility and generalizability of our work, as it makes the methodology accessible to researchers with more limited computational resources. It also significantly speeds up computation, which is essential for future large-scale experiments. While we also experimented with several other LLMs (variants of GPT [34] and Llama [14], among others) that fit this criterion, we ultimately selected to only include Mistral-7B as it consistently showed superior performance on our specific task.

The first is naive zero-shot classification, whereby the model is prompted with the task phrased as a multiple-choice problem. This involves predicting the argumentative relation from the set *attacks*, *supports*, or *none* between two argumentative components, given as text spans. Mistral is then asked to respond with a single word, i.e., the classification of the relation. The detailed prompts can be found in Appendix D, including the class distribution of the in-context examples.

Since argumentative relation prediction is a complex task [3], we also applied a more sophisticated few-shot learning method, utilizing in-context learning based on similar examples [28]. We use the same template as in the zero-shot classification, but enrich it with examples of triples formed by component pairs and their connecting relation. These examples are dynamically generated by identifying the five most semantically similar components in the training set by deriving their embeddings with Sentence-BERT [42] and applying cosine similarity. We select the five most semantically similar examples for in-context learning because they are hypothesized to provide the most relevant contextual cues and linguistic patterns for the LLM to learn from, facilitating better generalization. This approach aims to maximize the effectiveness of the limited in-context examples by ensuring their direct applicability to the target prediction, thereby improving the model's ability to discern complex argumentative relationships. We then retrieve the ground truth relations for these five semantically similar pairs to create the in-context examples. The final prompt is composed of the task description, followed by these curated examples, and finally, the components for which the relation is to be predicted. We hypothesize that providing the LLM with information about relations from semantically similar components will help it to predict the relation correctly.

Finally, for a fair comparison to the other two methods, we will evaluate the LLM approach with and without providing the context of the argumentative components. To this end, the LLM will receive the full sentences from which the components are taken in the prompt, alongside the argumentative component spans themselves. This allows the model to leverage a more complete understanding of the surrounding discourse, which is known to be a crucial factor in argumentative reasoning and provides a direct way to measure the impact of context on LLM performance for this task.

3.3.4 Evaluation

For the evaluation of the three approaches, we employ the routines implemented by BAM [44], our benchmark for Argument Mining. It is represented by the scale in the square on the right in Figure 2. It splits the evaluation into the four stages (*sentence classification*, *boundary detection*, *component identification*, and *relation prediction*) of the Argument Mining pipeline described in [27], from which we only use the last step.

To evaluate the argumentative relation prediction, BAM treats it as a binary classification (retrieved or missed) of triples *(subject, predicate, object)* and applies the F1-score [50]. While argumentative relations in our dataset consist of three classes ("attacks", "supports", and "no relation"), BAM's relation prediction evaluation frames the problem as identifying relevant/irrelevant and retrieved/missed triples. This means, for each potential (subject, object) pair, the system's task is to determine if a specific "attacks" or "supports" triple exists and, if so, to correctly identify it. The "no relation" case is implicitly handled as the absence of such a positively identified argumentative triple. The subject and object are represented by the identifiers of the corresponding argumentative component (i.e., text span) as given in the ground truth annotations of the Sci-Arg dataset. The predicate is the label of the argumentative relationship between the two. By comparing the ground truth of the test set and the predictions, we get a result for the relation prediction score represented by F1 between zero and one for each approach, where bigger signifies better. The code involved in the evaluation is available in the online repository.[2]

Table 3 The results of the experiments on the overall dataset. We report the F1-score as measured by BAM [44], the precision, and the recall. Also, we indicate if the method had access to the context.

Approach		Context	F1	Precision	Recall
Rule-Based		✓	0.437	0.554	0.372
Argument Miner	[31]		0.251	0.410	0.187
Zero-Shot LLM	[18]		0.113	0.061	0.802
Zero-Shot LLM	[18]	✓	0.119	0.065	0.850
Few-Shot LLM	[18]		0.109	0.141	0.095
Few-Shot LLM	[18]	✓	0.132	0.099	0.217

4 Results

This section discusses the experiments conducted to create the baseline and their results. We first look at the outcome of the approaches on the overall dataset. Then, we examine the differences when they are applied in the intra- and inter-document settings.

The code of the experiments and the results are available in the online repository.[2] All statistical test procedures and their results ($\alpha < 0.05$) can be found in Appendix E.

4.1 Overall Dataset

Table 3 shows the results for the approaches on the overall dataset. A class-based evaluation can be found in Appendix F. Furthermore, we show more detailed statistics on the distribution of the misclassified samples per relation type of each system in Appendix G.

With a relation prediction score of 0.437, the naive approach, which leverages only the presence of discourse indicators, clearly – and statistically significantly – outperforms the other more sophisticated neural methods. This may be surprising, but it can be put into perspective with the following three points. First, discourse indicators have been shown to work well for predicting argument relations [25]. Second, it was the only technique that had access to not only the content but also to the context of the argument components. This has been noted to contribute to accuracy positively [35]. Still, the only influence the access to the context had on the approach was that the discourse indicator could be contained in either the sentence holding the components or in the components themselves. Finally, even though the rule-based approach achieved the highest score, the result is nowhere near where it could be, as the 0.437 score clearly indicates significant room for improvement.

Scoring 0.251, the transformer-based Argument Mining system [31] came in second place and considerably ahead of the LLM approaches. This result is consistent with the outcome for TRABAM in the original showcase of BAM [44]. Even with the training on the dataset, it appears the system still fails to predict most of the relations for argument component pairs correctly.

The results of the LLM-based approaches are nuanced. Both the zero-shot and few-shot methods show a statistically significant improvement in their mean F1-scores when provided with context (p=0.008 and p=0.012, respectively). However, the effect size of this improvement is markedly different. For the zero-shot LLM, the improvement is negligible (d=-0.173), while for the few-shot LLM, the effect is medium (d=-0.618). This finding corrects our initial hypothesis, as it indicates that the information from semantically similar examples *is* effective at improving accuracy, but only when combined with the contextual information from the surrounding sentences.

When we examine the performance with context more closely, the few-shot LLM achieves the best F1-score among the LLM variants (0.132). However, this improvement comes with a trade-off: while its recall increases substantially ($0.095 \rightarrow 0.217$), its precision decreases ($0.141 \rightarrow 0.099$).

Table 4 The results of the experiments on the intra- and inter-document subsets. We report the F1-score as measured by BAM [44], the precision, and the recall.

Approach		Context	F1	Precision	Recall
Rule-Based	Intra	✓	0.430	0.514	0.380
	Inter		0.432	0.649	0.348
Argument Miner	Intra		0.238	0.360	0.184
	Inter		0.345	0.502	0.276
Zero-Shot LLM	Intra		0.083	0.044	0.718
	Inter		0.249	0.145	0.970
Zero-Shot LLM	Intra	✓	0.090	0.048	0.782
	Inter		0.252	0.146	0.987
Few-Shot LLM	Intra		0.091	0.131	0.075
	Inter		0.136	0.136	0.144
Few-Shot LLM	Intra	✓	0.113	0.085	0.182
	Inter		0.165	0.118	0.294

This behavior indicates that providing context and few-shot examples enables the model to identify more potential relations but at the cost of generating more false positives. This finding confirms that while LLMs are sensitive to contextual information, they still struggle to accurately and precisely discern the correct relations, particularly without extensive fine-tuning.

A key observation from our class-based results (cf. Appendix F) is the significant disparity in performance between "supports" and "attacks" relations, where models, particularly the LLM-based approach, consistently exhibit much higher F1-scores for "supports". This phenomenon can be attributed to a combination of factors. Firstly, the inherent class imbalance in our dataset, as detailed in Subsection 3.2 (Table 2), means "supports" relations are substantially more frequent than "attacks". This naturally biases models towards the majority class. Secondly, identifying "attacks" relations is often an intrinsically harder task in argument mining. They frequently involve more nuanced linguistic cues, require deeper semantic understanding of contradiction or refutation, and may manifest in more diverse textual patterns compared to expressions of support. The confusion matrices (Appendix G) further illuminate this, revealing instances where models tend to conflate "attacks" with "no relation" or even incorrectly classify them as "supports", highlighting the challenge in accurately discerning these critical dissenting links.

4.2 Intra- Versus Inter-Document Setting

The results for the intra- and inter-document settings are shown in Table 4, revealing that all systems achieve higher scores for the latter. We make pairwise comparisons to see whether these numbers are statistically significantly different for F1.

This is neither the case for the rule-based nor for the few-shot LLM approach. The two represented the two different ends of the scale of the results. The former is at the top (0.430 and 0.432), and the latter is at the bottom (0.091 and 0.136). However, they do not exhibit statistically significantly different scores for the argumentative relation prediction in the intra- and the inter-document settings.

Still, the results of the rule-based approach give insights. They indicate that discourse indicators in scientific papers are a simple but reliable signal for predicting argumentative relations in the two situations. This is shown in both settings by the higher precision than recall of the method, confirming the findings in [24].

Meanwhile, the transformer-based argument miner [31] achieves a distinctly higher score in the inter-document (0.345) than in the intra-document (0.238) setting. The difference has statistical significance, suggesting that it is better at predicting this type of argumentative relationship. The precision and recall values, with the former being higher than the latter, also indicate that the AM system is able to pick up on the cues in the content of the components. However, it fails to capture most of the argumentative relations.

For the zero-shot LLM approach, the statistically significant difference in the relation prediction scores is even larger: 0.083 and 0.249, respectively. The very high recall values stand out for both settings, indicating that it catches many of the argumentative relations to predict, but also produces a lot of garbage annotations, considering the very low precision. The analysis for the few-shot LLM approach is the same as that for the overall dataset. It is inadequate for predicting argumentative relations either way, with some of the lowest F1, precision, and recall scores across the board. Since these systems are based on black-boxes that are the neural network architectures, we can only speculate about the reasons for the disparities. For some LLMs, we have knowledge about the data involved in their pre-training, while others may be completely closed off. However, given that their training involved coherent and, most likely, academic text, we can assume that their capabilities involve recognizing argumentative components such as claims and premises but struggle to pick up on cues for the more complex task of argumentative relation prediction. More investigations in this direction are clearly necessary, also when taking the difficulties of effective prompting into account.

For both the argument miner and the LLM, the results may seem counterintuitive when comparing the relation prediction scores to those of the naive rule-based approach. The components in the inter-document setting tend to contain less semantic and syntactic information, as at least one of them is a reference to another document: a citation. In most cases, that does not give any details about the referenced document apart from possibly author names and the publication year. Neither of which carries significant information without more semantic context. Therefore, the only conclusion is that for the inter-document setting, the complementary component of the pair (i.e., not the citation) contains particularly useful information for predicting the relation. Surprisingly, this would also imply that the cues for the inter-document argumentative relation prediction from only one component can be leveraged more effectively than those from two components in the intra-document setting, resulting in higher scores for the former.

The LLMs show a varied response to the addition of context. The few-shot LLM with context shows no statistically significant difference between intra- and inter-document performance. In contrast, the zero-shot LLM with context exhibits a large and statistically significant difference between intra- and inter-document F1 scores. This indicates that while context helps both approaches, the few-shot learning method, when combined with context, is better able to adapt and apply its learned knowledge more consistently to both intra- and inter-document relationships, reducing the performance gap between the two, as was also the case without context.

5 Use Cases for an Argument Web

The Argument Web [40] and, hence, our extended dataset and the developed approaches for inter-document argumentative relation prediction, provide a foundational layer for a range of applications, particularly within the scientific domain. By explicitly modeling how scientific arguments interact across different papers, our work enables a deeper understanding and more efficient navigation of scholarly discourse.

Firstly, our approach can significantly enhance scholarly recommendation systems. Moving beyond traditional keyword matching or citation networks, our dataset allows for the development of systems that suggest papers based on their argumentative relationship to a user's current reading.

This means a researcher could be proactively recommended not only supporting evidence but also crucial counter-arguments or alternative perspectives, fostering a more critical and comprehensive literature review process. Such systems could also help researchers build a better understanding of a topic's argumentative landscape, rather than just its content.

Secondly, these insights are crucial for the automated construction of scientific knowledge graphs. By identifying and classifying explicit "supports" and "attacks" relations between documents, our methods facilitate the population of knowledge graphs with argumentative links. This enriches the semantic representation of scientific discourse, allowing for complex queries that trace the evolution of ideas, identify the provenance of specific claims, or map the full spectrum of evidence surrounding a hypothesis. Such structured argumentative knowledge can serve as a backbone for advanced AI applications in scientific discovery.

In the evolving landscape of LLMs and conversational AI, the explicit modeling of argumentative structures, as enabled by our dataset extension, gains particular significance. Our annotations provide a crucial resource for training LLMs to generate more accurate, reasoned, and evidence-backed responses. By understanding support and attack relations across documents, LLMs can help with:

1. Enhance Factuality: Ground their generated content in verifiable evidence by identifying supporting arguments from established literature, thereby combating hallucinations with a belief graph [19].
2. Improve Reasoning: Generate more coherent and logically structured arguments by mimicking observed patterns of claims, premises, and their interconnections.
3. Synthesize Debates: Effectively summarize complex scientific discussions by identifying the core arguments for and against specific theories or findings from multiple sources.
4. Provide Justifications: Equip chatbots and conversational agents with the ability to offer transparent justifications for their answers, referencing supporting evidence or acknowledging counter-arguments.
5. Detect Controversies and Bias: Recognize areas of scientific disagreement or identify potential biases in presented arguments by mapping where attacks are concentrated or support is lacking.

Ultimately, our work contributes foundational data structured by explicit inter-document relations, establishing a basis for future research. This data can empower LLMs to move beyond mere text generation toward more sophisticated, critically aware, and trustworthy engagement with scientific knowledge by providing access to the evidentiary chain of scientific claims.

Furthermore, the ability to identify cross-document argumentative relations can greatly facilitate automated literature review, survey generation, and summarization [37]. Researchers often spend considerable time synthesizing arguments and counter-arguments across a vast body of scientific literature. Our approach could help automate this process by providing an "argument map" for a given research question, highlighting key supporting evidence, summarizing different positions on controversial topics, or even generating preliminary argumentative outlines for review articles, thereby drastically reducing manual effort.

Beyond general scientific discourse, Argument Mining and the Argument Web approach hold significant promise for applications in healthcare [30]. Clinicians and researchers in medicine constantly grapple with vast amounts of evidence for and against various therapies, diagnostic methods, and treatment protocols. An Argument Web in the clinical space could provide structured, machine-readable evidence for decision-making, allowing users to trace the supporting and attacking arguments for specific medical recommendations or interventions across countless research papers, clinical trials, and guidelines. While the current dataset is from computer graphics, the methodology for extracting and linking inter-document arguments is potentially transferable, suggesting a powerful tool for evidence-based medicine, systematic reviews, and even supporting clinical guideline development by explicitly mapping the underlying argumentative landscape of medical knowledge. We believe this area represents a particularly impactful future direction for the Argument Web.

Finally, by comprehensively mapping the argumentative landscape of scientific fields, our work provides a robust framework based on explicit inter-document support and attack relations for subsequent analysis aimed at quantitatively identifying research gaps, key disagreements, and emergent scientific controversies across a field. The explicit identification of "attacks" relations, particularly when concentrated on specific arguments or findings, can pinpoint areas of ongoing debate, unresolved issues, or even fundamental assumptions that lack robust support. Conversely, the absence of strong support relations for a new claim might indicate a research gap. This capability can guide future research directions by highlighting critical areas for further investigation or areas where a particular line of argument has been consistently refuted, enabling the scientific community to focus efforts more effectively.

6 Limitations and Future Work

The main limitation of this work is that all investigations and evaluations were conducted on a dataset that only represents a specific domain of scientific papers. Therefore, whether the insights generalize well or at all to other natural language texts remains to be evaluated. In addition, while we know the domain of the initial dataset is computer graphics [22], we do not have any information about the newly added papers. Since we did not find an automated way to get the topics for them, we leave it as future work to see if and how many new topics have been added to the dataset.

Another constraint was already pointed out in the description of the corpus creation: the inter-document relations are only annotated on the document level, and, therefore, there is a lack of detail. The new annotations are valuable information for evaluating methods for predicting argumentative relations. Still, *anchoring* the relations (i.e., identifying the exact location) in the new documents is vital. This goes hand in hand with bringing these annotations to the same level of detail as the Sci-Arg corpus. Furthermore, this would enable an iterative process by further extending this dataset with newly identified inter-document relations for the additional papers, and so forth.

Curating these annotations is very time-consuming, even more so for documents as specialized as scientific papers, since they require extensive domain knowledge. Therefore, the question of how human annotations can be reduced to facilitate the tasks for the annotators should be considered. This exploration is left as future work.

Furthermore, we imposed some limitations on the argument model in the evaluation. They stem from the way the evaluation is set up using BAM, because only in doing so could we produce comparable results for the different Argument Mining approaches. This also entailed a degree of information loss since we did not incorporate all available information from the original dataset. For example, we did not distinguish between *own* or *background* claims, and for relations, we left out the *semantically-same* and *parts-of-same*.

To enhance the robustness and generalizability of argumentative relation prediction, particularly for the challenging "attacks" class, future work should prioritize targeted strategies. Given the observed class imbalance and the inherent difficulty of the task, approaches such as cost-sensitive learning, advanced data augmentation for minority classes, or specialized prompt engineering for LLMs that emphasizes adversarial reasoning could prove beneficial. Furthermore, a detailed analysis of the misclassified instances, guided by the inter-class confusion matrices, will be instrumental. This fine-grained error analysis can help in identifying specific linguistic or structural patterns that currently mislead models, giving rise to the development of more discriminative features or targeted training strategies to reduce confusion between "attacks" and other relation types. Improving the detection of "attacks" is crucial for tasks like identifying scientific controversies and understanding dissenting viewpoints.

Moreover, there are further approaches that could be applied to argumentative relation prediction that we did not include in our work. The explanations for this are twofold. First, we aim to provide a baseline on the dataset. Second, we wanted to use off-the-shelf methods, which also facilitate the reproducibility of our results. Our main contribution remains the new dataset that we generated and published as machine-readable data. Using it, we can improve the baseline by putting out a shared task or a challenge and getting more methods involved.

While the groundwork has been laid in [35] with their analysis of content and context for argument relations, a detailed study of what works and what does not remains pending. We can only surmise why the simple, rule-based approach outperforms the more sophisticated approaches so clearly. Examining the shortcomings of the transformer-based and the LLM methods should shed some light on the matter.

Another aspect that we left out in this work is investigating whether the contents of the documents alone can be used to predict the document-level argumentative relations. With the increasing context windows of LLMs [43], they might be capable of correctly identifying the relations between documents without relying on detailed component annotations. They could construct Argument Graphs from document contents alone. However, as made evident in the results of the LLM approach in our work, the relation prediction remains challenging for them. Thus, we hypothesize that this is no trivial task and leave it for future work.

To further facilitate the adoption and application of our work by the broader research community, future efforts can include the development of a dedicated use cases website, accompanied by comprehensive documentation and tutorials. This resource could aim to provide practical guidance and showcase various applications of the extended dataset and the developed argumentative relation prediction approaches.

7 Conclusions

This work addressed the challenging task of mining inter-document arguments in scientific papers. We focused on predicting argumentative relations, such as *attacks* and *supports*, between argument components (*claims*, *premises*). In this context, we extended an existing dataset [22] by explicitly annotating it with inter-document argumentative relations. Then, we explored three automated argumentative relation prediction approaches and evaluated them on the original dataset and the newly annotated inter-document relations.

This work aligns with the ultimate goal of constructing an Argument Web [39] of Science. Our contributions include the creation of a new dataset with explicitly annotated inter-document argumentative relations. We published it using Semantic Web technologies, extending its size from the original 40 papers to over 800. This endeavor hopes to contribute to advancing the dissemination of scholarly discourse.

Furthermore, our analysis of the baseline results indicates that a simple rule-based classifier leveraging the presence of discourse indicators outperforms neural methods for argumentative relation prediction. This finding emphasizes the effectiveness of exploiting the linguistic features of the context of components in predicting argumentative relations, which has previously also been noted in [35]. Furthermore, we observed statistically significant differences in accuracy between the intra- and inter-document settings for the evaluated approaches. This highlights the importance of distinguishing between these two.

In summary, our efforts mark a step forward in understanding and harnessing the complex web of arguments in scientific papers. By providing an extended, well-analyzed dataset for intra-document Argument Mining, we hope to bootstrap the effort for large-scale datasets that pave the way to the Argument Web of Science [45], which in turn could serve as a major pillar for the Web of Data with respect to science, the scientific process and discourse, and its results.

References

1 Pablo Accuosto and Horacio Saggion. Transferring knowledge from discourse to arguments: A case study with scientific abstracts. In Benno Stein and Henning Wachsmuth, editors, *Proceedings of the 6th Workshop on Argument Mining*, pages 41–51, Florence, Italy, August 2019. Association for Computational Linguistics. doi:10.18653/v1/W19-4505.

2 Titipat Achakulvisut, Chandra Bhagavatula, Daniel E. Acuna, and Konrad P. Körding. Claim extraction in biomedical publications using deep discourse model and transfer learning. *CoRR*, abs/1907.00962, 2019. arXiv:1907.00962.

3 Khalid Al Khatib, Tirthankar Ghosal, Yufang Hou, Anita de Waard, and Dayne Freitag. Argument mining for scholarly document processing: Taking stock and looking ahead. In Iz Beltagy, Arman Cohan, Guy Feigenblat, Dayne Freitag, Tirthankar Ghosal, Keith Hall, Drahomira Herrmannova, Petr Knoth, Kyle Lo, Philipp Mayr, Robert M. Patton, Michal Shmueli-Scheuer, Anita de Waard, Kuansan Wang, and Lucy Lu Wang, editors, *Proceedings of the Second Workshop on Scholarly Document Processing*, pages 56–65, Online, June 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.sdp-1.7.

4 Sören Auer, Viktor Kovtun, Manuel Prinz, Anna Kasprzik, Markus Stocker, and Maria Esther Vidal. Towards a knowledge graph for science. In *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*, WIMS '18, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3227609.3227689.

5 Sören Auer, Allard Oelen, Muhammad Haris, Markus Stocker, Jennifer D'Souza, Kheir Eddine Farfar, Lars Vogt, Manuel Prinz, Vitalis Wiens, and Mohamad Yaser Jaradeh. Improving access to scientific literature with knowledge graphs. *Bibliothek Forschung und Praxis*, 44(3):516–529, 2020. doi:10.1515/bfp-2020-2042.

6 Khalid Belhajjame, Jun Zhao, Daniel Garijo, Matthew Gamble, Kristina M. Hettne, Raúl Palma, Eleni Mina, Óscar Corcho, José Manuél Gómez-Pérez, Sean Bechhofer, Graham Klyne, and Carole A. Goble. Using a suite of ontologies for preserving workflow-centric research objects. *J. Web Semant.*, 32:16–42, 2015. doi:10.1016/J.WEBSEM.2015.01.003.

7 Iz Beltagy, Kyle Lo, and Arman Cohan. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China, November 2019. Association for Computational Linguistics. doi:10.18653/v1/D19-1371.

8 Tim Berners-Lee, James Hendler, Lassila, and Ora. The Semantic Web. *Scientific American*, 284(5):34–43, 2001.

9 Elena Cabrio and Serena Villata. Five years of argument mining: a data-driven analysis. In Jérôme Lang, editor, *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 5427–5433. ijcai.org, 2018. doi:10.24963/IJCAI.2018/766.

10 Lucas Carstens and Francesca Toni. Towards relation based argumentation mining. In Claire Cardie, editor, *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 29–34, Denver, CO, June 2015. Association for Computational Linguistics. doi:10.3115/v1/W15-0504.

11 Oana Cocarascu and Francesca Toni. Identifying attack and support argumentative relations using deep learning. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1374–1379, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi:10.18653/v1/D17-1144.

12 Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi:10.18653/v1/N19-1423.

13 John Domingue, Dieter Fensel, and James A. Hendler, editors. *Handbook of Semantic Web Technologies*. Springer, 2011. doi:10.1007/978-3-540-92913-0.

14 Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel

Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul

Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024. doi:10.48550/arXiv.2407.21783.

15 Paul Groth, Andrew Gibson, and Jan Velterop. The anatomy of a nanopublication. *Inf. Serv. Use*, 30(1-2):51–56, 2010. doi:10.3233/ISU-2010-0613.

16 Steffen Herbold. Autorank: A python package for automated ranking of classifiers. *J. Open Source Softw.*, 5(48):2173, 2020. doi:10.21105/JOSS.02173.

17 Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, 1997. doi:10.1162/NECO.1997.9.8.1735.

18 Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b. *CoRR*, abs/2310.06825, 2023. doi:10.48550/arXiv.2310.06825.

19 Nora Kassner, Oyvind Tafjord, Ashish Sabharwal, Kyle Richardson, Hinrich Schuetze, and Peter Clark. Language models with rationality. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14190–14201, Singapore, December 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.emnlp-main.877.

20 Christian Kirschner, Judith Eckle-Kohler, and Iryna Gurevych. Linking the thoughts: Analysis of argumentation structures in scientific publications. In Claire Cardie, editor, *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 1–11, Denver, CO, June 2015. Association for Computational Linguistics. doi:10.3115/v1/W15-0501.

21 Tobias Kuhn, Albert Meroño-Peñuela, Alexander Malic, Jorrit H. Poelen, Allen H. Hurlbert, Emilio Centeno Ortiz, Laura I. Furlong, Núria Queralt-Rosinach, Christine Chichester, Juan M. Banda, Egon L. Willighagen, Friederike Ehrhart, Chris T. A. Evelo, Tareq B. Malas, and Michel Dumontier. Nanopublications: A growing resource of provenance-centric scientific linked data. In *14th IEEE International Conference on e-Science, e-Science 2018, Amsterdam, The Netherlands, October 29 - November 1, 2018*, pages 83–92. IEEE Computer Society, 2018. doi:10.1109/ESCIENCE.2018.00024.

22 Anne Lauscher, Goran Glavaš, and Simone Paolo Ponzetto. An argument-annotated corpus of scientific publications. In Noam Slonim and Ranit Aharonov, editors, *Proceedings of the 5th Workshop on Argument Mining*, pages 40–46, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi:10.18653/v1/W18-5206.

23 Anne Lauscher, Goran Glavaš, Simone Paolo Ponzetto, and Kai Eckert. Annotating arguments in scientific publications, 2018. URL: https://data.dws.informatik.uni-mannheim.de/sci-arg/annotation_guidelines.pdf.

24 John Lawrence and Chris Reed. Combining argument mining techniques. In Claire Cardie, editor, *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 127–136, Denver, CO, June 2015. Association for Computational Linguistics. doi:10.3115/v1/W15-0516.

25 John Lawrence, Jacky Visser, and Chris Reed. Harnessing rhetorical figures for argument mining. *Argument Comput.*, 8(3):289–310, 2017. doi:10.3233/AAC-170026.

26 Timothy Lebo, Satya Sahoo, and Deborah McGuinness. PROV-O: The PROV Ontology. https://www.w3.org/TR/prov-o/, 2013. W3C Recommendation 30 April 2013. Accessed: 2025-07-30.

27 Marco Lippi and Paolo Torroni. Argumentation mining: State of the art and emerging trends. *ACM Trans. Internet Techn.*, 16(2):10:1–10:25, 2016. doi:10.1145/2850417.

28 Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. What makes good in-context examples for GPT-3? In Eneko Agirre, Marianna Apidianaki, and Ivan Vulić, editors, *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online, May 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.deelio-1.10.

29 Kyle Lo, Zejiang Shen, Benjamin Newman, Joseph Chang, Russell Authur, Erin Bransom, Stefan Candra, Yoganand Chandrasekhar, Regan Huff, Bailey Kuehl, Amanpreet Singh, Chris Wilhelm, Angele Zamarron, Marti A. Hearst, Daniel Weld, Doug

Downey, and Luca Soldaini. PaperMage: A unified toolkit for processing, representing, and manipulating visually-rich scientific documents. In Yansong Feng and Els Lefever, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 495–507, Singapore, December 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.emnlp-demo.45.

30 Tobias Mayer, Elena Cabrio, Marco Lippi, Paolo Torroni, and Serena Villata. Argument mining on clinical trials. In Sanjay Modgil, Katarzyna Budzynska, and John Lawrence, editors, *Computational Models of Argument - Proceedings of COMMA 2018, Warsaw, Poland, 12-14 September 2018*, volume 305 of *Frontiers in Artificial Intelligence and Applications*, pages 137–148. IOS Press, 2018. doi:10.3233/978-1-61499-906-5-137.

31 Tobias Mayer, Elena Cabrio, and Serena Villata. Transformer-based argument mining for healthcare applications. In Giuseppe De Giacomo, Alejandro Catalá, Bistra Dilkina, Michela Milano, Senén Barro, Alberto Bugarín, and Jérôme Lang, editors, *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain, August 29 - September 8, 2020 - Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 2108–2115. IOS Press, 2020. doi:10.3233/FAIA200334.

32 Peter Bjorn Nemenyi. *Distribution-free multiple comparisons*. PhD thesis, Princeton University, 1963.

33 Allard Oelen, Mohamad Yaser Jaradeh, and Sören Auer. ORKG ASK: a neuro-symbolic scholarly search and exploration system. In Daniel Garijo, Anna Lisa Gentile, Anelia Kurteva, Andrea Mannocci, Francesco Osborne, and Sahar Vahdati, editors, *Joint Proceedings of Posters, Demos, Workshops, and Tutorials of the 20th International Conference on Semantic Systems co-located with 20th International Conference on Semantic Systems (SEMANTiCS 2024), Amsterdam, The Netherlands, September 17-19, 2024*, volume 3759 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2024. URL: https://ceur-ws.org/Vol-3759/paper7.pdf.

34 OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave

Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024. arXiv:2303.08774.

35 Juri Opitz and Anette Frank. Dissecting content and context in argumentative relation analysis. In Benno Stein and Henning Wachsmuth, editors, *Proceedings of the 6th Workshop on Argument Mining*, pages 25–34, Florence, Italy, August 2019. Association for Computational Linguistics. doi:10.18653/v1/W19-4503.

36 Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi:10.3115/v1/D14-1162.

37 Georgios Petasis and Vangelis Karkaletsis. Identifying argument components through TextRank. In Chris Reed, editor, *Proceedings of the Third Workshop on Argument Mining (ArgMining2016)*, pages 94–102, Berlin, Germany, August 2016. Association for Computational Linguistics. doi:10.18653/v1/W16-2811.

38 Jason Priem, Heather A. Piwowar, and Richard Orr. Openalex: A fully-open index of scholarly works, authors, venues, institutions, and concepts. *CoRR*, abs/2205.01833, 2022. doi:10.48550/arXiv.2205.01833.

39 Iyad Rahwan, Fouad Zablith, and Chris Reed. Laying the foundations for a world wide argument web. *Artif. Intell.*, 171(10-15):897–921, 2007. doi:10.1016/J.ARTINT.2007.04.015.

40 Iyad Rahwan, Fouad Zablith, and Chris Reed. Laying the Foundations for a World Wide Argument Web. *Artificial Intelligence*, 171(10-15):897–921, 2007. doi:10.1016/j.artint.2007.04.015.

41 Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics, 2019. doi:10.18653/V1/D19-1410.

42 Nils Reimers, Benjamin Schiller, Tilman Beck, Johannes Daxenberger, Christian Stab, and Iryna Gurevych. Classification and clustering of arguments with contextualized word embeddings. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 567–578, Florence, Italy, July 2019. Association for Computational Linguistics. doi:10.18653/v1/P19-1054.

43 Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton-Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. Code llama: Open foundation models for code. *CoRR*, abs/2308.12950, 2023. doi:10.48550/arXiv.2308.12950.

44 Florian Ruosch, Cristina Sarasua, and Abraham Bernstein. BAM: benchmarking argument mining on scientific documents. In Amir Pouran Ben Veyseh, Franck Dernoncourt, Thien Huu Nguyen, Walter Chang, and Viet Dack Lai, editors, *Proceedings of the Workshop on Scientific Document Understanding co-located with 36th AAAI Conference on Artificial Intelligence, SDU@AAAI 2022, Virtual Event, March 1, 2022*, volume 3164 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2022. URL: https://ceur-ws.org/Vol-3164/paper5.pdf.

45 Florian Ruosch, Cristina Sarasua, Chris Reed, and Abraham Bernstein. Toward the Argument Web of Science. In *Computational Models of Argument - Proceedings of COMMA 2024, Hagen, Germany, 18-20 September 2024*, Frontiers in Artificial Intelligence and Applications, 2024.

46 David M. Shotton. Cito, the citation typing ontology. *J. Biomed. Semant.*, 1(S-1):S6, 2010. URL: http://www.jbiomedsem.com/content/1/S1/S6.

47 Christian Stab and Iryna Gurevych. Identifying argumentative discourse structures in persuasive essays. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 46–56, Doha, Qatar, October 2014. Association for Computational Linguistics. doi:10.3115/v1/D14-1006.

48 Markus Stocker, Allard Oelen, Mohamad Yaser Jaradeh, Muhammad Haris, Omar Arab Oghli, Golsa Heidari, Hassan Hussein, Anna-Lena Lorenz, Salomon Kabenamualu, Kheir Eddine Farfar, Manuel Prinz, Oliver Karras, Jennifer D'Souza, Lars Vogt, and Sören Auer. Fair scientific information with the open&nbsp;research knowledge graph. *FAIR Connect*, 1:19–21, 2023. 1. doi:10.3233/FC-221513.

49 Simone Teufel, Jean Carletta, and Marc Moens. An annotation scheme for discourse-level argumentation in research articles. In Henry S. Thompson and Alex Lascarides, editors, *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, pages 110–117, Bergen, Norway, June 1999. Association for Computational Linguistics. URL: https://aclanthology.org/E99-1015.

50 C. J. van Rijsbergen. *Information Retrieval*. Butterworth, 1979.

51 Rosni Vasu, Cristina Sarasua, and Abraham Bernstein. SciHyp: A Fine-grained Dataset Describing Hypotheses and Their Components from Scientific Articles (Version V1) [Dataset], December 2023. Zenodo. https://doi.org/10.5281/zenodo.10298218.

52 Rosni Vasu, Cristina Sarasua, and Abraham Bernstein. Scihyp: A fine-grained dataset describing hypotheses and their components from sci-

entific articles. In Gianluca Demartini, Katja Hose, Maribel Acosta, Matteo Palmonari, Gong Cheng, Hala Skaf-Molli, Nicolas Ferranti, Daniel Hernández, and Aidan Hogan, editors, *The Semantic Web - ISWC 2024 - 23rd International Semantic Web Conference, Baltimore, MD, USA, November 11-15, 2024, Proceedings, Part III*, volume 15233 of *Lecture Notes in Computer Science*, pages 134–152. Springer, 2024. doi:10.1007/978-3-031-77847-6_8.

53 Jacky Visser, Rory Duthie, John Lawrence, and Chris Reed. Intertextual correspondence for integrating corpora. In Nicoletta Calzolari, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga, editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA). URL: https://aclanthology.org/L18-1554.

54 Douglas Walton. Argumentation theory: A very short introduction. In Guillermo Ricardo Simari and Iyad Rahwan, editors, *Argumentation in Artificial Intelligence*, pages 1–22. Springer, 2009. doi:10.1007/978-0-387-98197-0_1.

55 Douglas Walton and Nanning Zhang. The epistemology of scientific evidence. *Artif. Intell. Law*, 21(2):173–219, 2013. doi:10.1007/S10506-012-9132-9.

56 Kuansan Wang, Zhihong Shen, Chiyuan Huang, Chieh-Han Wu, Yuxiao Dong, and Anshul Kanakia. Microsoft academic graph: When experts are not enough. *Quantitative Science Studies*, 1(1):396–413, February 2020. doi:10.1162/qss_a_00021.

57 An Yang and Sujian Li. SciDTB: Discourse dependency TreeBank for scientific abstracts. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 444–449, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi:10.18653/v1/P18-2071.

A Regular Expressions for Identifying Citations

We used the following regular expressions to identify argumentative components that contain citations.

```
^\[.*?\]$
```

This matches square brackets with any content; for example, “[Smith et al., 2000]” or “[1].”

```
^.*?\[?(18|19|20)\d{2}(a|b|c|d|e)?\]?$
```

This matches patterns ending in years (1800–2099) plus optionally a letter or brackets; for example, “1992a” or “[2005].”

```
^\d?\d$
```

This matches matches one or two digits; for example, “1” or “13.”

```
^\d?\d-\d?\d$
```

This matches matches digit rangs, “1–2” or “9–11.”

```
^.{0,10} ?(5|6|7|8|9|0|1|2)\d$
```

This matches abbreviations and the last two digits of a year (50-29, space optional); for example, “EiHe92” or “RSB 23.”

B Discourse Indicators for Rule-Based Argumentative Relation Prediction

The discourse indicators used for the rule-based argumentative relation prediction are shown in Table 5. The type for the predicted relation is also indicated for each column.

Table 5 The list of discourse indicators used for the rule-based argumentative relation prediction taken from [24, p. 128].

Supports	**Attacks**
because	however
therefore	but
after	though
for	except
since	not
when	never
assuming	no
so	whereas
accordingly	nonetheless
thus	yet
hence	despite
then	
consequently	

Table 6 The dataset splits for the fine-tuning of the transformer-based argument miner [31].

Set	**Items**
Train	A03, A04, A05, A06, A07, A08, A12, A13, A14, A15, A17, A19, A20, A22, A23, A24, A25, A28, A32, A33, A35, A36, A38, A39
Val	A02, A11, A21, A31
Test	A01, A09, A10, A16, A18, A26, A27, A29, A30, A34, A37, A40

C Fine-Tuning of the Transformer-Based Argument Miner

We fine-tuned the transformer-based argument miner on the argumentative relation annotations of the Sci-Arg corpus [22]. To this end, we used the uncased SciBERT [7] with the following parameters. The maximum sequence length was set to 128, the batch size per GPU to 32, the learning rate was $2e-5$, and we trained for three epochs. The system used an Adam optimizer with an epsilon of $1e-8$. All other parameters used the default values, as specified in the API of the argument miner. For the data splits, we followed [22] with only the validation set being our own choice. The exact subsets are shown in Table 6, where `AXX` denotes the identifier of one of the 40 papers in the dataset.

D Prompts for the Large Language Model

Here, we show the prompts for the two approaches using Mistral [18] for the argumentative relation prediction.

For the zero-shot, the prompt looked as follows. x and y are replaced with the components to be predicted.

```
Classify the type of argumentative relationship between two given text spans.  The
relationship can be from the list [none, supports, attacks].  Reply with only one word.
COMPONENT A: x
COMPONENT B: y
RELATION:
```

Table 7 The abbreviations used for the different approaches.

Approach	Abbreviation
Rule-Based	diam
TRABAM	trabam
Zero-Shot LLM	llmam_zero
Few-Shot LLM	llmam_prompt
Zero-Shot LLM with Context	llmam_zero_context
Few-Shot LLM with Context	llmam_prompt_context

For the few-shot, the prompt looked as follows. x and y are replaced with the components to be predicted. a and b are replaced with the components of the example pair, and r with their relation. Furthermore, the $5\times\{\dots\}$ indicates that this block is repeated five times with five different example pairs and their relations. The examples pairs are drawn from the $66,668$ relations in the test set of the data based on the semantic similarity to the components to be classified. This similarity was computed with Sentence Transformer [41]. The class distribution of these in-context examples was as follows:

- noRel: $62,226$ examples
- supports: $3,644$ examples
- attacks: 798 examples

```
Classify the type of argumentative relationship between two given text spans. The
relationship can be from the list [none, supports, attacks]. Reply with only one word.
Examples:
5×
{COMPONENT A: a
COMPONENT B: b
RELATION: r}
COMPONENT A: x
COMPONENT B: y
RELATION:
```

E Statistical Significance Testing

Below, we report the p-values where we claim to have found significant differences for the results shown in Table 3 and in Table 4. For the comparison of the accuracy on the overall dataset for all systems, we show the mean rank differences in Table 8 and for the comparison of the intra- and inter-document setting in Table 9. Bold numbers indicate statistically significant differences ($p < 0.05$). The testing for the statistical significance of the results was conducted with Autorank [16]. For brevity, the different approaches use abbreviations which are explained in Table 7. Furthermore, the suffix indicates whether subsets ("-intra" for intra-document, "-inter" for inter-document) or the entirety of the dataset ("-all") was used. The detailed reports on the conducted tests for statistical significance, including all procedures and assumptions testing, are partially shown below, but all are available in the code repository.[2]

Table 8 Pairwise absolute mean rank differences from the Nemenyi test [32]. Values in bold indicate a significant difference (p < 0.05, based on a critical distance of 2.176).

Population	Rule-Based	TRABAM	Zero-Shot LLM	Few-Shot LLM	Zero-Shot LLM (with Context)	Few-Shot LLM (with Context)
Rule-Based		1.000	**4.167**	**4.000**	**3.167**	**2.667**
TRABAM	1.000		**3.167**	**3.000**	2.167	1.667
Zero-Shot LLM	**4.167**	**3.167**		0.167	1.000	1.500
Few-Shot LLM	**4.000**	**3.000**	0.167		0.833	1.333
Zero-Shot LLM (with Context)	**3.167**	2.167	1.000	0.833		0.500
Few-Shot LLM (with Context)	**2.667**	1.667	1.500	1.333	0.500	

Table 9 The p-values of the paired t-test for the comparison of the results in the intra- versus inter-document setting.

	p-Value
Rule-Based	0.977
TRABAM	**0.003**
Zero-Shot LLM	**0.000**
Few-Shot LLM	0.149
Zero-Shot LLM with Context	**0.000**
Few-Shot LLM with Context	0.131

Overall Dataset for All Systems

The statistical analysis was conducted for 6 populations with 12 paired samples. The family-wise significance level of the tests α=0.050. We failed to reject the null hypothesis that the population is normal for all populations (minimal observed p-value=0.139). Therefore, we assume that all populations are normal. We applied Bartlett's test for homogeneity and reject the null hypothesis (p=0.039) that the data is homoscedastic. Thus, we assume that our data is heteroscedastic. Because we have more than two populations and the populations are normal but heteroscedastic, we use the non-parametric Friedman test as omnibus test to determine if there are any significant differences between the mean values of the populations. We use the post-hoc Nemenyi test to infer which differences are significant. We report the mean value (M), the standard deviation (SD) and the mean rank (MR) among all populations over the samples. Differences between populations are significant, if the difference of the mean rank is greater than the critical distance CD=2.176 of the Nemenyi test. We reject the null hypothesis (p=0.000) of the Friedman test that there is no difference in the central tendency of the populations llmam_zero-all (M=0.113±0.032, SD=0.034, MR=5.167), llmam_prompt-all (M=0.109±0.041, SD=0.044, MR=5.000), llmam_zero_context-all (M=0.119±0.035, SD=0.037, MR=4.167), llmam_prompt_context-all (M=0.132±0.028, SD=0.030, MR=3.667), trabam-all (M=0.251±0.049, SD=0.052, MR=2.000), and diam-all (M=0.437±0.067, SD=0.073, MR=1.000). Therefore, we assume that there is a statistically significant difference between the median values of the populations. Based on the post-hoc Nemenyi test, we assume that there are no significant differences within the following groups: llmam_zero-all, llmam_prompt-all, llmam_zero_context-all, and llmam_prompt_context-all; llmam_zero_context-all, llmam_prompt_context-all, and trabam-all; trabam-all and diam-all. All other differences are significant.

Inter- Versus Intra-Document

For each of the four comparisons, we had two populations with 12 paired samples. Since BAM [44] reports the mean of the samples, we conducted the paired t-test, the p-values of which are shown in Table 9. As we fail to reject the null hypothesis for normal distribution for all populations,

Table 10 The p-values of the Wilcoxon Signed-Rank Test comparison for different settings ($N = 12$ per population).

Comparison	M_1 (SD_1)	Mdn_1	M_2 (SD_2)	Mdn_2	W	p
diam-intra vs. diam-inter	0.430 (± 0.083)	0.421	0.432 (± 0.135)	0.424	39.0	1.000
trabam-intra vs. trabam-inter	0.238 (± 0.059)	0.215	0.345 (± 0.100)	0.351	1.0	**0.001**
llmam_zero-intra vs. llmam_zero-inter	0.083 (± 0.028)	0.084	0.249 (± 0.065)	0.257	0.0	**0.000**
llmam_prompt-intra vs. llmam_prompt-inter	0.091 (± 0.052)	0.092	0.136 (± 0.079)	0.139	18.0	0.110
llmam_zero_context-intra vs. llmam_zero_context-inter	0.090 (± 0.031)	0.093	0.252 (± 0.069)	0.256	0.0	**0.000**
llmam_prompt_context-intra vs. llmam_prompt_context-inter	0.113 (± 0.043)	0.121	0.165 (± 0.082)	0.176	17.0	0.092
llmam_prompt-all vs. llmam_prompt_context-all	0.109 (± 0.044)	0.104	0.132 (± 0.030)	0.128	11.0	**0.027**
llmam_zero-all vs. llmam_zero_context-all	0.113 (± 0.034)	0.108	0.119 (± 0.037)	0.114	3.0	**0.002**

we also report the p-values of the more robust Wilcoxon Signed-Rank Test in Table 10. They yield the same results with respect to the statistical significance of the differences between the populations.

Rule-Based: Inter- Versus Intra-Document

The statistical analysis was conducted for 2 populations with 12 paired samples. The family-wise significance level of the tests α=0.050. No check for homogeneity was required because we only have two populations. We use the t-test to determine differences between the mean values of the populations and report the mean value (M) and the standard deviation (SD) for each population. We failed to reject the null hypothesis (p=0.977) of the paired t-test that the mean values of the populations diam-intra (M=0.430±0.062, SD=0.083) and diam-inter (M=0.432±0.101, SD=0.135) are equal. *Therefore, we assume that there is no statistically significant difference between the mean values of the populations.*

Argument Miner: Inter- Versus Intra-Document

The statistical analysis was conducted for 2 populations with 12 paired samples. The family-wise significance level of the tests α=0.050. No check for homogeneity was required because we only have two populations. We use the t-test to determine differences between the mean values of the populations and report the mean value (M) and the standard deviation (SD) for each population. We reject the null hypothesis (p=0.003) of the paired t-test that the mean values of the populations trabam-intra (M=0.238±0.044, SD=0.059) and trabam-inter (M=0.345±0.075, SD=0.100) are equal. *Therefore, we assume that the mean value of trabam-inter is significantly larger than the mean value of trabam-intra with a large effect size (d=-1.299).*

Zero-Shot LLM: Inter- Versus Intra-Document

The statistical analysis was conducted for 2 populations with 12 paired samples. The family-wise significance level of the tests α=0.050. No check for homogeneity was required because we only have two populations. We use the t-test to determine differences between the mean values of the populations and report the mean value (M) and the standard deviation (SD) for each population. We reject the null hypothesis (p=0.000) of the paired t-test that the mean values of the populations llmam_zero-intra (M=0.083±0.021, SD=0.028) and llmam_zero-inter (M=0.249±0.049, SD=0.065) are equal. *Therefore, we assume that the mean value of llmam_zero-inter is significantly larger than the mean value of llmam_zero-intra with a large effect size (d=-3.323).*

Prompted LLM: Inter- Versus Intra-Document

The statistical analysis was conducted for 2 populations with 12 paired samples. The family-wise significance level of the tests α=0.050. No check for homogeneity was required because we only have two populations. We use the t-test to determine differences between the mean values of the populations and report the mean value (M) and the standard deviation (SD) for each population. We failed to reject the null hypothesis (p=0.149) of the paired t-test that the mean values of the populations llmam_prompt-intra (M=0.091±0.039, SD=0.052) and llmam_prompt-inter (M=0.136±0.059, SD=0.079) are equal. *Therefore, we assume that there is no statistically significant difference between the mean values of the populations.*

Zero-Shot LLM with Context: Inter- Versus Intra-Document

The statistical analysis was conducted for 2 populations with 12 paired samples. The family-wise significance level of the tests α=0.050. No check for homogeneity was required because we only have two populations. We use the t-test to determine differences between the mean values of the populations and report the mean value (M) and the standard deviation (SD) for each population. We reject the null hypothesis (p=0.000) of the paired t-test that the mean values of the populations llmam_zero_context-intra (M=0.090±0.023, SD=0.031) and llmam_zero_context-inter (M=0.252±0.052, SD=0.069) are equal. *Therefore, we assume that the mean value of llmam_zero_context-inter is significantly larger than the mean value of llmam_zero_context-intra with a large effect size (d=-3.022).*

Few-Shot LLM with Context: Inter- Versus Intra-Document

The statistical analysis was conducted for 2 populations with 12 paired samples. The family-wise significance level of the tests α=0.050. No check for homogeneity was required because we only have two populations. We use the t-test to determine differences between the mean values of the populations and report the mean value (M) and the standard deviation (SD) for each population. We failed to reject the null hypothesis (p=0.131) of the paired t-test that the mean values of the populations llmam_prompt_context-intra (M=0.113±0.032, SD=0.043) and llmam_prompt_context-inter (M=0.165±0.061, SD=0.082) are equal. *Therefore, we assume that there is no statistically significant difference between the mean values of the populations.*

Zero-Shot LLM: Context Versus No Context

The statistical analysis was conducted for 2 populations with 12 paired samples. The family-wise significance level of the tests α=0.050. No check for homogeneity was required because we only have two populations. We use the t-test to determine differences between the mean values of the populations and report the mean value (M) and the standard deviation (SD) for each population. We reject the null hypothesis (p=0.008) of the paired t-test that the mean values of the populations llmam_zero-all (M=0.113±0.026, SD=0.034) and llmam_zero_context-all (M=0.119±0.028, SD=0.037) are equal. *Therefore, we assume that the mean value of llmam_zero_context-all is significantly larger than the mean value of llmam_zero-all with a negligible effect size (d=-0.173).*

Few-Shot LLM: Context Versus No Context

The statistical analysis was conducted for 2 populations with 12 paired samples. The family-wise significance level of the tests α=0.050. No check for homogeneity was required because we only have two populations. We use the t-test to determine differences between the mean values of the populations and report the mean value (M) and the standard deviation (SD) for each population.

Table 11 The results of the experiments on the overall dataset. We report the class-based F1-score, precision, and recall.

Approach		F1	Precision	Recall
Rule-Based	Supports	0.346	0.929	0.212
	Attacks	0.342	0.318	0.371
Argument Miner	Supports	0.309	0.882	0.187
	Attacks	0.196	0.708	0.114
Zero-Shot LLM	Supports	0.842	0.792	0.899
	Attacks	0.010	1.000	0.005
Zero-Shot LLM + Context	Supports	0.863	0.790	0.950
	Attacks	0.010	0.286	0.005
Few-Shot LLM	Supports	0.182	0.936	0.101
	Attacks	0.020	0.667	0.010
Few-Shot LLM + Context	Supports	0.380	0.899	0.241
	Attacks	0.010	0.400	0.005

Table 12 The results of the experiments on the intra-document subset. We report the class-based F1-score, precision, and recall.

Approach		F1	Precision	Recall
Rule-Based	Supports	0.329	0.888	0.202
	Attacks	0.376	0.382	0.370
Argument Miner	Supports	0.308	0.865	0.187
	Attacks	0.185	0.694	0.107
Zero-Shot LLM	Supports	0.770	0.700	0.855
	Attacks	0.010	1.000	0.005
Zero-Shot LLM + Context	Supports	0.802	0.704	0.930
	Attacks	0.010	0.400	0.005
Few-Shot LLM	Supports	0.140	0.876	0.076
	Attacks	0.020	0.800	0.010
Few-Shot LLM + Context	Supports	0.329	0.830	0.205
	Attacks	0.010	0.400	0.010

We reject the null hypothesis (p=0.012) of the paired t-test that the mean values of the populations llmam_prompt-all (M=0.109±0.033, SD=0.044) and llmam_prompt_context-all (M=0.132±0.022, SD=0.030) are equal. *Therefore, we assume that the mean value of llmam_prompt_context-all is significantly larger than the mean value of llmam_prompt-all with a medium effect size (d=-0.618).*

F Class-Based Performance

Table 11 shows the class-based performance for the four approaches on the overall dataset. Tables 12 and 13 show the class-based results for the intra- and inter-document subset, respectively. The class distribution is shown in Table 14.

Table 13 The results of the experiments on the inter-document subset. We report the class-based F1-score, precision, and recall.

Approach		F1	Precision	Recall
Rule-Based	Supports	0.375	1.000	0.231
	Attacks	0.024	0.012	1.000
Argument Miner	Supports	0.429	1.000	0.273
	Attacks	0.000	0.000	0.000
Zero-Shot LLM	Supports	0.988	0.998	0.979
	Attacks	0.000	0.000	0.000
Few-Shot LLM	Supports	0.253	1.000	0.145
	Attacks	0.000	0.000	0.000
Zero-Shot LLM + Context	Supports	0.993	0.998	0.988
	Attacks	0.000	0.000	0.000
Few-Shot LLM + Context	Supports	0.470	1.000	0.307
	Attacks	0.000	0.000	0.000

Table 14 The distribution of the classes for the overall dataset and the intra- and inter-document subsets.

Class	Overall	Intra	Inter
Supports	1592	1025	567
Attacks	404	403	1
Total	1996	1428	568

G Statistics of Misclassified Samples

Table 15 shows the distributions of the misclassified samples per relation type for the different approaches. Each column indicates the real class of the samples, and each row shows the assigned relation type by the approaches. Since the zero-shot LLM approach did not only produce valid labels but also a variety of responses, Tables 15c and 15d have an additional row (*Various*) where these predictions are pooled.

H Technical Infrastructure and Computational Budget

All experiments were performed on a machine with eight NVIDIA GeForce RTX 4090 GPUs and one AMD EPYC 9124 3.0GHz 16-core CPU. Only a single GPU was used at a time for each experiment run, where applicable. Any experiment not leveraging the GPU was performed on the CPU: this only applies to the Argument Mining using the presence of discourse indicators.

The computational budget for all experiments was approximately 120 GPU hours.

Table 15 Distribution of the misclassified samples per relation type for all six approaches evaluated. The gold standard relation is shown in the columns, and the predicted system relation is shown in the rows. Subfigure (a) is the rule-based approach, (b) the argument miner, (c) the zero-shot LLM, (d) the zero-shot LLM with context, (e) the few-shot LLM, and (f) the few-shot LLM with context.

(a) Rule-based approach.

System \ Gold	Attacks	Supports	No Relation
Attacks	–	0.257	0.440
Supports	0.102	–	0.560
No Relation	0.898	0.743	–

(b) Argument miner [31] approach.

System \ Gold	Attacks	Supports	No Relation
Attacks	–	0.015	0.134
Supports	0.112	–	0.866
No Relation	0.888	0.985	–

(c) Zero-shot LLM approach.

System \ Gold	Attacks	Supports	No Relation
Attacks	–	0.000	0.000
Supports	0.935	–	0.911
No Relation	0.005	0.087	–
Various	0.060	0.913	0.089

(d) Zero-shot LLM approach with context.

System \ Gold	Attacks	Supports	No Relation
Attacks	–	0.063	0.004
Supports	0.998	–	0.953
No Relation	0.000	0.051	–
Various	0.002	0.886	0.043

(e) Few-shot LLM approach.

System \ Gold	Attacks	Supports	No Relation
Attacks	–	0.001	0.034
Supports	0.028	–	0.966
No Relation	0.972	0.999	–

(f) Few-shot LLM approach with context.

System \ Gold	Attacks	Supports	No Relation
Attacks	–	0.002	0.017
Supports	0.107	–	0.983
No Relation	0.893	0.998	–

LLM-Supported Manufacturing Mapping Generation

Wilma Johanna Schmidt[1]
Corporate Research, Robert Bosch GmbH, Renningen, Germany
AG Corporate Semantic Web, Freie Universität Berlin, Germany

Irlan Grangel-González
Corporate Research, Robert Bosch GmbH, Renningen, Germany

Adrian Paschke
Data Analytic Center (DANA), Fraunhofer Institute FOKUS, Berlin, Germany
AG Corporate Semantic Web, Freie Universität Berlin, Germany

Evgeny Kharlamov
Corporate Research, Robert Bosch GmbH, Renningen, Germany

Abstract

In large manufacturing companies, such as Bosch, that operate thousands of production lines with each comprising up to dozens of production machines and other equipment, even simple inventory questions such as of location and quantities of a particular equipment type require non-trivial solutions. Addressing these questions requires to integrate multiple heterogeneous data sets which is time consuming and error prone and demands domain as well as knowledge experts. Knowledge graphs (KGs) are practical for consolidating inventory data by bringing it into the same format and linking inventory items. However, the KG creation and maintenance itself pose challenges as mappings are needed to connect data sets and ontologies. In this work, we address these challenges by exploring LLM-supported and context-enhanced generation of both *YARRRML* and *RML* mappings. Facing large ontologies in the manufacturing domain and token limitations in LLM prompts, we further evaluate ontology reduction methods in our approach. We evaluate our approach both quantitatively against reference mappings created manually by experts and, for *YARRRML*, also qualitatively with expert feedback. This work extends the exploration of the challenges with LLM-supported and context-enhanced mapping generation *YARRRML* [18] by comprehensive analyses on *RML* mappings and an ontology reduction evaluation. We further publish the source code of this work. Our work provides a valuable support when creating manufacturing mappings and supports data and schema updates.

2012 ACM Subject Classification Computing methodologies → Semantic networks

Keywords and phrases Mapping Generation, Knowledge Graph Construction, Ontology Reduction, RML, YARRRML, LLM, Manufacturing

Digital Object Identifier 10.4230/TGDK.3.3.5

Category Use Case

Supplementary Material
Software (Source code, example data set): `https://github.com/boschresearch/myamr_tgdk` [19]

Funding *Evgeny Kharlamov*: Partially supported by the EU Projects Graph Massivizer (GA 101093202), enRichMyData (GA 101070284), and SMARTY (GA 101140087).

Acknowledgements The authors would like to thank Ishan Dindorkar for his support in setting up the technical environment.

Received 2025-06-02 **Accepted** 2025-11-17 **Published** 2025-12-10

Special Issue Use-Case Articles

[1] corresponding author

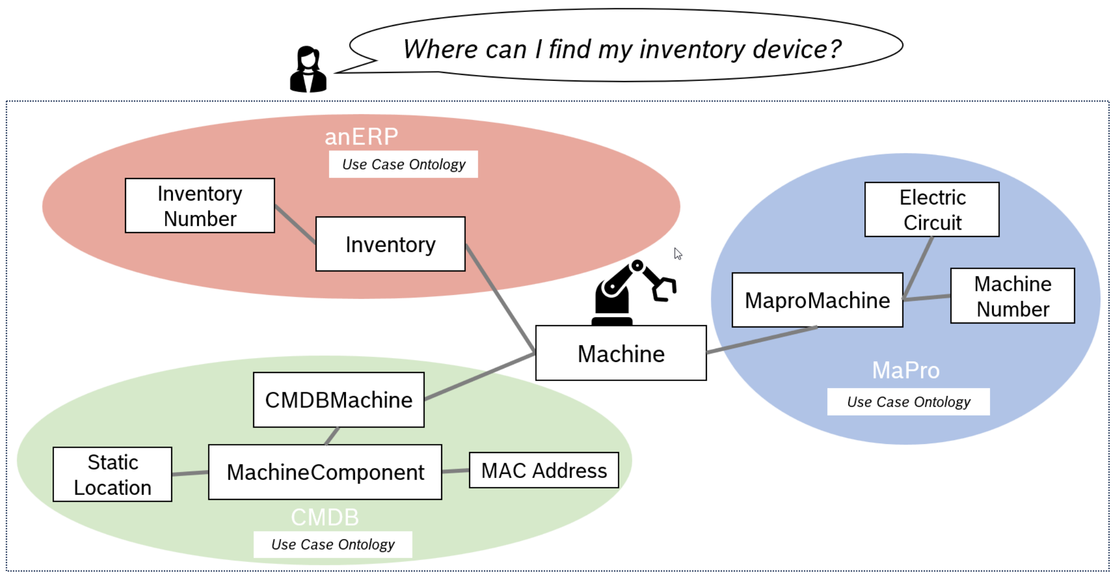


Figure 1 Motivational question and illustration of distributed data. Data belonging to the same machine is stored in different, use case specific sources for an enterprise resource planning, configuration management, and manufacturing project management.

1 Introduction

In large manufacturing companies, such as Bosch, data related to a specific machine inventory item is stored in distributed databases, which are built and curated to serve the needs for one or multiple functions within the company, see Figure 1. Cross-functional questions, e.g., linking physical location and financial information, often lead to *treasure hunting* with laborious human efforts, experts' alignments, and delays when looking for the answer. While efforts increase with growing data volumes, the pressure on fast and sound decisions in the manufacturing domain rises.

The semantic approach has proven to be successful in bringing heterogeneous data to a common format and interlinking datasets, e.g., tables from relational databases, with the help of ontologies and Knowledge Graphs (KGs). Indeed, it has already been adopted in various industrial sectors such as energy [12], manufacturing [17], e-commerce [1] and others. In particular, Bosch has been relying on ontologies and KGs in many applications ranging from analyses of discrete manufacturing [29] to autonomous driving [22] and combinatorial optimization [5]. Moreover, Bosch has been scaling the use of semantics in various large projects such as the European data infrastructure project *Gaia-X*[2] and its derivative *Catena-X*.

In these applications the key task is to link a domain ontology to the original data with the help of declarative mappings and then execute the mappings in order to create a domain KG. The creation and maintenance of manufacturing mappings is non-trivial. Especially, domain-specific vocabulary, updates in data and data structure as well as complex facts and relations across different data silos pose challenges. As of now, it is common practice to create them manually. For this, a domain expert and a knowledge expert are required and they align during the mapping generation. Mappings must be conform to the ontology of the KG. The domain expert understands which real object or concept is actually referenced by the label in the data set, whereas the knowledge expert retrieves related ontologies, understands the ontology syntax, and creates syntactically correct mappings.

[2] `https://gaia-x.eu/news-press/the-role-of-ontologies-in-gaia-x/`

YARRRML has been introduced as a human-readable representation of RDF mappings and provides an alternative to the standard mapping languages *Relational Database to RDF Mapping Language (R2RML)*[3] and *RDF Mapping Language (RML)*[4]. Van Assche *et al.* [23] assess in a systematic literature review, *eight* mapping languages excluding *YARRRML*, noting that "it was excluded because it is published as a demo", yet acknowledging its widespread use in practice. We can second the industrial use of *YARRRML*, e.g., at several plants at Bosch, due to its vendor independence as well as further enhancements (see e.g. [10], [24]). Van Assche *et al.* [23] assess *RML* which is an extension of the W3C standard language *R2RML* from databases only to further structured sources such as *.csv* or *.json*. High quality mappings are of core importance for a functional manufacturing KG, yet their creation remains, with both *RML* or human-friendly *YARRRML*, challenging. We explore mapping generation on each of the two languages in our work.

LLMs in a manufacturing context often work with unstructured data such as text (e.g., error descriptions) or images (e.g., optical inspection of parts). Research on neuro-symbolic KG construction in manufacturing ([20]) shows that most such approaches are conducted on unstructured data sources. Exploration of neuro-symbolic AI approaches for creating mappings for KG construction in manufacturing is still missing. As LLMs with their well-known strength in generating text, and also code, from unstructured text rapidly evolve, it is promising to explore their capabilities of dealing with structured data, e.g., mapping tasks [11]. However, varying accuracy, token size limitations and missing domain-focus of LLMs require advanced generation workflows and configurations to realize the applicability of this approach. Retrieval-Augmented Generation (RAG) approaches and neuro-symbolic architectures such as semantically context-enhanced prompts can increase accuracy and domain-focus. Additionally, ontology reduction methods show means of reducing the prompt size, while potentially improving accuracy. We explore this in our work. Our guiding question is:

Can LLM-supported and context-enhanced mapping generation support human expert mapping creation for manufacturing KGs?

This work focuses on addressing this question and the contributions of this paper are summarized as follows:

1. We provide the first exploratory approach on LLM-supported and context-enhanced mapping generation for KGs as support for knowledge experts in a manufacturing company.
2. We propose a novel combination of context-based ontology reduction and a RAG-based approach for automatic mapping generation to create manufacturing KGs.
3. We follow a pragmatic evaluation approach for mapping quality on manufacturing KGs and apply our method on several configurations against gold-standard mapping references which are created by domain and technical experts. By this, we obtain comprehensive insights on influencing factors for high quality manufacturing mapping creation.

The further parts of this paper are structured as follows: We introduce the use case and existing challenges in Section 2. In Section 3, we propose our approach for manufacturing mapping generation and explain the experimental implementations. We present our results on LLM-supported and context-enhanced mapping generation and lessons learned in Section 4 and briefly discuss related work in Section 5. We conclude with future work in Section 6.

[3] `https://www.w3.org/TR/r2rml/`, `https://www.w3.org/2001/sw/wiki/RDB2RDF`
[4] `https://rml.io/specs/rml/`

2 Use Case

In this section, we outline the use case and challenges in manufacturing inventory identification. Currently in Bosch, process experts and data analysts have no common shared and accessible data base that provides information about machines executing the same process. Data items for a single machine are spread across different silos. If the relevant information is available and enriched with various other influencing factors, process experts can roll out solutions easier and faster. For instance, to be able to answer the following question: "How many winding machines are there in a certain plant"? data from different sources needs to be integrated. To that end, a KG has been built to semantically integrate and harmonize data to cope with such questions. Next, we list the main competency questions (*CQs*) for the KG creation and maintenance. The questions are a prioritized excerpt from a list of questions from practice. The questions were originally collected by domain experts for the overall ontology- and graph-based approach and the selected questions address all three manufacturing ontologies from our scenario. For validation of their currentness, the questions have additionally been reviewed by a subject expert. The *CQs* are:

- **CQ 1:** How many machines of a specific type are located in a given plant?
- **CQ 2:** Where is the machine inventory located that is allocated on my cost center?
- **CQ 3:** Is the insecure machine control component still used in other stations within the plant?
- **CQ 4:** What are all financial information to a line with only the machine identifier given?
- **CQ 5:** What is the unique identifier number to a CMDB asset?

Over time, changes are done in the current data set structure and further data sets will need to be added, so that answers to these questions are up-to-date: the KG supports these tasks pragmatically. For this, mapping generation and updates are needed. With growing data volumes and ontology sizes as well as the need for more efficiency, (semi-)automated mapping generation approaches are needed. However, multiple challenges occur in manufacturing mapping generation in practice if mappings are generated automatically, e.g., with the support of LLMs:

- **Challenge 1:** How to feed a manufacturing ontology into a prompt?
- **Challenge 2:** Can context-based ontology reduction methods improve mapping generation for manufacturing KGs?
- **Challenge 3:** Can prompt context enhancements improve the quality of the generated mapping?
- **Challenge 4:** Which sample size of a data set is needed for a generated mapping of high quality?

Next, we describe the data sources required to answer the main competency questions.

2.1 Data sources

In this part, we describe the data sources of *MaPro*, *CMDB*, and *anERP* in detail, which are identified as relevant for our example.

MaPro. The Manufacturing Project Management (*MaPro*) is a data source of the special machinery that covers project management details, e.g., timeline and capacity planning. It contains information for mechanical construction, e.g., machines of a production line and the link to the plans.

CMDB. The Configuration Management Database (*CMDB*) is a database for documenting and linking hardware and software assets, i.e., configuration items, e.g., users, relationships between assets or IP addresses used. In the context of this work it provides, e.g., the IP and MAC address from the machines in the manufacturing environment.

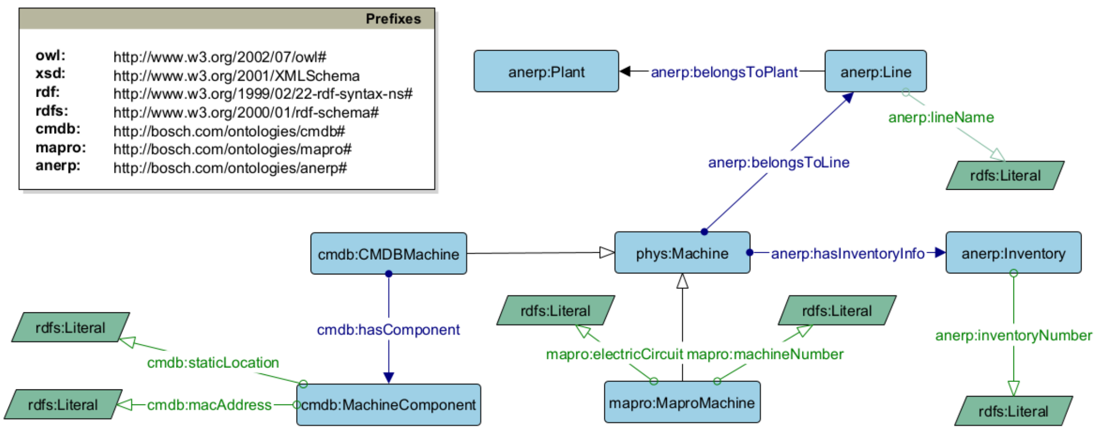


Figure 2 Main classes and properties of the MaPro, CMDB, and anERP ontologies. Illustration of our use case relevant classes, properties and property types, and the object properties. Prefixes are shown as supplements.

anERP. The *anERP* is an analytics platform for enterprise supply chain data. For this particular project, the anERP source enables the collection of financial data of the assets, i.e., machines and machine components. Furthermore, it contains inventory data of the assets.

2.2 Ontologies

In this section, we describe the structure of the ontologies used to build the KG. While the ontologies are core for our mapping generation (cf. Figure 2), they are proprietary and we may not share them publicly. However, we will provide quantifiable details on these ontologies in Section 3. The three ontologies employed in this work model the domains of datasets *MaPro*, *CMDB*, and *anERP*, respectively. The ontologies are based on the Core Information Model for Manufacturing (CIMM) ontologies [8] and comparable in their structure as they describe a similar domain. The CIMM ontologies provide a framework to reuse most important concepts from the manufacturing domain, e.g., machines, components, plants, lines, among others[5]. While each of our ontologies targets the corresponding dataset-relevant perspective from the manufacturing domain, they are rather simple w.r.t. hierarchy, number of classes etc. as their main intention is to describe the data from the data sources. They all share the formalization in the sublanguage *Description Logics (DL)* of the *Web Ontology Language (OWL)*[6]. Our ontologies comprise entries of type `owl:Class`, `owl:ObjectProperty`, and `owl:DatatypeProperty`. While classes mainly contain a label information, both property types may contain next to label, type, and comments also domain and range restrictions.

3 Approach for LLM-Supported and Context-based Mapping Generation

In this section, we introduce *MYAM+R*, a **M**anufacturing **YA**RRRML **M**apping generation approach, which also support *RML* mappings. It comprises four input modules, an ontology reduction module, the core mapping module which generates a mapping, an evaluation module

[5] `https://github.com/eclipse-esmf/esmf-manufacturing-information-model`
[6] `https://www.w3.org/TR/owl-features/`

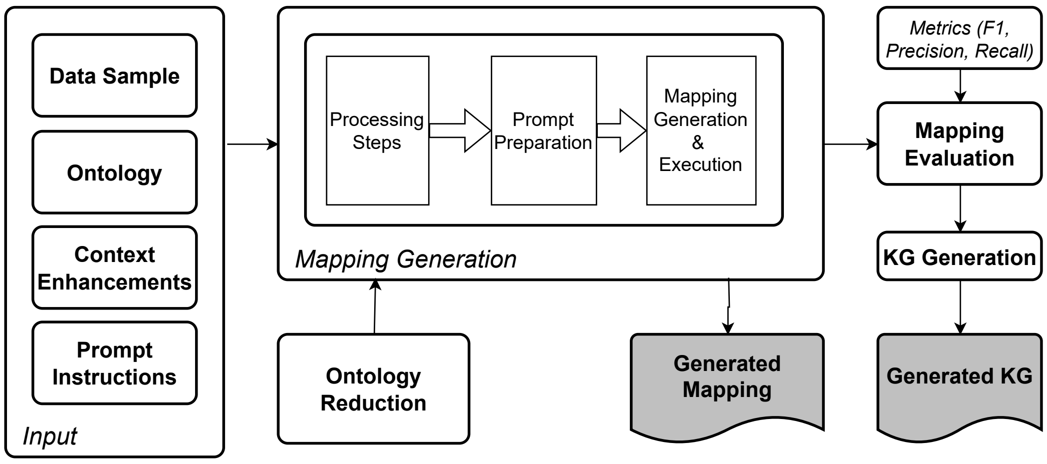


Figure 3 Manufacturing Mapping Generation Architecture (*MYAM+R*) for *YARRRML* and *RML* mappings. Illustration of the *MYAM+R* modules for input (data sample, ontology, context enhancements, and prompt instructions), ontology reduction, mapping generation, evaluation, and KG generation.

and a KG generation module. In Figure 3 we show an overview of *MYAM+R*. In the following, we explain all modules and process steps. We conclude this section with implementation details of *MYAM+R*.

3.1 *MYAM+R* Modules

First, we introduce the input modules which provide (1) a data sample, (2) an ontology, (3) prompt context enhancements, and (4) prompt instructions.

Data Sample. This module provides data from a relational database, described in Section 2.1, in *csv* format. The data size configuration, i.e., the portion of the data used for mapping generation, is specified with a parameter in *MYAM+R*. *MYAM+R* retrieves the full data set and then crops it accordingly to the configured setting. An example snippet is given in Figure 7.

Ontology. This module provides an ontology file relevant for the provided data set in *turtle* format. To meet prompt size restrictions, *MYAM+R* contains an ontology reduction module which we introduce later in this section. An example ontology snippet is given in Figure 7.

Enhancements. This module provides a prompt context enhancement as a *txt* file. The enhancements can be of different type, e.g., generic mapping templates or few shot examples. We show generic mapping templates for both *YARRRML* and *RML*, independent of data input in the prompt, yet teaching the syntax, in the following.

We further show generic one-shot examples which cover a correct mapping fragment for a given data input for each mapping language in Figures 5 and 6. In Section 3.2 we provide details on our enhancement implementation. The enhancement configuration is specified with a parameter in *MYAM+R*. The mapping generation module retrieves the context enhancement from this enhancement module based on the parameter value.

Prompt Instructions. This module provides instructions for the final mapping generation prompt.

```
prefixes:
  mach: <http://example.com/MachineOntology#>

mappings:
  machine:
    sources:
      - [examples/data.csv]
    s: mach:$(MachineID)
    po:
      - [a, mach:Machine]
      - [mach:machineId, $(MachineID)]
      - [mach:machineName, $(MachineName)]
      - p: mach:hasComponent
        o:
        - mapping: component

  component:
    sources:
      - [examples/data/machine_component.csv]
    s: mach:$(ComponentID)
    po:
      - [a, mach:Component]
      - [mach:componentId, $(ComponentID)]
      - [mach:componentName, $(ComponentName)]
```

```
@prefix rr: <http://www.w3.org/ns/r2rml#> .
@prefix rml: <http://semweb.mmlab.be/ns/rml#> .
@prefix mach: <http://example.org/MachineOntology#> .

<#MachineMapping>
    a rr:TriplesMap ;
    rml:logicalSource [
        rml:source "examples/data.csv" ;
        rml:referenceFormulation ql:CSV     ] ;
    rr:subjectMap [
        rr:template
        ↪  "http://example.org/MachineOntology/
        ↪  {MachineID}" ;
        rr:class ont:Machine     ] ;
    rr:predicateObjectMap [
        rr:predicate ont:machineId ;
        rr:objectMap [ rml:reference "MachineID" ] ] ;
    rr:predicateObjectMap [
        rr:predicate ont:machineName ;
        rr:objectMap [ rml:reference "MachineName" ] ]
        ↪  ;
    rr:predicateObjectMap [
        rr:predicate ont:hasComponent ;
        rr:objectMap [
            rr:parentTriplesMap <#ComponentMapping>  ]
        ↪  ] .
<#ComponentMapping>
    a rr:TriplesMap ;
    rml:logicalSource [
        rml:source "examples/data.csv" ;
        rml:referenceFormulation ql:CSV     ] ;
    rr:subjectMap [
        rr:template
        ↪  "http://example.org/MachineOntology/
        ↪  {ComponentID}" ;
        rr:class ont:Component     ] ;
    rr:predicateObjectMap [
        rr:predicate ont:componentId ;
        rr:objectMap [ rml:reference "ComponentID" ]  ]
        ↪  ;
    rr:predicateObjectMap [
        rr:predicate ont:componentName ;
        rr:objectMap [ rml:reference "ComponentName" ]
        ↪  ] .
```

Figure 4 Mapping Templates in *YARRRML* (left) and *RML* (right).

Ontology Reduction. This module takes data and ontology as an input and returns a reduced ontology. This module is needed because the provided manufacturing ontology contains elements which are not relevant for the mapping of the specific input data. This module supports in our implementation *three* reduction approaches, one generic and two content-based ones.

Mapping Generation. This module provides the core functionalities of *MYAM+R* and interacts with all other modules. *MYAM+R* supports different experimental settings and can be configured with parameters for convenient execution. Based on the parameter values, *MYAM+R* preprocesses the data, reduces the ontology, collects a prompt enhancement, and prompt instructions. The mapping generation module creates a prompt template with the collected ingredients as a prompt with input variables and sends this to an LLM. Finally, *MYAM+R* processes and validates the response and sends the generated mapping to the evaluation module. We call this step *mapping generation* as it is building upon the existing, reduced ontology in our previous step. The mapping is generated on the basis of this ontology. So, an alignment and consistency with this fixed ontology are ensured. We validate the generated mapping in an evaluation part (see next step) against a consistent, manually created gold standard mapping.

```
[
    {
        "input_ontology": "@prefix owl: <http://www.w3.org/2002/07/owl#> .
           @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
           @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
           @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
           @prefix ont: <http://example.com/PlantOntology#> .

           ont: rdf:type owl:Ontology ;
           rdfs:label \"plant-ontology\" ;
           .

           ont:Plant
           rdf:type owl:Class ;\
           rdfs:label \"ONT Plant\"@en ;
           rdfs:comment \"Describes the Plant entity in the ONT data base.\"@en .

           ont:plantName
           rdf:type owl:DatatypeProperty ;
           rdfs:label \"plant name\" ;
           rdfs:range xsd:string ;
           rdfs:domain ont:Plant .",
        "input_data": "PlantID,PlantName
        1,Plant_A
        2,Plant_B",
        "result_mappings": "...(see YARRML and RML result mappings figures)..."
    }
]
```

Figure 5 One Shot Example Input Data and Ontology.

Evaluation. This module receives a generated mapping, a reference mapping, and an ontology as input and returns evaluation scores for the generated mapping. The specific evaluation is described in Section 4.

KG Generation. This module generates a KG based on the generated mapping and a data set as input.

3.2 *MYAM+R* Implementation

MYAM+R is implemented in Python 3.13. For the mapping generation execution and for the LLM-based ontology reduction, we choose *gpt-4-turbo (version:1106-Preview)*. Aiming at a solution close to practice, we evaluate on a pre-trained LLM which could be available in large companies. A typical user in such an enterprise could get API access while avoiding steps of deployment and a potential training or fine-tuning of a base model (which would include dependencies on involvement of data specialists). At the same time, the user's task can become easier as he or she benefits from high quality output as most LLMs with higher payment models come with higher performance. Next, we describe the configurations that we explore in our experiments to address the challenges described in Section 2.

Data Sample. We run our experiments on data samples from *MaPro*, *CMDB*, and *anERP*. The metrics of these sets are given in Table 1. We set different configurations for the input: 1) only column headers, 2) column headers and first 20 data value rows, and 3) column headers and first 200 data value rows.

```
"results_mappings":
    "prefixes:
    @prefix ont:
    ↪  <http://example.com/
    ↪  PlantOntology#> .

    mappings:
      plant:
        sources:
           - [examples/
            ↪  data.csv]
    s: ont:$(PlantID)
    po:
      - [a, ont:Plant]
      - [ont:plantName,
       ↪  $(PlantName)]. "
```

```
"result_mappings": "@prefix rr: <http://www.w3.org/ns/r2rml#> .
    @prefix rml: <http://semweb.mmlab.be/ns/rml#> .
    @prefix ql: <http://semweb.mmlab.be/ns/ql#> .
    @prefix ont: <http://example.com/PlantOntology#> .

    <#PlantMapping>\n    a rr:TriplesMap ;
        rml:logicalSource [
            rml:source \"examples/data.csv\" ;
            rml:referenceFormulation ql:CSV   ] ;
        rr:subjectMap [
           rr:template \"http://example.com/PlantOntology#{PlantID}\" ;
           rr:class ont:Plant          ] ;
        rr:predicateObjectMap [
            rr:predicate ont:plantName ;
            rr:objectMap [
                rml:reference \"PlantName\"   ]   ]   ."
```

Figure 6 One Shot Example Result Mappings in *YARRRML* (left) and *RML* (right).

```
PlantID,PlantName,LineID,LineName
1,Plant_A,101,Line_X1
1,Plant_A,102,Line_X2
2,Plant_B,103,Line_Y1
2,Plant_B,104,Line_Y2
3,Plant_C,105,Line_Z1
3,Plant_C,106,Line_Z2
4,Plant_D,107,Line_W1
4,Plant_D,108,Line_W2
5,Plant_E,109,Line_V1
5,Plant_E,110,Line_V2
```

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix ont: <http://example.com/PlantOntology#> .

ont:PlantOntology a owl:Ontology ;
  owl:versionInfo "1.0" .

ont:Plant a owl:Class ;
  rdfs:label "Plant" ;
  rdfs:comment "A plant is a facility where production takes place." .
ont:Line a owl:Class ;
  rdfs:label "Line" ;
  rdfs:comment "A line is a production line within a plant." .
ont:hasLine a owl:ObjectProperty ;
  rdfs:label "has line" ;
  rdfs:comment "A plant has one or more lines." ;
  rdfs:domain ont:Plant ;
  rdfs:range ont:Line .
ont:plantID a owl:DatatypeProperty ;
  rdfs:label "plant ID" ;
  rdfs:comment "A unique identifier for a plant." ;
  rdfs:domain ont:Plant ;
  rdfs:range xsd:integer .
```

Figure 7 Example data source (left) and corresponding ontology snippet (right).

Ontology. The metrics of the corresponding ontologies to the sources are listed in Table 1. The files in our approach have been selected by a knowledge expert.

Context Enhancement. We explore zero shot, with and without a generic *YARRRML* mapping template, and few shot examples. The zero shot example with template contains a *YARRRML* mapping independent of the data source. The few shot examples are data-source- and ontology-specific and are manually created by domain and knowledge experts. For each data source we use one file of examples in *JSON* format. We implement the few shots with a RAG approach; *MYAM+R* retrieves the example file, vectorizes the examples as strings, embeds and stores them into a vector store. We select with the preprocessed, i.e., potentially cropped input data, and reduced ontology only the one most similar example and further process it in *MYAM+R*.

Prompt Instructions. Following common guidelines, we apply concise prompt instructions for the LLM to generate a *YARRRML* mapping. The instructions include a role *("You are a Senior Manufacturing Knowledge Expert...")*, an input description, and specific instructions depending on the shot variant. E.g., a zero shot prompt with a template states that the template is only

Table 1 Metrics of data sets and ontologies. *Key: DP – Datatype Properties, OP – Object Properties.*

Data Source	**Data Set**		**Ontology**			
	Columns	Rows	Classes	DP	OP	Namespaces
MaPro	17	2,215	4	13	4	5
CMDB	16	18,082	5	15	2	5
anERP	17	15,451	3	7	2	5

relevant for syntax, whereas a few shot prompt suggests to also learn from the content of the given mapping for the target one. The instructions are data- and ontology-independent. The prompt instructions are included in the published repository[7] to support reproducibility.

Ontology Reduction Methods. Ontology summarization [27], modularization [13], and matching [15] are important research topics[8], yet for the sake of pragmatic mapping generation we do not dive deeper into the current state of research. Instead, we address this in future work, and next, describe *three* reduction approaches which we evaluate in our experiments.

- **Naive:** It simplifies ontologies to specific elements associated with the input data. In our case, only the classes with their properties `rdf:type`, `rdfs:label`, and `rdfs:comment` as well as datatype and object properties remain in the ontology. In addition, we bind all prefixes and namespaces to the reduced ontology. Finally, all potential four consecutive empty space characters are removed from the reduced ontology.
- **Similarity-based:** It reduces the ontology to those elements (and a neighborhood) which have a similarity above a specified threshold with the column names from the data source. For this, ontology and data set embeddings are needed. We embed both ontology and data set with the transformers model *distilbert-base-uncased* [16]. *Distilbert* is a light, performant version compared to the *BERT* model and a common approach. We apply cosine similarity and bind namespaces and prefixes after identifying the most similar elements.
- **LLM-based:** It receives input data with column headers and first *200* data value rows as well as the ontology as prompt enhancement and returns a reduced ontology. We additionally extract the ontology from surrounding natural language response, if needed, see 4.7, and evaluate the syntax, see Section 4.

Next, we explain the implementation of the mapping generation module.

Mapping Generation. The embedding of the vectorized few shot examples is implemented with the model *sentence-transformers/paraphrase-multilingual-mpnet-base-v2* from an instance of *HuggingFaceEmbeddings*. For the vector store we use the *FAISS* library for easy semantic similarity search. To select the best shot example, we use an instance of the *SemanticSimilarityExampleSelector*. Prior to the evaluation, the mapping module extracts the mapping from any surrounding natural language, similar to the step in *LLM-based ontology reduction*. *MYAM+R*, hence, only evaluates the generated mapping. Next, *MYAM+R* checks the LLM response against *YARRRML* syntax by storing the file. If this returns an error, the syntax is considered invalid. If the file is successfully stored, it is considered valid. If valid, *MYAM+R* evaluates the mapping quality with the *YARRRML* evaluation module which compares the generated triples with the reference ones in relaxed mode. In addition, we conducted a qualitative *MYAM+R* expert evaluation. The results and evaluation steps are presented in the next section.

[7] `https://github.com/boschresearch/myamr_tgdk`
[8] See e.g., Workshop Ontology Matching (`https://om.ontologymatching.org/2024/`) as part of ISWC 2024: `https://iswc2024.semanticweb.org/`

```
prefixes:
    ont: http://example.com/PlantOntology#

mappings:
    plant:
        sources:
            -  [plants.csv~csv]
        s: ont:Plant/$(PlantID)
        po:
            -  [a, ont:Plant]
            -  [ont:plantID, $(PlantID)]
            -  [ont:plantName, $(PlantName)]
```

```
prefixes:
  ont: http://example.com/PlantOntology#

mappings:
  plant:
    sources:
      - [examples/data_sources/csv/generic/data/
      ↪  plant_line_machine.csv]
    s: ont:plant-$(PlantID)
    po:
      - [a, ont:Plant]
      - [ont:plantID, $(PlantID)]
      - [ont:plantName, $(PlantName)]
```

Figure 8 Comparison of snippets of a generated mapping (left) and gold standard mapping (right) of example class *plant* in *YARRRML*. A difference in *IRI* can be seen.

KG Generation. The original data set and generated mapping are input to generate an RDF KG in *turtle* format. For this, each triple is populated with the corresponding data via IDs in the triple with column names, e.g., a mapping triple subject with ID *$(PlantName)* in the *IRI* is populated with *Plant_A*, *Plant_B*, *Plant_C*, *Plant_D*, and *Plant_E* from the data in Figure 7.

4 Evaluation

In this section, we show our experimental results and evaluate the outcome, both quantitatively and qualitatively. With multiple setting configurations, we address the challenges described in Section 2 to achieve mapping generations with high quality and reduced prompt size. We evaluate the generated mappings against gold standards with *F1*-score, *precision* and *recall*. Lastly, we collect feedback on our approach and outcome. We first describe the gold standard mapping.

4.1 Gold Standard Mappings

Our *YARRRML* reference mappings are gold standard mappings and created manually by domain and knowledge experts. They are applied in practice for mapping generation. For *RML* evaluation, we manually map these *YARRRML* mappings to *RML* while preserving the mapping logic. The reference mappings represent the standard way in which an RDF KG is generated, if data source, ontology, and mappings are available. The reference mappings contain only the triples which are relevant for mapping the respective input data.

4.2 Metrics

We are evaluating experiments with different configurational settings for mapping generation. We assess the quality of each generated mapping, specifically the triples, w.r.t. the reference mapping. The reference triples are correctly generated by experts. All triples generated with *MYAM+R* are called *positives*, all other triples *negatives*. A correct triple generation is labeled with *true*, an incorrect one with *false*. The first interesting value are the *true positives*, which is the number of correct triple generations. However, errors may occur. A *false positive* is a generated triple which is not in the reference set. A *false negative* triple is not generated by the model, although it is in the reference set. The remaining *true negatives* are irrelevant and not analyzed. To evaluate each experiment, we employ F1-score (*F1*), precision (P) and recall (R). P represents the number of true positives in relation to overall positives by the model. R represents the number of true positives in relation to all positives. Finally, *F1* is the harmonic mean of P and R and formally defined as: $F1 = \frac{2}{\frac{1}{P}+\frac{1}{R}}$.

```
@prefix rr: <http://www.w3.org/ns/r2rml#> .
@prefix rml: <http://semweb.mmlab.be/ns/rml#> .
@prefix ql: <http://semweb.mmlab.be/ns/ql#> .
@prefix rdfs:
↪  <http://www.w3.org/2000/01/rdf-schema#> .
@prefix ont: <http://example.com/PlantOntology#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<#PlantMapping> a rr:TriplesMap ;
    rml:logicalSource [
        rml:source "data.csv" ;
        rml:referenceFormulation ql:CSV ] ;

    rr:subjectMap [
        rr:template "http://example.com/
        ↪  PlantOntology#Plant_{PlantID}" ;
        rr:class ont:Plant ] ;

    rr:predicateObjectMap [
        rr:predicate ont:plantID ;
        rr:objectMap [
            rml:reference "PlantID" ;
            rr:datatype xsd:integer ] ] ;

    rr:predicateObjectMap [
        rr:predicate ont:plantName ;
        rr:objectMap [
            rml:reference "PlantName" ;
            rr:datatype xsd:string ] ] .
```

```
@prefix rr: <http://www.w3.org/ns/r2rml#> .
@prefix rml: <http://semweb.mmlab.be/ns/rml#> .
@prefix ont: <http://example.com/PlantOntology#> .

<#PlantMapping>
    a rr:TriplesMap ;
    rml:logicalSource [
        rml:source
        ↪  "examples/data_sources/csv/data.csv" ;
        rml:referenceFormulation ql:CSV
    ] ;
    rr:subjectMap [
        rr:template
        ↪  "http://example.com/PlantOntology/
        ↪  {PlantID}" ;
        rr:class ont:Plant
    ] ;
    rr:predicateObjectMap [
        rr:predicate ont:plantId ;
        rr:objectMap [ rml:reference "PlantID" ]
    ] ;
    rr:predicateObjectMap [
        rr:predicate ont:plantName ;
        rr:objectMap [ rml:reference "PlantName" ]
    ] .
```

Figure 9 Comparison of snippets of a generated mapping (left) and gold standard mapping (right) of example class *plant* in *RML*. Differences in prefix declaration, source and subject *IRI* can be seen.

4.3 Mapping Generation Evaluation

Next, we examine how far the challenges described in Section 2 are addressed and, whether specific *MYAM+R* configurations can give an efficient support for experts during mapping generation. As prework for our evaluation criteria, we discuss known quality criteria for RML mappings [11] and *YARRRML* mappings [14]. Moreau *et al.* [14] update in their work dimensions and metrics for RML mapping evaluations while leaning on proposed dimensions and metrics from Zaveri *et al.* [26] on quality assessments for linked data. Hofer *et al.* [11] evaluate LLM-generated RML-mappings and the resulting KGs. The focus of their evaluations is on "fitness for use" [11], e.g., with relaxed triple matches, which compare predicate and object of a triple, but only the contained *ID* and type of a subject instead of the exact *IRI*. Similar to Hofer *et al.*, the majority of Moreau *et al.*'s dimensions and metrics is not directly applicable to our requirements as (1) LLM-supported mapping evaluation differs to the assessment of manually created mappings, (2) gold standard mapping evaluation differs to measuring mapping quality in general, and (3) our objective of expert support differs from fully automated generations. We focus on criteria to support experts best ("fitness of use" [11]) instead of a comprehensive mapping quality assessment. First, we validate the syntax of any ontology or mapping generated by an LLM. Additionally, we conduct a comparison of *(i)* the subontology against a reference ontology (manually created based on input ontology and reference mapping) and *(ii)* of generated with reference triples. Hofer *et al.* realize the generated triple evaluation on several layers with exact and relaxed scores on triples, subject, predicate, objects, properties and more. We focus on the relaxed match of generated triples as our sources provide no constraints on *IRI* construction and the data has complex assumptions on building *IRI*s. Instead of evaluating a variety of mapping aspects, we focus on those which give insights into the choice of configurational settings for *MYAM+R* to support experts on mapping generation. We evaluate our approach on the following criteria: $\boldsymbol{C_1}$**:** Valid reduced ontology syntax *(only necessary for LLM-based reduction)*, $\boldsymbol{C_2}$**:** Match of prefixes, classes, datatype properties, and object properties of generated subontology against reference ontology, $\boldsymbol{C_3}$**:** Valid generated

Table 2 Mapping Evaluation on All Configurations and Data Sets for *YARRRML*. Evaluation (*precision (P)*, *recall (R)*, *F1-Score (F1)*) of relaxed match of generated mapping triples. *Key: Config. – Configuration, D – Data Size, O – Ontology Reduction, E – Context Enhancement, dnf – did not finish.*

Configuration (C)				MaPro			CMDB			anERP			Mean		
ID	DS	OR	CE	P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	columns	naive	zero	0.80	0.53	0.64	0.35	0.47	0.40	0.75	**0.90**	0.82	0.63	0.63	0.62
2		naive	template	0.86	0.80	0.83	0.38	0.40	0.39	0.67	0.80	0.73	0.63	0.67	0.65
3		naive	few	dnf	dnf	dnf	0.53	0.67	0.59	0.75	0.30	0.43	0.64	0.48	0.55
4		similar	zero	0.58	0.73	0.65	0.50	0.67	0.57	0.75	**0.90**	0.82	0.61	0.77	0.68
5		similar	template	0.69	0.60	0.64	0.47	0.53	0.50	0.80	0.80	0.80	0.65	0.64	0.65
6		similar	few	0.80	0.80	0.80	dnf	dnf	dnf	0.64	0.70	0.67	0.72	0.75	0.73
7		LLM	zero	0.50	0.60	0.55	0.64	0.46	0.54	0.64	0.70	0.67	0.59	0.59	0.59
8		LLM	template	0.62	0.53	0.57	0.67	0.53	0.60	0.63	0.50	0.56	0.64	0.52	0.57
9		LLM	few	0.67	0.53	0.59	0.67	0.53	0.60	0.63	0.50	0.56	0.65	0.52	0.58
10	first20	naive	zero	0.75	0.80	0.77	0.40	0.53	0.46	0.67	0.80	0.73	0.61	0.71	0.65
11		naive	template	0.79	0.73	0.76	0.65	**0.73**	0.69	0.82	**0.90**	**0.86**	0.75	**0.79**	**0.77**
12		naive	few	0.63	0.80	0.71	**1.00**	0.27	0.42	0.60	0.30	0.40	0.74	0.46	0.57
13		similar	zero	0.74	**0.93**	0.82	0.40	0.53	0.46	0.67	0.80	0.73	0.60	0.76	0.67
14		similar	template	0.75	0.80	0.77	0.61	**0.73**	0.67	0.73	0.80	0.76	0.70	0.78	0.73
15		similar	few	0.63	0.80	0.71	0.55	**0.73**	0.63	0.67	0.80	0.73	0.62	0.78	0.69
16		LLM	zero	0.53	0.53	0.53	0.67	0.53	0.60	0.64	0.70	0.67	0.61	0.59	0.60
17		LLM	template	0.73	0.53	0.62	0.57	0.53	0.55	0.88	0.70	0.78	0.72	0.59	0.65
18		LLM	few	0.67	0.53	0.59	0.70	0.47	0.56	0.80	0.80	0.80	0.72	0.60	0.66
19	first200	naive	zero	0.58	0.73	0.65	0.58	**0.73**	0.65	0.58	0.70	0.64	0.58	0.72	0.64
20		naive	template	0.73	0.73	0.73	0.73	**0.73**	**0.73**	0.80	0.80	0.80	0.76	0.76	0.76
21		naive	few	**1.00**	0.20	0.33	**1.00**	0.27	0.42	0.75	0.30	0.43	0.92	0.26	0.40
22		similar	zero	0.55	0.73	0.63	0.50	0.67	0.57	dnf	dnf	dnf	0.53	0.70	0.60
23		similar	template	0.69	0.73	0.71	0.44	0.53	0.48	0.80	0.80	0.80	0.64	0.69	0.67
24		similar	few	0.81	0.87	**0.84**	**1.00**	0.27	0.42	0.75	0.30	0.43	0.85	0.48	0.61
25		LLM	zero	0.64	0.60	0.62	0.64	0.47	0.54	0.80	0.80	0.80	0.69	0.62	0.66
26		LLM	template	0.64	0.47	0.54	0.70	0.47	0.56	0.60	0.60	0.60	0.65	0.51	0.57
27		LLM	few	**1.00**	0.20	0.33	**1.00**	0.27	0.42	**0.86**	0.60	0.71	**0.95**	0.36	0.52

mapping syntax, and C_4**:** Relaxed match of generated mapping triples against gold standard (on *F1*, *P*, and *R*). We give examples of a generated mapping class, *plant*, and the respective reference mapping class for *YARRRML* in Figure 8 and for *RML* in Figure 9.

4.3.1 YARRRML Results

We refer to the experimental results in 2 with *Y-* and ID number, e.g., *Y-1* for the first experiment (data size columns-only, naive ontology reduction, zero shot context) conducted. Our mapping evaluation consists of two phases. First, we assess the syntax correctness of the generated mappings as described in Subsection 3.2. This is an important step as LLMs, specifically *GPT-models* [6] show difficulties on this task. The mapping syntax is valid in all *81* runs except for *three* experiments, *Y-3*, *Y-6*, and *Y-22*, which are marked as *dnf* in Table 2. Each invalid run originates from a duplicate key in the generated mapping. Hence, C_3 is partially fulfilled. Next, we evaluate the generated mapping triples relaxed against the reference and show our results in Table 2. Configuration *Y-27* achieves highest precision across all data sets (*P=0.95*) and also on each

Table 3 Mapping Evaluation on All Configurations and Data Sets for *RML*. Evaluation (*precision (P)*, *recall (R)*, *F1-Score (F1)*) of relaxed match of generated mapping triples. *Key: DS – Data Size, OR – Ontology Reduction, CE – Context Enhancement.*

Configuration (C)				**MaPro**			**CMDB**			**anERP**			**Mean**		
ID	DS	OR	CE	P	R	F1	P	R	F1	P	R	F1	P	R	F1
1	first20	naive	template	0.65	**0.68**	0.67	0.76	0.62	0.68	**0.75**	**0.75**	**0.75**	0.72	**0.68**	**0.70**
2		naive	few	0.72	**0.68**	0.70	0.74	0.67	0.67	0.73	0.67	0.70	0.73	0.67	**0.70**
3		similar	template	0.48	0.53	0.50	0.65	0.62	0.63	**0.75**	**0.75**	**0.75**	0.63	0.63	0.63
4		similar	few	0.61	0.58	0.59	0.67	0.67	0.67	0.73	0.67	0.70	0.67	0.64	0.65
5		LLM	template	0.60	0.47	0.53	0.71	0.48	0.57	0.73	0.67	0.70	0.68	0.54	0.60
6		LLM	few	0.60	0.47	0.53	0.86	0.57	0.69	0.70	0.58	0.64	0.72	0.54	0.62
7	first200	naive	template	0.71	0.63	0.67	0.61	0.52	0.56	0.50	0.50	0.50	0.61	0.55	0.58
8		naive	few	**0.81**	**0.68**	**0.74**	0.78	0.67	0.72	0.64	0.58	0.61	**0.74**	0.64	0.69
9		similar	template	0.50	0.53	0.51	0.57	0.57	0.57	**0.75**	**0.75**	**0.75**	0.61	0.62	0.61
10		similar	few	0.53	0.53	0.53	0.83	**0.71**	**0.77**	0.70	0.58	0.64	0.69	0.61	0.64
11		LLM	template	0.73	0.42	0.53	0.83	0.48	0.61	0.57	0.33	0.42	0.71	0.41	0.52
12		LLM	few	0.55	0.32	0.40	**0.88**	0.33	0.48	**0.75**	0.50	0.60	0.72	0.38	0.49

individually. Overall, *Y-11* shows best *recall* (*R=0.78*, with even *R=0.93* on data set *MaPro*), and overall best *F1 (0.77)*. We observe that all best *precision* values come from few shot configurations. All configurations show most difficulties on data set *CMDB*.

4.3.2 RML Results

We refer to the experimental results in 3 with *R*- and ID number, e.g., *R-1* for the first experiment (data size first20, naive ontology reduction, template context enhancement) conducted. Following our evaluation structure explained in the previous section, we first assess the syntax correctness of the generated mappings as described in Subsection 3.2. The mapping syntax is valid in all *36* runs, so, C_3 is fulfilled. Next, we evaluate the generated mapping triples relaxed against the reference and show our results in Table 3. Note that we build upon our learnings from [18] on *YARRRML* results with poor performance from datasize columns-only and context enhancement zero shot. Hence, we execute *RML* experiments on configurations with datasize first20 or first200, with context enhancement mapping template or few shot, and with all ontology reduction methods. In our experiments, we encountered in *four* out of *36* experiments rate size limits with our *gpt* model. Hence, *R-12* are conducted on only 150 instead of 200 rows for the *MaPro* dataset and *R-8*, *R-10* as well as *R-12* are conducted on only 100 instead of 200 rows. Configuration *R-8* achieves highest precision across all data sets (*P=0.74*) caused mainly by the precision value from the *MaPro* dataset. The highest precision on an individual dataset is achieved by *R-12* which is the LLM-based ontology reduction with few shot examples on the first200, but due to rate limit issues only on first 100 data rows from CMDB. Overall, *R-1* shows best *recall* (*R=0.68*, and, together with *R-2* overall best *F1 (0.70)*. With a slight deviation from the *YARRRML* results, we observe that best *precision* values may come from few shot or template configurations. We see varying result performance on an approach across datasets. E.g., *R-12* shows low performance on *MaPro* (e.g., P=0.55), yet achieves highest precision values compared to the other approaches on *CMDB* (0.88) and on *anERP* (0.75, shared with *C9*).

Table 4 Ontology Reduction Method Evaluation on All Configurations and Data Sets for *RML*. Evaluation (*F1-Score (F1)*) of classes, datatype properties, and object properties against reference ontology. *Key: OR – Ontology Reduction.*

OR		**MaPro**					**CMDB**					**anERP**					**Mean Method**		
		c-F1	d-P	d-R	d-F1	o-F1	c-F1	d-P	d-R	d-F1	o-F1	c-F1	d-P	d-R	d-F1	o-F1	c-F1	d-F1	o-F1
naive	*1*	1.0	0.83	1.0	0.91	1.0	1.0	0.87	1.0	0.93	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.95	1.0
	2	1.0	0.83	1.0	0.91	1.0	1.0	0.87	1.0	0.93	1.0	1.0	1.0	1.0	1.0	1.0			
	3	1.0	0.83	1.0	0.91	1.0	1.0	0.87	1.0	0.93	1.0	1.0	1.0	1.0	1.0	1.0			
	4	1.0	0.83	1.0	0.91	1.0	1.0	0.87	1.0	0.93	1.0	1.0	1.0	1.0	1.0	1.0			
similar	*1*	1.0	0.83	1.0	0.91	1.0	1.0	0.87	1.0	0.93	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.95	1.0
	2	1.0	0.83	1.0	0.91	1.0	1.0	0.87	1.0	0.93	1.0	1.0	1.0	1.0	1.0	1.0			
	3	1.0	0.83	1.0	0.91	1.0	1.0	0.87	1.0	0.93	1.0	1.0	1.0	1.0	1.0	1.0			
	4	1.0	0.83	1.0	0.91	1.0	1.0	0.87	1.0	0.93	1.0	1.0	1.0	1.0	1.0	1.0			
LLM	*1*	0.89	0.89	.80	0.84	0.67	1.0	0.75	0.46	0.57	0.0	0.8	1.0	0.57	0.73	0.67	0.88	0.73	0.43
	2	0.75	0.86	0.60	0.71	0.40	1.0	0.88	0.53	0.67	0.67	0.8	1.0	0.71	0.83	0.67			
	3	0.75	0.86	0.60	0.71	0.40	1.0	0.75	0.46	0.57	0.0	1.0	1.0	0.86	0.92	0.0			
	4	0.89	0.89	0.80	0.84	0.67	0.67	0.83	0.38	0.53	0.0	1.0	1.0	0.86	0.92	1.0			

4.4 Evaluation of Ontology Reduction Methods

In the following, we analyse the *three* ontology reduction methods introduced in Subsection 3.2. With the results, we address *Challenge 2* raised in Section 2. We validate the ontology syntax manually. For *YARRRML*, in all *27* experiments, the LLM-based reduced ontology is correct (C_1 fulfilled). For *RML*, in all *12* experiments, the LLM-based reduced ontology is correct (C_1 fulfilled). We show the *F1-score* of all three reduction methods on all datasets from the *RML* runs in Table 4. Note that the mapping language has no influence on the ontology reduction as it is not part of the input for this step. You may not map the IDs to the previous *RML* IDs as we merely stress the fact of four runs per ontology method at this point. All reduction methods perform well when retrieving relevant classes. Only in exceptions, specifically in more than half of the LLM-based runs, not all relevant classes were retrieved, hence, a lower *recall* led to a slight drop in *F1*-score. We analyse the datatype properties in more detail as datasets and approaches show different results. *Naive* and *similar* reduction perform perfect on *anERP*. This is also the reason for best performance of these two approaches overall (*F1* of classes: 1.0, *F1* of datatype properties: 0.95, *F1* of object properties: 1.0). The LLM-based ontology reduction methods shows difficulties in comparison to these two methods, especially with an overall *F1*-score of only 0.43 for object properties. Hence, C_2 is partially fulfilled.

4.5 Evaluation of Knowledge Graph Generation

Next, we present the KG generation results for *YARRRML* and *RML*. For *YARRRML*, we show the best performing configurations (with $F1 \geq 0.60$) on mapping generation in Table 5. The generated KGs contain tens of thousands of triples with different subjects which underlines the necessity of mappings and supported mapping generation. Configuration *Y-2* generates the most triples across all data sets and especially outperforms for data set *anERP*. The most triples on *MaPro* are generated by *Y-19*, for *CMDB* by *Y-1*, despite their average performances on mapping generation on these data sets. The reason for this lies in a high match on the subject *IRI*s which are not represented in the relaxed triple score. In *nine* cases, a configuration did not generate a KG because either the mapping was not generated in the previous step or due to an error such

Table 5 Generated KG Triples from *YARRRML* on Configurations with *mean F1* ≥ *0.60* in mapping generation. Total triples (TT), unique subjects (S), unique predicates (P), classes (C), datatype properties (D), object properties (O). Missing configurations are marked as (–) (not generated).

ID	MaPro						CMDB						anERP						Mean
	TT	S	P	C	O	D	TT	S	P	C	O	D	TT	S	P	C	O	D	TT
1	11,399	2,708	11	**5**	1	10	**66,544**	9,704	16	**4**	1	15	79,963	11,869	10	**3**	1	9	52,635
2	13,099	2,708	15	**5**	**5**	10	34,883	1,994	15	3	2	13	**162,232**	**11,951**	**17**	**3**	**3**	**14**	**70,071**
4	15,918	2,708	15	**5**	**5**	10	63,865	9,704	**17**	**4**	1	**16**	68,338	11,869	10	**3**	1	9	49,374
5	12,718	2,304	12	4	3	9	59,323	**9,824**	16	**4**	**3**	13	69,017	**11,951**	10	**3**	**3**	7	47,019
6	11,401	2,766	11	**5**	1	10	–	–	–	–	–	–	–	–	–	–	–	–	11,401
10	10,884	2,708	12	**5**	1	**11**	–	–	–	–	–	–	80,822	**11,951**	11	**3**	**3**	8	45,853
11	19,594	4,030	14	**5**	**5**	9	63,912	9,704	16	**4**	**3**	13	59,543	**11,951**	9	**3**	2	7	47,683
13	15,914	2,708	15	**5**	4	**11**	–	–	–	–	–	–	59,572	**11,951**	9	**3**	2	7	37,743
14	15,284	2,304	**16**	**5**	**5**	**11**	65,754	9,704	**17**	**4**	**3**	14	80,822	**11,951**	10	**3**	**3**	7	53,953
15	15,862	2,748	15	**5**	**5**	10	63,860	9,704	**17**	**4**	1	**16**	69,017	**11,951**	10	**3**	1	9	49,580
16	–	–	–	–	–	–	39,178	9,704	9	**4**	1	8	56,766	11,869	9	**3**	**3**	6	47,972
17	6,705	1,763	10	4	3	7	53,253	9,704	13	**4**	**3**	10	47,773	**11,951**	8	**3**	**3**	5	35,910
18	8,289	2,304	9	4	1	8	35,659	7,867	8	3	2	6	79,668	11,869	9	**3**	**3**	6	41,205
19	**21,895**	**4,518**	15	**5**	**5**	10	62,169	9,704	16	**4**	1	15	60,608	**11,951**	10	**3**	**3**	7	48,224
20	–	–	–	–	–	–	60,388	9,704	14	**4**	**3**	11	69,017	**11,951**	10	**3**	**3**	7	64,703
22	15,284	2,304	**16**	**5**	**5**	**11**	–	–	–	–	–	–	–	–	–	–	–	–	15,284
23	15,261	2,285	16	**5**	**5**	**11**	62,532	9,704	**17**	**4**	**3**	14	69,017	**11,951**	10	**3**	**3**	7	48,937
24	11,452	2,748	12	**5**	1	**11**	27,651	7,710	4	1	1	3	–	–	–	–	–	–	19,552
25	10,508	2,392	10	**5**	4	6	18,033	4,644	9	3	1	8	37,329	**11,951**	8	**3**	2	6	21,957

as non-retrievable *ID*s from the mapping. For *RML*, we show all configurations from the *RML* mapping generation in Table 6. The generated KGs contain tens of thousands of triples with different subjects which underlines the necessity of mappings and supported mapping generation. Configuration *R-4* (which corresponds to *R-4 in Table 6*) generates the most triples across all data sets. Continuing our running example, we finally show a snippet of generated RDF triples in Figure 10 with our generated RML mapping on the example data and ontology from the previous figures.

4.6 Qualitative Evaluation

As a final step, for the *YARRRML* mapping, we collect feedback from *two* domain and *two* knowledge experts on *MYAM+R* regarding effectiveness, overall satisfaction, and open feedback. The feedback on the first *two* categories is shown in Figure 11. We see that the experts on average agree to strongly agree that the generated mappings and the generated KG support their work. They also agree that they are satisfied with the approach overall and are neutral to agreeing to recommend *MYAM+R* to their colleagues. One main reason for a hesitance in recommendation to colleagues is a missing user-friendly frontend, which is, in addition to improving the provided context and *MYAM+R* documentation noted as areas of improvement by the experts. We will address this in future work. As strengths of *MYAM+R* the experts see handling large, complex ontologies and source tables with many columns. They further highlight the time savings and data quality improvement with *MYAM+R*.

Table 6 Generated KG Triples from all *RML* mapping generations. Total triples (TT) and unique subjects (S).

ID	MaPro		CMDB		anERP		Mean	
	TT	S	TT	S	TT	S	TT	S
1	**17,739**	**4,396**	38,472	6,480	**68,338**	11,869	41,516	7,582
2	11,450	2,410	60,474	**9,704**	68,043	11,869	46,656	7,994
3	13,201	2,474	62,098	**9,704**	**68,338**	11,869	47,879	**8,016**
4	11,608	2,410	**63,998**	**9,704**	68,043	11,869	**47,883**	7,994
5	9,084	2,020	48,858	**9,704**	56,713	11,869	38,218	7,864
6	9,079	2,018	43,691	8,026	68,043	11,869	40,271	7,304
7	11,893	2,472	35,221	3,613	63,518	11,869	45,204	8,015
8	8,135	2,410	54,228	**9,704**	68,043	11,869	41,293	7,435
9	11,339	2,107	35,221	3,613	**68,338**	11,869	38,299	5,863
10	8,455	1,973	54,228	**9,704**	67,800	11,869	43,494	7,849
11	9,784	2,381	25,424	6,481	56,173	11,868	30,460	6,910
12	9,064	1,970	35,192	7,711	57,036	**11,950**	33,764	7,210

4.7 Discussion and Lessons Learned

As data preparation is an issue, we chose *csv* to have a harmonization of the data set appearances. Domain-specific vocabulary and abbreviations in data such as *P_No*, which represents the project number of a machine, can be difficult for LLMs. Further, in some ontology elements we encounter unfinished comments and typos, e.g., `rdfs:comment "Describes the Plant entitiy [...]"`, which pose challenges in understanding. To directly access the relational data sources, *ontology-based data access (OBDA)* is a well-known paradigm as it encompasses "a semantic paradigm for providing a convenient and user-friendly access to data repositories" [25] including relational data bases. Our work supports the *OBDA* approach by having the ontology and source data to support users by creating mappings more rapidly and highlighting important points in the data.

In test runs, we observe multiple times a "chattiness" in LLM responses, i.e., a wrapping in natural language around the ontology or mapping, e.g., "Certainly, here is your mapping: ". This happens despite explicit instructions to *(i) not respond in natural language, but only return the mapping* and to *(ii) not include any explanations in your response*. Hence, we implement a processing step to remove any pattern surrounding the desired artefact. While the ontologies listed in our use case are rather small, in general, they are much larger in practice since they reuse domain and upper ontologies. We address this in future work.

When executing the experiments, we learn that with configurations with a high mean *F1*, we are able to generate KGs which can answer, e.g., *How many machines of a specific type are located in a particular plant? (CQ1)*. In Section 2, we list the main competency questions, yet in reality, there are many more. With our best performing mapping configurations and resulting KGs, these questions can be answered. While the *YARRRML* mapping configurations with the best *precision* utilize few shot prompts, we encounter *two YARRRML* experiments with few shots, specifically with columns only and naive reduction, that do not finish in generating a mapping. Further, *one* zero shot configuration on *200* data lines and similarity reduction leads to a non finishing run, which motivates us to conduct more runs in future work for further insights. Best precision values for *RML* mappings originate from template or few shot enhancements (we do not evaluate zero shot for *RML*). All *RML* experiments finish in generating a mapping and also, generating triples.

```
@prefix ns1: <http://example.com/PlantOntology#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

ns1:Line_101 a ns1:Line ;
    ns1:hasLine ns1:Plant_1 ;
    ns1:lineId 101 ;
    ns1:lineName "Line_X1" .

ns1:Line_102 a ns1:Line ;
    ns1:hasLine ns1:Plant_1 ;
    ns1:lineId 102 ;
    ns1:lineName "Line_X2" .

ns1:Plant_1 a ns1:Plant ;
    ns1:hasLine ns1:Plant_2 ;
    ns1:plantID 1 ;
    ns1:plantName "Plant_A" .
```

Figure 10 Generic example snippet of generated triples from a generated RDF mapping, generic input data and ontology with *MYAM*.

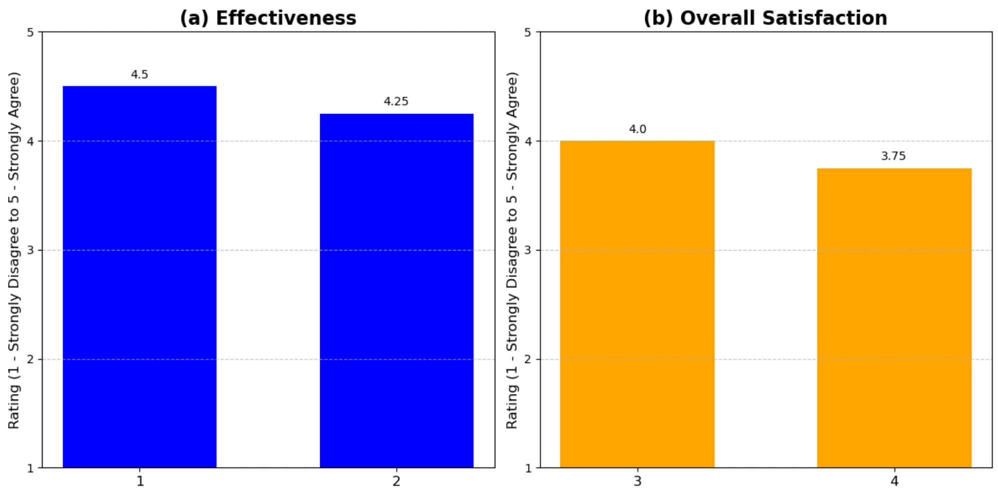


Figure 11 Expert Feedback on *MYAM+R*. Average answer on *(a)* effectiveness of and *(b)* satisfaction with *MYAM+R. Key: 1 – The mappings..., 2 – The KGs generated with MYAM+R provide support for experts on manufacturing mapping generation, 3 – Overall, I am satisfied with the MYAM+R approach, 4 – I would recommend MYAM+R to other experts for the task of mapping and KG generation.*

When analyzing the exact triples match, the results drop significantly for both mapping languages. The reason lies in the challenges of *IRI* generations that match the reference mapping. The column *ID* is in the mapping, yet the exact string often does not match the one in the reference mapping. This can be corrected by a knowledge expert with low effort, so that *MYAM+R* overall shows good support for mapping generation tasks. Nevertheless, improving the quality of LLM responses and handling the prompt size are still open challenges.

Clear limitations of our work are, first, the skipped step of accessing the data directly from the relational DB instead of a manually reviewed *.csv* file. The research on this task is active (see e.g., *TEXT2SPARQL*[9]) and it is interesting to incorporate state-of-the-art approaches in the future for

[9] `https://text2sparql.aksw.org/` co-located with Text2KG at ESWC 2025.

this step. Second, an evaluation on different LLMs is missing as well as an in-depth cost analysis which impacts a potential adoption of our approach in practice. Specifically, considering potential future regulatory updates for industry, we will expand our work in next steps with evaluations on other LLMs including open weight models. Further, we evaluate only on our proprietary data set as a public in-use benchmark set for this problem is missing. We publish a generic running example for reproducibility, however, the gap to our real industrial data remains.

Last but not least, one core motivation is the reduction of expert efforts during KG construction. Assessing these efforts in *MYAM+R*, we identify via the expert feedback an effort reduction compared to the process implemented currently in practice in our example plant. Instead of starting mapping generation from scratch, experts can in practice start from a *MYAM+R*-based mapping proposal, review the mapping and update it accordingly. However, we do not believe that experts' guidance or reviews will become completely obsolete in a target state due to the dynamic complexity of industrial data landscapes.

5 Related Work

In this section, we discuss relevant work on LLM-supported mapping generation in manufacturing. First, we review works on automatic mapping generation for KG creation and their applicability to the manufacturing domain, as well as briefly consider schema-supported mapping generation. Second, we discuss the current state of research on LLM-based KG construction. Our approach of using data input, ontology and instructions is comparable to a Graph RAG architecture as the steps of enriching an LLM prompt with schema information to generate an output, in our case a mapping, are comparable to the components of a Graph RAG. However, there is one major distinction: as our goal is KG construction, we must not assume existing supporting graph instance data in our framework.

The research field of automatic KG construction has been popular for a long time with new recent approaches. Zhong et al. [28] state in their corresponding survey (2024) that "[l]arge pre-trained models have been leading a significant impact on multiple KG construction tasks and their related applications". Our focus on *Mapping Generation* corresponds to the step of *Knowledge Acquisition* which mainly covers *Entity Discovery* and *Relation Extraction* for (semi-)structured data sources according to [28] as they also include tables in their analysis (however, without focus on them). The authors state that rule-based extraction methods "are the general solutions for NER" [28], with *Named Entity Recognition (NER)* being the first step of identifying and extracting entities from a data source as part of *Entity Discovery*. In the manufacturing domain however, the disadvantage of this approach lies in the challenge of identifying the correct pairs of (i) source column name and (ii) corresponding entity in the ontology as these pairs may differ drastically on a lexicographical level. Hence, experts conduct this decision manually or, for overall manufacturing KG construction, other "advanced deep learning methods such as BiLSTM-CRF, BiLSTM-ALBERT, etc." [21] are utilized. While experts' time is rare and costly, deep learning methods require training to prepare a model for the task at hand. Both approaches, hence, are not optimal for the requirements of efficiency and scalability in manufacturing. Chen et al. [2] state that for domain KGs "existing KG construction methods heavily rely on human intervention to attain qualified KGs, which severely hinders the practical applicability in real-world scenarios". The authors exploit LLMs for KG construction – focussing on extraction from text – and achieve a result surpassing precision values of existing SOTA KG construction methods.

While multiple KG construction approaches acquire knowledge from unstructured data sources, *Mapping Generation* from structured sources such as relational databases differs, e.g., in the availability of a structure (schema). Utilizing this schema during knowledge extraction can prove helpful, as seen in e.g., Medeiros *et al.* [3] who introduce MIRROR, a system to generate R2RML mappings. One mapping type produces RDF triples homomorphic to the ones generated by a DM approach [3]. Another results in additional mappings containing implicit knowledge from database schemas [3]. Also, Hazber *et al.* [9] integrate the database schema R2RML mapping generation.

Next, we review first approaches on LLM-supported mapping generations. Hofer *et al.* explore LLM-supported RML mapping generation. The authors see promising results of commercial LLMs, such as *Claude3 Opus (20240229)* and *GPT4 Turbo (01-25-preview)* with zero-shot and on a subset from the Internet Movie Database. Our approach differs from these works in three aspects: (1) we not only explore *RML*, but further *YARRRML* syntax, (2) we investigate necessary ontology reduction, and (3) we evaluate the performance on real manufacturing data from a plant. There exist benchmark tasks [6] for LLM performance evaluation on KG engineering tasks, e.g., parsing and creating KGs serialized in Turtle syntax. Frey *et al.* [7] assess the evolution in Turtle skills of selected commercial LLMs on such tasks. While they show an improvement on the majority of tasks of the LLMs over their predecessors, the authors observe different responses from the models, especially on the output format, and conclude that stricter multi-shot tests are worth evaluating. They further see an open question on the evaluation on RML related tasks. We address both aspects for *RML* and *YARRRML* syntax. Thoroughly assessing the mapping quality is needed from a practical manufacturing perspective, yet research shows gaps in this area. Dimou *et al.* [4] introduce a quality assessment and a refinement workflow for RDF mappings. There is further a "framework to assess the quality of RDF mappings" [14] in *YARRRML* syntax, yet, the set of metrics is not validated against an existing ontology as in our use case. Some approaches on LLM-supported and context-enhanced mapping generations have been proposed and case-study-explored [11]. To the best of our knowledge, none of the approaches have been applied in a real manufacturing setting and none considers the *YARRRML* syntax. The previous works do not take an existing target ontology into consideration, which is a constraint in our setting. As a result, state-of-the-art solutions are insufficient to address our use case. We address these research gaps in our work.

6 Conclusion and Outlook

In this paper, we introduce *MYAM+R*, an approach for LLM-supported and context-enhanced *YARRRML* and *RML* mapping generation, as a support for manufacturing knowledge experts. *MYAM+R* comprises of *eight* modules: *four* for input, as well as one for ontology reduction, mapping generation, evaluation and KG generation, each. The input modules focus on a data source sample, a related ontology, context enhancements, and prompt instructions for mapping generation. The mapping generation module can be configured with parameters for data size, ontology reduction method, and context enhancements. This module further orchestrates the processing steps, calls the ontology reduction module, and prepares the final prompt. Next, the generated mapping is evaluated against gold standard mappings which are created manually by experts. Finally, a KG generation module creates a KG based on the data and generated mapping. We conduct experiments with different configurations of *MYAM+R* across *three* real world data sets from Bosch. Naive ontology reduction and a mapping template as enhancement show promising as they achieve with column headers and first twenty data rows as input the best performance of *F1=0.74* for *YARRRML* and *F1=0.70* for *RML* mappings across all data sets.
We achieve our objective in supporting experts in manufacturing mapping generation with the obtained results and feedback. Furthermore, our results present a baseline for future experiments in the manufacturing domain. We publish the source code[10] to support further research in this direction.

[10] `https://github.com/boschresearch/myamr_tgdk`

In future work, next to addressing the limitations discussed in Section 4, we enhance *MYAM+R* in several aspects. We will improve the implementation with ontology learning from data and matching with the given ontology. With this, we will evaluate *MYAM+R* on domain and upper ontologies. Further, it is promising to extend the input configuration with a representative sample from the data input. We will explore mapping quality improvements with multiple prompts and sequential chain approaches. With the intention to increase the level of support for mapping generation in manufacturing, we explore different research avenues, e.g., the inclusion of other LLMs as well as fine-tuned small language models for a given domain.

References

1 Bruno Charron, Yu Hirate, David Purcell, and Martin Rezk. Extracting semantic information for e-commerce. In Paul Groth, Elena Simperl, Alasdair J. G. Gray, Marta Sabou, Markus Krötzsch, Freddy Lécué, Fabian Flöck, and Yolanda Gil, editors, *ISWC*, volume 9982 of *Lecture Notes in Computer Science*, pages 273–290, 2016. doi:10.1007/978-3-319-46547-0_27.

2 Hanzhu Chen, Xu Shen, Qitan Lv, Jie Wang, Xiaoqi Ni, and Jieping Ye. Sac-kg: Exploiting large language models as skilled automatic constructors for domain knowledge graphs. *arXiv preprint arXiv:2410.02811*, 2024. doi:10.48550/arXiv.2410.02811.

3 Luciano Frontino de Medeiros, Freddy Priyatna, and Oscar Corcho. Mirror: Automatic R2RML mapping generation from relational databases. In *Engineering the Web in the Big Data Era: 15th International Conference, ICWE 2015, Rotterdam, The Netherlands, June 23-26, 2015, Proceedings 15*, pages 326–343. Springer, 2015. doi:10.1007/978-3-319-19890-3_21.

4 Anastasia Dimou, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, Sebastian Hellmann, and Rik Van de Walle. Assessing and refining mappingsto RDF to improve dataset quality. In *The Semantic Web-ISWC 2015: 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part II 14*, pages 133–149. Springer, 2015. doi:10.1007/978-3-319-25010-6_8.

5 Thomas Eiter, Tobias Geibinger, Nysret Musliu, Johannes Oetsch, Peter Skocovský, and Daria Stepanova. Answer-set programming for lexicographical makespan optimisation in parallel machine scheduling - ADDENDUM. *Theory Pract. Log. Program.*, 24(2):421, 2024. doi:10.1017/S1471068424000061.

6 Johannes Frey, Lars-Peter Meyer, Natanael Arndt, Felix Brei, and Kirill Bulert. Benchmarking the abilities of Large Language Models for RDF knowledge graph creation and comprehension: How well do LLMs speak Turtle? *arXiv preprint arXiv:2309.17122*, 2023. doi:10.48550/arXiv.2309.17122.

7 Johannes Frey, Lars-Peter Meyer, Felix Brei, Sabine Gründer-Fahrer, and Michael Martin. Assessing the evolution of LLM capabilities for knowledge graph engineering in 2023. In *ESWC*, volume 24, pages 26–30, 2024.

8 Irlán Grangel-González, Felix Lösch, and Anees ul Mehdi. Knowledge graphs for efficient integration and access of manufacturing data. In *25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, pages 93–100, 2020. doi:10.1109/ETFA46521.2020.9212156.

9 Mohamed AG Hazber, Ruixuan Li, Guandong Xu, and Khaled M Alalayah. An approach for automatically generating R2RML-based direct mapping from relational databases. In *Social Computing: Second International Conference of Young Computer Scientists, Engineers and Educators, ICYCSEE 2016, Harbin, China, August 20-22, Proceedings, Part I 2*, pages 151–169. Springer, 2016.

10 Pieter Heyvaert, Ben De Meester, Anastasia Dimou, and Ruben Verborgh. Declarative rules for linked data generation at your fingertips! In *The Semantic Web: ESWC 2018 Satellite Events: ESWC 2018 Satellite Events, Heraklion, Crete, Greece, June 3-7, 2018, Revised Selected Papers 15*, pages 213–217. Springer, 2018. doi:10.1007/978-3-319-98192-5_40.

11 Marvin Hofer, Johannes Frey, and Erhard Rahm. Towards self-configuring knowledge graph construction pipelines using llms-a case study with rml. In *Fifth International Workshop on Knowledge Graph Construction@ ESWC2024*, 2024.

12 Evgeny Kharlamov, Gulnar Mehdi, Ognjen Savkovic, Guohui Xiao, Elem Güzel Kalayci, and Mikhail Roshchin. Semantically-enhanced rule-based diagnostics for industrial internet of things: The SDRL language and case study for siemens trains and turbines. *J. Web Semant.*, 56:11–29, 2019. doi:10.1016/J.WEBSEM.2018.10.004.

13 Andrew LeClair, Alicia Marinache, Haya El Ghalayini, Wendy MacCaull, and Ridha Khedri. A review on ontology modularization techniques - a multi-dimensional perspective. *IEEE Transactions on Knowledge and Data Engineering*, 35(5):4376–4394, 2023. doi:10.1109/TKDE.2022.3152928.

14 Benjamin Moreau and Patricia Serrano-Alvarado. Assessing the quality of RDF mappings with evamap. In *The Semantic Web: ESWC 2020 Satellite Events: ESWC 2020 Satellite Events, Heraklion, Crete, Greece, May 31–June 4, 2020, Revised Selected Papers 17*, pages 164–167. Springer, 2020. doi:10.1007/978-3-030-62327-2_28.

15 Jan Portisch, Michael Hladik, and Heiko Paulheim. Background knowledge in ontology match-

ing: A survey. *Semantic Web*, 15(6):2639–2693, 2024. doi:10.3233/SW-223085.

16 Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019. arXiv:1910.01108.

17 Patrick Sapel, Lina Molinas Comet, Iraklis Dimitriadis, Christian Hopmann, and Stefan Decker. A review and classification of manufacturing ontologies. *Journal of Intelligent Manufacturing*, 2024. doi:10.1007/s10845-024-02425-z.

18 Wilma Johanna Schmidt, Irlan Grangel-González, Tobias Huschle, Lena Wagner, Evgeny Kharlamov, and Adrian Paschke. Llm-supported mapping generation for semantic manufacturing treasure hunting. In *European Semantic Web Conference*, pages 84–101. Springer, 2025.

19 Wilma Johanna Schmidt, Irlan Grangel-González, Adrian Paschke, and Evgeny Kharlamov. MYAMR_TGDK. Software, partially funded by https://graph-massivizer.eu/, partially funded by https://www.smarty-project.eu/, partially funded by https://enrichmydata.eu/ (visited on 2025-12-08). URL: https://github.com/boschresearch/myamr_tgdk, doi:10.4230/artifacts.25262.

20 Wilma Johanna Schmidt, Diego Rincon-Yanez, Evgeny Kharlamov, and Adrian Paschke. Systematic Literature Review on Neuro-Symbolic AI in Knowledge Graph Construction for Manufacturing. *under review*, 2025.

21 Yuhu Shang, Yimeng Ren, Hao Peng, Yue Wang, Gang Wang, Zhong Cheng Li, Yangzhao Yang, and Yangyang Li. A perspective survey on industrial knowledge graphs: Recent advances, open challenges, and future directions. In *2023 International Conference on Machine Learning and Cybernetics (ICMLC)*, pages 194–200. IEEE, 2023. doi:10.1109/ICMLC58545.2023.10327989.

22 Zhigang Sun, Zixu Wang, Lavdim Halilaj, and Juergen Luettin. Semanticformer: Holistic and semantic traffic scene representation for trajectory prediction using knowledge graphs. *IEEE Robotics Autom. Lett.*, 9(9):7381–7388, 2024. doi:10.1109/LRA.2024.3426386.

23 Dylan Van Assche, Thomas Delva, Gerald Haesendonck, Pieter Heyvaert, Ben De Meester, and Anastasia Dimou. Declarative RDF graph generation from heterogeneous (semi-)structured data: A systematic literature review. *Journal of Web Semantics*, 75:100753, 2023. doi:10.1016/j.websem.2022.100753.

24 Dylan Van Assche, Thomas Delva, Pieter Heyvaert, Ben De Meester, and Anastasia Dimou. Towards a more human-friendly knowledge graph generation & publication. In *ISWC2021, The International Semantic Web Conference*, volume 2980. CEUR, 2021. URL: https://ceur-ws.org/Vol-2980/paper384.pdf.

25 Guohui Xiao, Diego Calvanese, Roman Kontchakov, Domenico Lembo, Antonella Poggi, Riccardo Rosati, and Michael Zakharyaschev. Ontology-based data access: A survey. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, 2018.

26 Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, and Soeren Auer. Quality assessment for linked data: A survey. *Semantic Web*, 7(1):63–93, 2016. doi:10.3233/SW-150175.

27 Xiang Zhang, Gong Cheng, and Yuzhong Qu. Ontology summarization based on rdf sentence graph. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 707–716, New York, NY, USA, 2007. Association for Computing Machinery. doi:10.1145/1242572.1242668.

28 Lingfeng Zhong, Jia Wu, Qian Li, Hao Peng, and Xindong Wu. A comprehensive survey on automatic knowledge graph construction. *ACM Computing Surveys*, 56(4):1–62, 2023. doi:10.1145/3618295.

29 Baifan Zhou, Yulia Svetashova, Andre Gusmao, Ahmet Soylu, Gong Cheng, Ralf Mikut, Arild Waaler, and Evgeny Kharlamov. Semml: Facilitating development of ML models for condition monitoring with semantics. *J. Web Semant.*, 71:100664, 2021. doi:10.1016/J.WEBSEM.2021.100664.