

# Consistent Query Answering over SHACL Constraints (Extended Abstract)

Shqiponja Ahmetaj<sup>1</sup>, Timo Camillo Merkl<sup>1</sup> and Reinhard Pichler<sup>1</sup>

<sup>1</sup>TU Wien, Vienna, Austria

## 1. Introduction

The Shapes Constraint Language (SHACL) is the World Wide Web recommended language for expressing and validating constraints on RDF graphs [1]. SHACL uses the notion of *shapes graph* to describe a set of *shape* constraints paired with *targets*, that specify which nodes of the RDF graph should satisfy which shapes. The main computational problem in SHACL is to check whether an RDF graph *validates* a shapes graph. Indeed, large RDF triple stores, which are subject to frequent change, may be missing some facts to validate a target or may have conflicting or contradictory facts. Therefore, detecting such inconsistencies and measures to deal with data graphs that fail to validate a shapes graph is very relevant in practice.

A possible solution is to fix the data graph before reasoning. In the style of database *repairs*, [2, 3] propose to fix the data graph through (minimal) additions or removals of facts such that the resulting data graph validates the shapes graph. However, this may not always be desirable in practice as there may be a large number of possible repairs and selecting one may keep wrong facts, or remove true facts. Another alternative is to *tolerate* the non-validation and find ways to leverage the consistent part of the data and obtain meaningful and correct answers to queries despite the non-validation. This view is known as *consistent query answering* (CQA), and it has gained a lot of attention since the seminal paper [4]. The idea is to accept as answers to a query those that are true over all (minimal) repairs of the input database. This is also known as the *AR* semantics [4, 5, 6, 7]. Several other inconsistency-tolerant semantics have also been studied such as *brave* [8] and *IAR* [6] semantics. The former accepts answers that are true in *some* repair, and the latter accepts the most reliable answers, that is those that are true in the *intersection of all* repairs. There is now a large body of research works on CQA in both the database and KR setting; we refer to [5, 9, 10] for nice surveys.

In this work, we focus on CQA in the presence of (recursive) SHACL shapes. We thus consider a fundamental fragment of the standard query language SPARQL. Specifically, we focus on *basic graph patterns* (BGPs) and the well-behaved extension with the OPTIONAL operator [11], referred to as *well-designed queries* (WDQs) in this paper. Our main goal is a detailed complexity analysis of the CQA problem. In total, we study several variants of the CQA problem by considering 4 query languages (BGPs and WDQs, with or without projection), under 3 semantics (brave, AR, IAR), with or without (cardinality or subset inclusion) minimality-restrictions.

AMW 2024: 16th Alberto Mendelzon International Workshop on Foundations of Data Management, September 30th–October 4th, 2024, Mexico City, Mexico

✉ shqiponja.ahmetaj@tuwien.ac.at (S. Ahmetaj); timo.merkl@tuwien.ac.at (T. C. Merkl);  
reinhard.pichler@tuwien.ac.at (R. Pichler)



© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

Moreover, we distinguish data complexity (where the SHACL constraints and the query are considered as fixed and only the data is allowed to vary) and combined complexity.

We initially consider the scenario that the repairs must validate *all* targets. If this is not possible, we have to settle for repairs that validate a subset of the targets. To address this, in the spirit of inconsistency tolerance, [3] proposes a relaxed notion of repairs, which aims at validating a maximal subset of the targets. We also study the complexity of CQA over *maximal repairs*. In all cases considered, we provide a complete complexity classification in the form of matching upper and lower bounds. All details of our results are provided in the full paper [12, 13].

## 2. SHACL Validation and Well-Designed SPARQL

*RDF Graphs.* Let  $N_N, N_C, N_P$  denote sets of *nodes* (constants), *class names*, and *property names*, respectively. An RDF (data) graph  $G$  is a finite set of (ground) *atoms* of the form  $B(c)$  and  $p(c, d)$ , where  $B \in N_C, p \in N_P$ , and  $c, d \in N_N$ .

*SHACL Validation.* Let  $N_S$  be a set of *shape names*. A *shape atom* is an expression of the form  $s(a)$ , where  $s \in N_S$  and  $a \in N_N$ . A *path expression*  $E$  is a regular expression built using the usual operators  $*$ ,  $\cdot$ ,  $\cup$ , property names  $p \in N_P$  and *inverse properties*  $p^-$ , where  $p \in N_P$ . A (*complex*) *shape* is an expression  $\varphi$  obeying the syntax:

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

where  $s \in N_S, p \in N_P, B \in N_C, c \in N_N, n$  is a positive integer, and  $E, E'$  are path expressions. A (*shape*) *constraint* is an expression  $s \leftrightarrow \varphi$  where  $s \in N_S$  and  $\varphi$  is a complex shape. A *shapes graph* is a pair  $(\mathcal{C}, \mathcal{T})$ , where  $\mathcal{C}$  is a set of constraints and  $\mathcal{T}$  is a set of shape atoms called *targets*. A set of constraints  $\mathcal{C}$  is *recursive*, if there is a shape name in  $\mathcal{C}$  that directly or indirectly refers to itself. A (*shape*) *assignment* for a data graph  $G$  is a set  $I = G \cup L$ , where  $L$  is a set of shape atoms. The evaluation of a complex shape w.r.t. an assignment  $I$  is given in terms of a function  $\llbracket \cdot \rrbracket^I$  that maps shape expressions  $\varphi$  to a set of nodes, and path expressions  $E$  to a set of pairs of nodes with the connectives interpreted in the usual way; we refer to [13] for more details.

Following the supported model semantics proposed in [14], given a shapes graph  $(\mathcal{C}, \mathcal{T})$ , an assignment  $I$  for  $G$  is a (*supported*) *model* of  $\mathcal{C}$  if  $\llbracket \varphi \rrbracket^I = s^I$  for all  $s \leftrightarrow \varphi \in \mathcal{C}$ . The data graph  $G$  *validates*  $(\mathcal{C}, \mathcal{T})$  if there exists a supported model  $I$  of  $\mathcal{C}$  such that  $\mathcal{T} \subseteq L$ .

*Well-Designed SPARQL.* Let  $N_V$  be a set of variables. A *basic graph pattern* (BGP) is a conjunctive query (without existentially quantified variables) over atoms of the form  $B(t)$  or  $p(t_1, t_2)$  with  $B \in N_C, p \in N_P$ , and  $t, t_1, t_2 \in N_V \cup N_N$ . A *well-designed* SPARQL query  $Q$  (wdQs) is built from BGPs and the *OPTIONAL* (or OPT) operator such that for every subquery  $Q' = (P_1 \text{ OPT } P_2)$  of  $Q$ , the variables of  $P_2$  that appear outside of  $Q'$  also appear in  $P_1$ . A mapping  $\mu$  is any partial function whose domain  $\text{dom}(\mu)$  is from  $N_V$ . The notion of a mapping  $\mu$  that is an *answer* to a BGP  $Q$  over a graph  $G$  is defined in the standard way. Mappings  $\mu_1$  and  $\mu_2$  are *compatible* (written  $\mu_1 \sim \mu_2$ ) if  $\mu_1(x) = \mu_2(x)$  for all  $x \in \text{dom}(\mu_1) \cap \text{dom}(\mu_2)$ . A mapping  $\mu$  is an answer to a wdQ  $Q = Q_1 \text{ OPT } Q_2$  if (1)  $\mu$  is of the form  $\mu_1 \cup \mu_2$ , where  $\mu_1$  is an answer to  $Q_1$ ,  $\mu_2$  is an answer to  $Q_2$ , and  $\mu_1 \sim \mu_2$ , or (2)  $\mu$  is an answer to  $Q_1$  for which there does not exist an

answer to  $Q_2$  that is compatible with  $\mu$ . We denote the query classes where we additionally allow top-level projections by  $\pi$ -BGP and  $\pi$ -WDQ.

### 3. Inconsistency-tolerant Semantics and Complexity Results

In this section, we formally introduce the problems considered in the paper and present our main results. As was done in [2], we can explain non-validation of a SHACL shapes graph in the style of database repairs. Hence, a repair is provided as a set  $A$  of facts to be added and a set  $D$  of facts to be deleted, so that the resulting data graph validates the shapes graph. Concretely, let  $G$  be a data graph,  $(\mathcal{C}, \mathcal{T})$  a SHACL shapes graph, and let  $H$  be another data graph disjoint from  $G$ , called *hypotheses*. Then, a *repair* is a pair  $R = (A, D)$ , such that  $D \subseteq G$ ,  $A \subseteq H$ , and the *repaired graph*  $G_R := (G \setminus D) \cup A$  validates  $(\mathcal{C}, \mathcal{T})$ . Note the  $H$  is necessary, as allowing arbitrary atoms to be added makes the problem of checking the existence of a repair undecidable.

Instead of considering all possible repairs, we consider *preference relations* given by a pre-order  $\preceq$  (a reflexive and transitive relation) on the set of repairs. Following [2], we study *subset-minimal* ( $\subseteq$ ), and *cardinality-minimal* ( $\leq$ ) repairs. We denote with  $=$  when there is no preference order, and we use  $\preceq$  as a placeholder for  $\subseteq$ ,  $\leq$ , and  $=$ . We say a repaired graph  $G_R$  is  $\preceq$ -minimal iff  $R$  is  $\preceq$ -minimal.

We define the three inconsistency-tolerant semantics *brave* ( $\exists$ ), *AR* ( $\forall$ ), and *IAR semantics* ( $\cap$ ). Consider a query  $Q$ , a mapping  $\mu$ , a data graph  $G$ , a shapes graph  $(\mathcal{C}, \mathcal{T})$ , and hypotheses  $H$ . Then,  $\mu$  is an answer of  $Q$  over  $\Psi = (G, \mathcal{C}, \mathcal{T}, H)$  and preference order  $\preceq \in \{=, \leq, \subseteq\}$  under:

- *brave semantics*, if  $\mu$  is an answer to  $Q$  over *some*  $\preceq$ -minimal repaired graph  $G_R$ ,
- *AR semantics*, if  $\mu$  is an answer to  $Q$  over *all*  $\preceq$ -minimal repaired graphs  $G_R$ ,
- *IAR semantics*, if  $\mu$  is an answer to  $Q$  over  $G_\cap := \bigcap \{G_R \mid R \text{ is a } \preceq\text{-minimal repair}\}$ .

We illustrate SHACL and the various inconsistency-tolerant semantics with an example.

**Example.** Consider data graph  $G$ , hypotheses  $H$ , and the shapes graph  $(\mathcal{C}, \mathcal{T})$ :

$$\begin{aligned} G &= \{ \text{Stud}(\text{Ann}), \text{Stud}(\text{Ben}), \text{id}(\text{Ann}, 3), \text{id}(\text{Ben}, 1), \text{id}(\text{Ben}, 2), \text{enrolledIn}(\text{Ben}, a) \}, \\ H &= \{ \text{enrolledIn}(\text{Ann}, b), \text{enrolledIn}(\text{Ben}, c) \}, \\ \mathcal{C} &= \{ \text{ActiveStud} \leftrightarrow \text{Stud} \wedge =_1 \text{id} \wedge \geq_1 \text{enrolledIn} \}, \\ \mathcal{T} &= \{ \text{ActiveStud}(\text{Ann}), \text{ActiveStud}(\text{Ben}) \}. \end{aligned}$$

*Intuitively, the constraint states that “active students” are students that have exactly one ID and are enrolled in at least one course. The targets ask to check whether Ann and Ben are active students. The data graph  $G$  does not validate the shapes graph. Intuitively, the reason is that Ben has two IDs and Ann is not enrolled in any course. Validation can be obtained by repairing  $G$  with the subset- and cardinality-minimal repairs  $R_1 = (A, D_1)$  and  $R_2 = (A, D_2)$ , where  $A = \{ \text{enrolledIn}(\text{Ann}, b) \}$ , and each  $D_j$  includes the fact  $\text{id}(\text{Ben}, j)$ . There are 4 more non-minimal repairs, i.e., repairs where additionally  $\text{enrolledIn}(\text{Ben}, c)$  is added and, optionally, the atom  $\text{enrolledIn}(\text{Ben}, a)$  is removed. Next, consider the BGP  $Q_1 = \text{Stud}(x) \wedge \text{id}(x, y)$  and WDQ  $Q_2 = \text{Stud}(x) \text{ OPT } \text{id}(x, y)$ . Clearly,*

$\mathcal{L} \setminus \preceq, \mathcal{S}$	$=, \exists$	$\leq, \exists$	$\subseteq, \exists$	$=, \forall$	$\leq, \forall$	$\subseteq, \forall$	$=, \cap$	$\leq, \cap$	$\subseteq, \cap$
BGP (DC)	NP	$\Theta_2P$	$\Sigma_2P$	coNP	$\Theta_2P$	$\Pi_2P$	coNP	$\Theta_2P$	$\Pi_2P$
$\pi$ -BGP (DC)	NP	$\Theta_2P$	$\Sigma_2P$	coNP	$\Theta_2P$	$\Pi_2P$	coNP	$\Theta_2P$	$\Pi_2P$
WDQ (DC)	NP	$\Theta_2P$	$\Sigma_2P$	coNP	$\Theta_2P$	$\Pi_2P$	DP	$\Theta_2P$	$DP_2$
$\pi$ -WDQ (DC)	NP	$\Theta_2P$	$\Sigma_2P$	coNP	$\Theta_2P$	$\Pi_2P$	$\Theta_2P$	$\Theta_2P$	$\Theta_3P$
BGP (CC)	NP	$\Theta_2P$	$\Sigma_2P$	coNP	$\Theta_2P$	$\Pi_2P$	coNP	$\Theta_2P$	$\Pi_2P$
$\pi$ -BGP (CC)	NP	$\Theta_2P$	$\Sigma_2P$	$\Pi_2P$	$\Pi_2P$	$\Pi_2P$	$\Theta_2P$	$\Theta_2P$	$\Pi_2P$
WDQ (CC)	$\Sigma_2P$	$\Sigma_2P$	$\Sigma_2P$	coNP	$\Theta_2P$	$\Pi_2P$	$\Theta_2P$	$\Theta_2P$	$DP_2$
$\pi$ -WDQ (CC)	$\Sigma_2P$	$\Sigma_2P$	$\Sigma_2P$	$\Pi_3P$	$\Pi_3P$	$\Pi_3P$	$\Sigma_2P$	$\Sigma_2P$	$\Theta_3P$

**Table 1**

Data (DC) and combined complexity (CC) of the problem  $CQA(\mathcal{L}, \preceq, \mathcal{S})$ .

the mapping  $\mu_1 = \{x \rightarrow \text{Ann}, y \rightarrow 3\}$  is an answer of  $Q_1$  under brave, AR, and IAR semantics. The mappings  $\mu_2 = \{x \rightarrow \text{Ben}, y \rightarrow 1\}$  and  $\mu_3 = \{x \rightarrow \text{Ben}, y \rightarrow 2\}$  are answers of  $Q_1$  over  $G_{R_1}$  and  $G_{R_2}$ , respectively, and hence under brave semantics, but not under AR nor IAR semantics. For  $Q_2$ ,  $\mu_1$  and  $\mu_4 = \{x \rightarrow \text{Ben}\}$  are the solutions under IAR semantics. Clearly,  $\mu_1$ ,  $\mu_2$ , and  $\mu_3$  are still the answers under brave semantics and  $\mu_1$  under AR semantics. Note that the above statements hold for each preference order  $\preceq \in \{\subseteq, \leq, =\}$ .

For query language  $\mathcal{L} \in \{\text{BGP}, \pi\text{-BGP}, \text{WDQ}, \pi\text{-WDQ}\}$ , preference order  $\preceq \in \{=, \leq, \subseteq\}$ , and inconsistency-tolerant semantics  $\mathcal{S} \in \{\exists, \forall, \cap\}$ , we define the  $CQA(\mathcal{L}, \preceq, \mathcal{S})$  problem as follows:

Input: A query  $Q \in \mathcal{L}$ ,  $\Psi = (G, \mathcal{C}, \mathcal{T}, H)$ , and a mapping  $\mu$ .

Question: Is  $\mu$  an answer of  $Q$  over  $\Psi$  and preference order  $\preceq$  under  $\mathcal{S}$ -semantics?

The complete picture of our complexity results in all settings is shown in Table 1.

We also consider the setting where there is no repair of the data graph that validates all the targets. E.g., consider constraints  $s1 \leftrightarrow B$  and  $s2 \leftrightarrow \neg B$  and targets  $s1(a)$  and  $s2(a)$ ; in this case, there exists no repair for any input data graph, since adding  $B(a)$  violates the second constraint and not adding it violates the first one. Following [3], we relax the notion of repairs and aim at satisfying the maximum number of targets. Such tuples  $R = (A, D)$  that maximize the number of targets validated by  $G_R$  are called *maximal repairs* and we define the problem  $MCQA(\mathcal{L}, \preceq, \mathcal{S})$  analogously to  $CQA(\mathcal{L}, \preceq, \mathcal{S})$  where maximal repairs play the role of repairs. It turns out that the complexities remain almost as in Table 1 – only the classes NP, coNP, DP have to be replaced by the boolean hierarchy BH in data complexity and by  $\Theta_2P$  in combined complexity.

## 4. Conclusion and Future Work

In this work, we have carried out a thorough complexity analysis of the CQA problem for data graphs with SHACL constraints and we have pinpointed the complexity of this problem in a multitude of settings – considering various query languages, inconsistency-tolerant semantics of CQA, and preference relations on repairs. Additionally, we have studied the CQA problem for maximal repairs in cases where no repair exists. Several new proof techniques had to be

developed to obtain these results. For instance, this has allowed us – in contrast to [2] – to prove all our hardness results without making use of recursion in shapes constraints.

The targets we considered here are shape atoms of the form  $s(c)$ . The SHACL standard also allows for richer targets over class and property names. Specifically, it allows to state that a data graph must validate a shape name at each node of a certain class name, or domain (or range) of a property name. All the membership results in this paper can be immediately updated to support these richer targets. Some features of SHACL (such as disjointness and closed constraints) are not considered here, but we strongly believe they do not change the complexity results.

An immediate direction for future work is to investigate the notion of optimal repairs of prioritized data graphs over SHACL constraints and specifically, the notions of global, Pareto and completion optimality of repairs [15]. In particular, it would be of interest to study the computational properties of the main reasoning tasks such as repair checking and repair existence, and to analyze CQA for each of the three notions of optimal repairs and the various settings studied in this paper.

In the absence of general tractability results, a further important next step is to devise practical algorithms for CQA over SHACL constraints, either by relying on heuristics or by identifying meaningful and relevant fragments of SHACL that admit better complexity results.

## Acknowledgements

This work was supported by the Vienna Science and Technology Fund (WWTF) [10.47379/ICT2201, 10.47379/VRG18013]. In addition, Ahmetaj was supported by the FWF and netidee SCIENCE project T1349-N.

## References

- [1] H. Knublauch, D. Kontokostas, Shapes constraint language (SHACL). W3C Recommendation, W3C., 2017. <https://www.w3.org/TR/shacl/>.
- [2] S. Ahmetaj, R. David, M. Ortiz, A. Polleres, B. Shehu, M. Simkus, Reasoning about explanations for non-validation in SHACL, in: M. Bienvenu, G. Lakemeyer, E. Erdem (Eds.), Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning, KR 2021, Online event, November 3-12, 2021, 2021, pp. 12–21. URL: <https://doi.org/10.24963/kr.2021/2>. doi:10.24963/kr.2021/2.
- [3] S. Ahmetaj, R. David, A. Polleres, M. Simkus, Repairing SHACL constraint violations using answer set programming, in: U. Sattler, A. Hogan, C. M. Keet, V. Presutti, J. P. A. Almeida, H. Takeda, P. Monnin, G. Pirrò, C. d’Amato (Eds.), 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*, Springer, 2022, pp. 375–391. URL: [https://doi.org/10.1007/978-3-031-19433-7\\_22](https://doi.org/10.1007/978-3-031-19433-7_22). doi:10.1007/978-3-031-19433-7\_22.
- [4] M. Arenas, L. E. Bertossi, J. Chomicki, Consistent query answers in inconsistent databases, in: V. Vianu, C. H. Papadimitriou (Eds.), Proceedings of the Eighteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, May 31 - June 2, 1999,

- Philadelphia, Pennsylvania, USA, ACM Press, 1999, pp. 68–79. URL: <https://doi.org/10.1145/303976.303983>. doi:10.1145/303976.303983.
- [5] L. E. Bertossi, Database Repairing and Consistent Query Answering, Synthesis Lectures on Data Management, Morgan & Claypool Publishers, 2011. URL: <https://doi.org/10.2200/S00379ED1V01Y201108DTM020>. doi:10.2200/S00379ED1V01Y201108DTM020.
  - [6] D. Lembo, M. Lenzerini, R. Rosati, M. Ruzzi, D. F. Savo, Inconsistency-tolerant semantics for description logics, in: P. Hitzler, T. Lukasiewicz (Eds.), Web Reasoning and Rule Systems - Fourth International Conference, RR 2010, Bressanone/Brixen, Italy, September 22-24, 2010. Proceedings, volume 6333 of *Lecture Notes in Computer Science*, Springer, 2010, pp. 103–117. URL: [https://doi.org/10.1007/978-3-642-15918-3\\_9](https://doi.org/10.1007/978-3-642-15918-3_9). doi:10.1007/978-3-642-15918-3\_9.
  - [7] S. Arming, R. Pichler, E. Sallinger, Complexity of repair checking and consistent query answering, in: W. Martens, T. Zeume (Eds.), 19th International Conference on Database Theory, ICDT 2016, Bordeaux, France, March 15-18, 2016, volume 48 of *LIPICs*, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016, pp. 21:1–21:18. URL: <https://doi.org/10.4230/LIPICs.ICDT.2016.21>. doi:10.4230/LIPICs.ICDT.2016.21.
  - [8] M. Bienvenu, R. Rosati, Tractable approximations of consistent query answering for robust ontology-based data access, in: F. Rossi (Ed.), IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013, IJCAI/AAAI, 2013, pp. 775–781. URL: <http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6904>.
  - [9] J. Wijsen, Foundations of query answering on inconsistent databases, *SIGMOD Rec.* 48 (2019) 6–16. URL: <https://doi.org/10.1145/3377391.3377393>. doi:10.1145/3377391.3377393.
  - [10] M. Bienvenu, C. Bourgaux, Inconsistency-tolerant querying of description logic knowledge bases, in: J. Z. Pan, D. Calvanese, T. Eiter, I. Horrocks, M. Kifer, F. Lin, Y. Zhao (Eds.), Reasoning Web: Logical Foundation of Knowledge Graph Construction and Query Answering - 12th International Summer School 2016, Aberdeen, UK, September 5-9, 2016, Tutorial Lectures, volume 9885 of *Lecture Notes in Computer Science*, Springer, 2016, pp. 156–202. URL: [https://doi.org/10.1007/978-3-319-49493-7\\_5](https://doi.org/10.1007/978-3-319-49493-7_5). doi:10.1007/978-3-319-49493-7\_5.
  - [11] J. Pérez, M. Arenas, C. Gutierrez, Semantics and complexity of SPARQL, *ACM Trans. Database Syst.* 34 (2009) 16:1–16:45. URL: <https://doi.org/10.1145/1567274.1567278>. doi:10.1145/1567274.1567278.
  - [12] S. Ahmetaj, T. C. Merkl, R. Pichler, Consistent query answering over shacl constraints, Accepted to KR 2024, the 21st International Conference on Principles of Knowledge Representation and Reasoning (2024).
  - [13] S. Ahmetaj, T. C. Merkl, R. Pichler, Consistent query answering over shacl constraints, CoRR abs/2406.16653 (2024). URL: <https://arxiv.org/abs/2406.16653>. arXiv:2406.16653.
  - [14] J. Corman, J. L. Reutter, O. Savkovic, Semantics and validation of recursive SHACL, in: D. Vrandečić, K. Bontcheva, M. C. Suárez-Figueroa, V. Presutti, I. Celino, M. Sabou, L. Kaffee, E. Simperl (Eds.), The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part I, volume 11136 of *Lecture Notes in Computer Science*, Springer, 2018, pp. 318–336. URL: [https://doi.org/10.1007/978-3-030-00671-6\\_19](https://doi.org/10.1007/978-3-030-00671-6_19). doi:10.1007/978-3-030-00671-6\_19.



- [15] S. Staworko, J. Chomicki, J. Marcinkowski, Prioritized repairing and consistent query answering in relational databases, *Ann. Math. Artif. Intell.* 64 (2012) 209–246. URL: <https://doi.org/10.1007/s10472-012-9288-8>. doi:10.1007/S10472-012-9288-8.