Logical Difference Computation with CEX2.5

Boris Konev, Michel Ludwig, and Frank Wolter

Department of Computer Science, University of Liverpool, United Kingdom
`{konev,mludwig,wolter}@liverpool.ac.uk`

Abstract. We present a new version of the CEX versioning tool for ontologies. CEX detects logical differences between acyclic terminologies in the lightweight description logic EL with role inclusions and domain and range restrictions. Depending on the application, CEX outputs differences between terminologies that capture derived concept inclusions, answers to instance queries, and answers to conjunctive queries. Experiments with versions of the NCI ontology are conducted to evaluate the performance of CEX and compare the three types of differences.

Keywords: Description Logics, Ontology Versioning, Logical Difference

1 Introduction

In life sciences, healthcare, and other knowledge intensive areas, large scale terminologies are employed to provide a common vocabulary for a domain of interest together with descriptions of the meaning of terms built from the vocabulary and relationships between them. Two examples are the medical terminology SNOMED CT which contains more than 300 000 term definitions [6] and the National Cancer Institute ontology (NCI) consisting of more than 60 000 axioms [4]. Terminologies of this size and complexity cannot be developed and maintained without adequate automated versioning support. As a consequence, the development of ontology versioning tools and theoretical foundations for versioning have become a popular and an important research problem [5, 7, 9, 14–16].

In this paper we give an update on the CEX versioning tool which is the only purely logic-based tool for ontology versioning. The first version of CEX was presented in [10] and was able to compute a logical difference between acyclic $\mathcal{EL}$ terminologies that captures the different concept inclusions that follow from the two terminologies. More precisely, for any two acyclic $\mathcal{EL}$ terminologies and any signature Σ relevant for the comparison between the two terminologies, CEX computed a finite representation of the different concept inclusions over Σ that follow from one terminology but not the other. Recently, ontology based data access has become a major application of ontologies in general, and of $\mathcal{EL}$ terminologies in particular [12, 13, 17]. In this case, it is not sufficient to compare the derived concept inclusions of terminologies, but answers to instance queries or even conjunctive queries should be considered as well. Thus, we have extended CEX so as to cover three distinct types of logical differences: differences w.r.t. concept inclusions, answers to instance queries, and answers to conjunctive queries.

Moreover, CEX now admits role inclusions and range and domain restrictions, and so acyclic $\mathcal{ELH}^r$ terminologies rather than only acyclic $\mathcal{EL}$ terminologies can be compared. The algorithms and theory behind CEX are presented in [11]. In contrast to the update presented here, the version of CEX discussed in [11] cannot compute differences w.r.t. conjunctive queries. In this paper, we therefore focus on experiments that show how moving from concept and instance queries to conjunctive queries influences the performance of CEX and the number of differences detected between distinct versions of NCI.

2 Preliminaries

An *$\mathcal{ELH}^r$-terminology* is a finite set of role inclusions $r \sqsubseteq s$ and concept inclusions and equations of the form $A \sqsubseteq C$, $A \equiv C$, $\mathsf{ran}(r) \sqsubseteq C$, and $\exists r.\top \sqsubseteq C$ such that no concept name occurs more than once on the left-hand side, where A is a *concept name*, r, s are *role names*, $\mathsf{ran}(r)$ refers to the *range* of the role r and C, D are *$\mathcal{EL}$-concepts*, that is, expressions of the form $C := A \mid \top \mid C \sqcap D \mid \exists r.C$. (Complete definitions can be found in [11], see also [1] where $\mathcal{ELH}^r$ was introduced.) A terminology is *acyclic* if the defined concept is not used (directly or indirectly) it its definition. For example, a terminology containing inclusions $A \sqsubseteq \exists r.B$ and $B \sqsubseteq A$ is *not* acyclic. Instance data are represented by *ABox assertions* of the form $A(a)$ and $r(a, b)$, where a, b are *individual names*, A is a concept name and r is a role name. An *ABox* is a non-empty finite set of ABox-assertions. The semantics of $\mathcal{ELH}^r$ can be given by interpreting TBoxes and ABoxes as first-order (FO) sentences where concepts are formulas with one free variable, roles are binary predicates, and individual names are constants. For example, the inclusion $A \sqsubseteq \exists rB$ can be interpreted as $\forall x(A(x) \Rightarrow \exists y(r(x, y) \wedge B(y)))$. We use $\mathcal{T} \models \varphi$, or $(\mathcal{T}, \mathcal{A}) \models \varphi$, to denote that φ follows from $\mathcal{T}$, or $\mathcal{T} \cup \mathcal{A}$, respectively, in FO. An *instance query* α is of the form $r(a, b)$ or $C(a)$ with C an $\mathcal{EL}$-concept. A *conjunctive query (CQ)* is a FO-formula $q(\boldsymbol{x}) = \exists \boldsymbol{y} \psi(\boldsymbol{x}, \boldsymbol{y})$, where ψ is constructed from atoms $A(t)$ and $r(t, t')$ using conjunction and t, t' range over individual variables from $\boldsymbol{x}, \boldsymbol{y}$ and individual names. A *signature* Σ is a finite set of concept and role names, and a Σ-concept (Σ-query, etc.) is a concept (query, etc.) that only uses concept and role names from Σ.

3 The CEX2.5 System

CEX2.5[1] takes as input two acyclic $\mathcal{ELH}^r$ terminologies $\mathcal{T}_1, \mathcal{T}_2$ and a signature Σ and analyses the following three types of logical difference:

- the *Σ-concept difference* between $\mathcal{T}_1$ and $\mathcal{T}_2$ is the set $\mathsf{cDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ of all Σ-role and Σ-concept inclusions α in $\mathcal{ELH}^r$ such that $\mathcal{T}_1 \models \alpha$ and $\mathcal{T}_2 \not\models \alpha$;
- the *Σ-instance difference* between $\mathcal{T}_1$ and $\mathcal{T}_2$ is the set $\mathsf{iDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ of pairs of the form $(\mathcal{A}, \alpha)$, where $\mathcal{A}$ is a Σ-ABox and α a Σ-instance assertion such that $(\mathcal{T}_1, \mathcal{A}) \models \alpha$ and $(\mathcal{T}_2, \mathcal{A}) \not\models \alpha$; and

[1] Available under an open-source license at `http://www.csc.liv.ac.uk/~michel/software/cex2/`

– the *Σ-query difference* between $\mathcal{T}_1$ and $\mathcal{T}_2$ is the set $\mathsf{qDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ of pairs $(\mathcal{A}, q(\boldsymbol{a}))$, where $\mathcal{A}$ is a Σ-ABox, $q(\boldsymbol{x})$ a Σ-CQ, and $\boldsymbol{a}$ a tuple of individual names in $\mathcal{A}$ such that $(\mathcal{T}_1, \mathcal{A}) \models q(\boldsymbol{a})$ and $(\mathcal{T}_2, \mathcal{A}) \not\models q(\boldsymbol{a})$.

If for one of the three types of queries, the Σ-differences between $\mathcal{T}_1$ and $\mathcal{T}_2$ and between $\mathcal{T}_2$ and $\mathcal{T}_1$ are empty, then the two terminologies can be regarded as equivalent and replaced one by another in applications using Σ-symbols only and that type of queries. Notice that, for all three types of logical difference, if the Σ-difference between terminologies is not empty, then it is infinite. We distinguish between two modes in which CEX presents an approximation of this infinite Σ-difference to the user. First, it is shown in [11] that within every member of the Σ-difference, one can find an "elementary" difference which is either a role inclusion or

– for $\mathsf{cDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$: a concept inclusion $C \sqsubseteq D$ in which either C is a concept name or an expression of the form $\mathsf{ran}(r)$ or $\exists r.\top$; or D is a concept name;
– for $\mathsf{iDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$: a pair $(\mathcal{A}, \alpha)$ in which either $\mathcal{A}$ is a singleton ABox or α an atomic instance query;
– for $\mathsf{qDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$: a pair $(\mathcal{A}, \alpha)$ in which either $\mathcal{A}$ is a singleton ABox or α an atomic CQ.

We call C and $\mathcal{A}$ the left-hand side of such an elementary difference and D and α its right-hand side. The concept or role name of the atomic (or singleton) left or right-hand side of such an elementary difference is termed a *Σ-difference witness.* Every Σ-concept difference witness is a Σ-instance difference witness is a Σ-query difference witness. The converse holds for the left-hand sides in the concept/instance case and the right-hand sides in the instance/query case.

Example 1. Consider the following terminologies $\mathcal{T}_1$ and $\mathcal{T}_2$

$$\begin{array}{llll}
\mathcal{T}_1: & A \equiv \exists r.(A_1 \sqcap B_2) & \mathcal{T}_2: & A \sqsubseteq \exists r.(A_1 \sqcap B_2) \\
& A_2 \sqsubseteq B_2 & & A_2 \sqsubseteq B_2 \\
& E \sqsubseteq \exists s.F & & E \sqsubseteq \exists r_1.\top \sqcap \exists r_2.\top \\
& s \sqsubseteq r_1,\ s \sqsubseteq r_2 & &
\end{array}$$

and signature $\Sigma = \{A, A_1, A_2, E, r, r_1, r_2\}$. Then

1) A is the only Σ-concept difference witness (and it is a right-hand side witness): it is a Σ-concept difference witness since the inclusion $\exists r.(A_1 \sqcap A_2) \sqsubseteq A$ is an elementary difference (observe that $\mathcal{T}_1 \models \exists r.(A_1 \sqcap A_2) \sqsubseteq A$ but $\mathcal{T}_2 \not\models \exists r.(A_1 \sqcap A_2) \sqsubseteq A$). No other Σ-concept difference witness exists since one can show that all elementary members of $\mathsf{cDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ have A on its right-hand side.

2) Similarly, A is the only Σ-instance difference witness (and it is again a right-hand side witness): an elementary difference is given by the pair $(\mathcal{A}_1, q_1)$, where $\mathcal{A}_1 = \{r(a, b), A_1(b), A_2(b)\}$ and $q_1 = A(a)$. No elementary Σ-instance difference without the atom A on its right-hand side exists.

3) The Σ-query difference witnesses are given by A and E (and A is a right-hand side witness while E is a left-hand side witness): in this case one can

show that all elementary Σ-query differences either have the query $A(a)$ for some a on the right-hand side or the ABox $\{E(a)\}$ for some a on the left-hand side. Examples of such differences are the pairs $\{(\mathcal{A}_1, q_1), (\mathcal{A}_2, q_2)\}$ where $(\mathcal{A}_1, q_1)$ is as above, and $\mathcal{A}_2 = \{E(a)\}$ and $q_2 = \exists x(r_1(a,x) \wedge r_2(a,x))$. For the same terminologies and $\Sigma = \{E, F\}$, one can see that there are neither concept nor instance difference witnesses; however, as $(\mathcal{T}_1, \{E(a)\}) \models \exists x.F(x)$ but $(\mathcal{T}_2, \{E(a)\}) \not\models \exists x.F(x)$, there is a Σ-query difference between the terminologies and E is its (left-hand side) witness.

Note that the set of Σ-difference witnesses is uniquely determined by $\mathcal{T}_1, \mathcal{T}_2$ and Σ and gives a rather abstract description of the Σ-difference. This set is empty iff no Σ-difference exists and can be computed in polynomial time, for all three types of queries. In its basic mode, CEX2.5 computes the set of all Σ-concept, instance and query witnesses and presents them (together with the information whether they are left or right-hand side witnesses) to the user. For a more detailed analysis of the Σ-difference between the two input terminologies $\mathcal{T}_1$ and $\mathcal{T}_2$, in its advanced mode CEX2.5 can also compute *examples* of elementary members of $\mathsf{cDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$, $\mathsf{iDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$, and $\mathsf{qDiff}_\Sigma(\mathcal{T}_1, \mathcal{T}_2)$ which illustrate why certain concept names are concept, instance, or query difference witnesses.

4 Experimental Results

In [11], we have conducted a detailed experimental evaluation of the performance of CEX2.5 in the concept and instance difference case. In this report we, therefore, focus on the CQ case and (a) compare the performance of CEX2.5 for the CQ case with its performance for the concept and instance case, and (b) compare the number of difference witnesses detected in the CQ case with the number of difference witnesses detected in the concept/instance case. The CEX2.5 system is implemented in OCaml, and it uses the reasoner CB [8] internally as classification engine. The experiments were conducted on PCs equipped with an Intel Core i5-2500 CPU and 4 GiB of main memory.

First, CEX2.5 is used to compare 71 consecutive acyclic $\mathcal{ELH}^r$-versions of the NCI Thesaurus.[2] For any two consecutive versions NCI_n and NCI_{n+1} within the considered range, we compute all instance and query difference witnesses together with corresponding examples for $\mathcal{T}_1 = \mathrm{NCI}_{n+1}$ and $\mathcal{T}_2 = \mathrm{NCI}_n$ and signature $\Sigma = \mathsf{sig}(\mathrm{NCI}_n) \cap \mathsf{sig}(\mathrm{NCI}_{n+1})$ with CEX2.5. The results are given in Table 1, where only those comparisons are reproduced for which there are query difference witnesses which are not instance difference witnesses. The first two columns give the NCI versions, $|\mathsf{qRhs}_\Sigma(\cdot,\cdot)|$ is the number of right-hand side Σ-query difference witnesses (which always coincides with the number $|\mathsf{iRhs}_\Sigma(\cdot,\cdot)|$ of right-hand side Σ-instance difference witnesses). $|\mathsf{qLhs}_\Sigma(\cdot,\cdot)|$ and $|\mathsf{iLhs}_\Sigma(\cdot,\cdot)|$ are the number of left-hand side query and instance difference witnesses, respectively. One can see that in some cases there are significantly more query difference

[2] Full versions are available from `http://evs.nci.nih.gov/ftp1/NCI_Thesaurus/`. We refer the reader to [5] for additional information on NCI versions and note that the full versions contain inclusions that are not in acyclic $\mathcal{ELH}^r$.

$\mathcal{T}_1$	$\mathcal{T}_2$	$\lvert\mathsf{qRhs}_\Sigma(\mathcal{T}_1,\mathcal{T}_2)\rvert =$ $\lvert\mathsf{iRhs}_\Sigma(\mathcal{T}_1,\mathcal{T}_2)\rvert$	$\lvert\mathsf{qLhs}_\Sigma(\mathcal{T}_1,\mathcal{T}_2)\rvert$	$\lvert\mathsf{iLhs}_\Sigma(\mathcal{T}_1,\mathcal{T}_2)\rvert$	Time (s) (query)	Time (s) (instance)
03.12e	03.12a	49	1747	289	177.78	14.85
04.03n	04.02h	431	8277	5494	14.02	13.74
04.06i	04.05f	99	1147	1080	48.31	39.50
05.05d	05.03d	1007	2683	747	513.78	17.17
06.01c	05.12f	798	2066	2053	449.8	22.39
07.01d	06.12d	814	290	222	41.19	40.99

Table 1. Detailed Results for Comparisons Between Consecutive $\mathcal{ELH}^r$-versions of the NCI Thesaurus Leading to Additional Conjunctive Query Differences

witnesses than instance difference witnesses. In fact, one of the conclusions one can draw from this table is that the instance difference between terminologies is not necessarily a good approximation of the query difference. Consequently, when terminologies are used to access instance data using CQs, a comparison at the concept or instance level cannot always replace an analysis tailored for CQs. Secondly, one can see that the time required to compute query witnesses can be significantly longer than the time necessary for detecting instance witnesses; we will comment on the reasons below.

To analyse the impact of the size of signatures on the running time of CEX2.5 and on the number of difference witnesses, in our second experiment CEX2.5 is used to compute concept, instance and query difference witnesses together with corresponding examples for $\mathcal{T}_1 = \mathrm{NCI}_{06.12d}$ and $\mathcal{T}_2 = \mathrm{NCI}_{07.01d}$ on randomly-generated signatures $\Sigma \subseteq \mathsf{sig}(\mathcal{T}_1) \cap \mathsf{sig}(\mathcal{T}_2)$. The signatures were composed of a varying number of concept names and 60 randomly-selected roles. For each considered sample size of concept names we generated 10 random signatures. The computation times and the number of difference witnesses that were detected on average for each sample size are shown in Fig. 1. It can be seen that the time required to compute concept difference witnesses almost coincides with the computation time necessary for computing instance difference witnesses. As in the previous experiment, on average there are significantly more query difference witnesses than concept and instance difference witnesses (of which there are almost the same number). Moreover, in contrast to the concept and instance difference case, the computation time in the query difference case increases with the number of concept names present in the considered signatures. Less than 287 MiB of memory were required in each of the comparisons involving NCI versions.

To evaluate the performance of CEX2.5 on very large terminologies, we compare three consecutive versions of SNOMED CT (January 2009, July 2009, and January 2010). We use CEX2.5 to compute instance and query difference witnesses with and without examples on the shared signature between two consecutive versions. All three versions of SNOMED CT considered have the same role names which are, therefore, also in the shared signature. In contrast to the experiments for NCI, in this case the set of query difference witnesses turned out to coincide with the set of instance difference witnesses and the computation times almost coincide: on average, 683 seconds for the instance witnesses and 672 seconds for the CQ witnesses. The running time rose to 1028 seconds

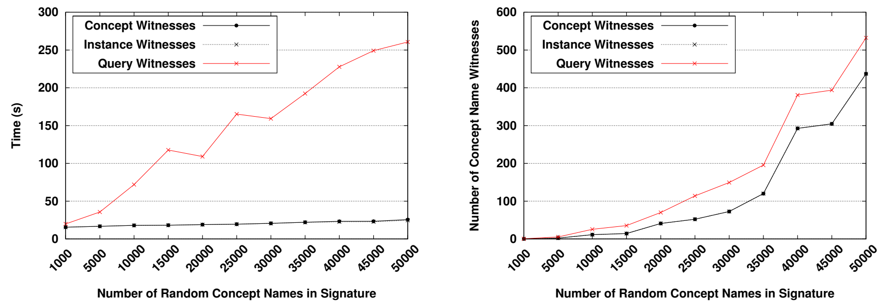

Fig. 1. Computation Time Required and Number of Difference Witnesses Detected between two Consecutive NCI Versions on Random Signatures

and, respectively, 1006 seconds when examples were additionally computed. On average 2.84 GiB and, respectively, 2.92 GiB of memory were required for the computation.

Finally, in the experiments above, 687 813 examples of elementary differences between terminologies were computed. The average length (i.e., the number of occurrences of concept and role names) of an example was 5.98 with a maximal length of 98. It follows that in most cases the examples generated by CEX2.5 are sufficiently small to be analysed by a human user (note that, in theory, in the worst case minimal examples are of exponential size [11]).

We close with a discussion as to why in the NCI experiments computing left-hand side Σ-query difference witnesses takes longer than computing left-hand side Σ-instance difference witnesses (and why this is not the case for SNOMED CT). To check whether $A \in \Sigma$ is such a witness for $\mathcal{T}_1, \mathcal{T}_2$ both algorithms check whether there is a certain Σ-*simulation* between the minimal models $\mathcal{I}_{\mathcal{T}_1,\{A(a)\}}$ and $\mathcal{I}_{\mathcal{T}_2,\{A(a)\}}$ for the knowledge bases $(\mathcal{T}_1, \{A(a)\})$ and $(\mathcal{T}_2, \{A(a)\})$ [11, 12]. The difference between the two cases is that for the instance difference witnesses a "standard" Σ-simulation between the node for a in $\mathcal{I}_{\mathcal{T}_1,\{A(a)\}}$ and the node for a in $\mathcal{I}_{\mathcal{T}_2,\{A(a)\}}$ is sufficient, whereas for the query difference the simulation has to, in addition, respect *intersections between Σ-roles* and has to be *global* (every node in $\mathcal{I}_{\mathcal{T}_1,\{A(a)\}}$ has to be simulated). The second condition is costly since it implies that one has to consider all nodes of $\mathcal{I}_{\mathcal{T}_1,\{A(a)\}}$ and find simulating nodes in $\mathcal{I}_{\mathcal{T}_2,\{A(a)\}}$ rather than consider nodes reachable from the node for a via Σ-paths only. In general, it therefore appears to be unavoidable that computation times for CQ are longer than for concept and instance queries. The SNOMED CT experiment is different: in this case Σ contains all role names in both terminologies and so any simulation of the node for a is a global simulation already.

We note that because of their importance in model checking and abstraction, a large variety of highly optimized algorithms computing simulations between Kripke models have been developed (e.g. [2,3]). In our implementation, however, we do *not* first construct the (potentially very large) minimal models and then check for Σ-simulation, but we check for Σ-simulation on-the-fly making heavy use of the condition that $\mathcal{T}_1$ and $\mathcal{T}_2$ are acyclic terminologies.

References

1. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope further. In: Proceedings of OWLED 2009. CEUR Workshop Proceedings, vol. 529. CEUR-WS.org (2008)
2. Crafa, S., Ranzato, F., Tapparo, F.: Saving space in a time efficient simulation algorithm. Fundamenta Informaticae 108(1-2), 23–42 (2011)
3. van Glabbeek, R.J., Ploeger, B.: Correcting a space-efficient simulation algorithm. In: Proceedings of CAV 2008. LNCS, vol. 5123, pp. 517–529. Springer (2008)
4. Golbeck, J., Fragaso, G., Hartel, F., Hendler, J., Oberhaler, J., Parsia, B.: The national cancer institute's thesaurus and ontology. Journal of Web Semantics 1(1), 75–80 (2003)
5. Gonçalves, R.S., Parsia, B., Sattler, U.: Analysing multiple versions of an ontology: A study of the NCI thesaurus. In: Proceedings of DL 2011. CEUR Workshop Proceedings, vol. 745. CEUR-WS.org (2011)
6. IHTSDO: SNOMED Clinical Terms User Guide. The International Health Terminology Standards Development Organisation (IHTSDO) (2008)
7. Jiménez-Ruiz, E., Grau, B.C., Horrocks, I., Llavori, R.B.: Supporting concurrent ontology development: Framework, algorithms and tool. Data & Knowledge Engineering 70(1), 146–164 (2011)
8. Kazakov, Y.: Consequence-driven reasoning for Horn SHIQ ontologies. In: Proceedings of IJCAI 2009. pp. 2040–2045 (2009)
9. Klein, M.C.A., Fensel, D., Kiryakov, A., Ognyanov, D.: Ontology versioning and change detection on the web. In: Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web, LNCS, vol. 2473, pp. 247–259. Springer Verlag, Berlin/Heidelberg, Germany (2002)
10. Konev, B., Walther, D., Wolter, F.: The logical difference problem for description logic terminologies. In: Proceedings of IJCAR 2008. LNCS, vol. 5195, pp. 259–274. Springer Verlag, Berlin/Heidelberg, Germany (2008)
11. Konev, B., Ludwig, M., Walther, D., Wolter, F.: The logical diff for the lightweight description logic $\mathcal{EL}$, submitted, available under `http://www.csc.liv.ac.uk/~frank/publ/publ.html`
12. Lutz, C., Toman, D., Wolter, F.: Conjunctive query answering in the description logic $\mathcal{EL}$ using a relational database system. In: Proceedings of IJCAI 2009. pp. 2070–2075. AAAI Press, Menlo Park, CA, USA (2009)
13. Mei, J., Liu, S., Xie, G.T., Kalyanpur, A., Fokoue, A., Ni, Y., Li, H., Pan, Y.: A practical approach for scalable conjunctive query answering on acyclic $\mathcal{EL}^+$ knowledge base. In: International Semantic Web Conference. pp. 408–423 (2009)
14. Noy, N.F., Musen, M.A.: PromptDiff: A fixed-point algorithm for comparing ontology versions. In: Proceedings of AAAI-02. pp. 744–750. AAAI Press, Menlo Park, CA, USA (2002)
15. Oliver, D.E., Shahar, Y., Shortliffe, E.H., Musen, M.A.: Representation of change in controlled medical terminologies. Artificial Intelligence in Medicine 15(1), 53–76 (1999)
16. Palma, R., Haase, P., Corcho, O., Gómez-Pérez, A.: Change representation for OWL 2 ontologies. In: Proceedings of OWLED 2009. CEUR Workshop Proceedings, vol. 529. CEUR-WS.org (2009)
17. Poggi, A., Lembo, D., Calvanese, D., Giacomo, G.D., Lenzerini, M., Rosati, R.: Linking data to ontologies. Journal of Data Semantics 10, 133–173 (2008)