Towards Searching for Transformation Patterns in Support of Language Profiling

Alexander Šimko[1] and Ondřej Zamazal[2]

[1] Comenius University in Bratislava, Slovakia
simko@fmph.uniba.sk
[2] University of Economics, Prague, Czech Republic
ondrej.zamazal@vse.cz

Motivation Semantic web ontology tools usually support a certain set of logical operators. In many cases this set of operators is not sufficient to completely capture the semantics of the OWL language. Thus, these tools cannot be used on certain ontologies or they provide incomplete reasoning results. In both cases, the transformation of the input ontology can improve the situation. In particular, the ontology can be transformed into a version that only uses the supported operators. Although construct replacement can be done inside the ontology tools, doing this outside the tools gives the user more flexibility because (s)he can design a transformation that is not directly hard-coded into the tool. Since it is hard to manually design a suitable transformation pattern, we try to automatically search for suitable transformation patterns (TPs) by *evolutionary algorithm* (EA) to support ontology profiling. The paper extends our previous work of general construct replacing approach [1].

An Evolutionary Algorithm for Searching of TPs At the head of our automatic transformation of forbidden language constructs in ontology is the *evolutionary algorithm*[3] searching the space of all TPs for a TP that keeps the most of the logical meaning. The EA keeps a population of TPs, which are generated randomly at the beginning. At each step, the algorithm creates a new population by quasi-randomly selecting TPs from the old population. Better TPs have bigger probability to enter into a new population. Moreover, TPs that are added to the new population are crossed and mutated with a given probability. After the maximal number of steps is reached, the best TP found so far is returned.

For EA a TP is represented as a triple (op_1, op_2, pt), where op_1 and op_2 are lists of axioms within a source (resp. target) OP and a pattern transformation pt is a list of transformation links. Axioms are written in a frame-based variant of Manchester syntax,[4] and are represented using syntax trees allowing us to easily mutate axioms. A transformation link is simply a pair of placeholders from corresponding OPs. Numbers of axioms in source/target OPs and transformation links are randomly generated but considering their respective assigned maxima.

A crossover of two TPs $tp_1 = (op_{11}, op_{12}, pt_1)$ and $tp_2 = (op_{21}, op_{22}, pt_2)$ is done in component-wise manner, i.e. op_{11} is crossed with op_{21} etc. Two OPs are

[3] Implementation is available at https://code.google.com/p/tpgen/
[4] http://www.w3.org/TR/owl2-manchester-syntax/

crossed by generating an index and switching the elements of the lists (similarly for two PTs). A TP is mutated by mutating each axiom in both OPs and each transformation link with a given probability. Since an axiom is represented as a syntactic tree it is mutated by visiting each node starting from the root and regenerating the sub-tree using the grammar rules corresponding to the symbol in the node. A transformation link is mutated by randomly switching a placeholder.

Fitness value is computed for each TP and given ontology so that the ontology is transformed accordingly and then we compute averaged F1-measure on results of five simple SPARQL queries (e.g. equivalency, subsumption, instantiation) considering results from original ontology as *expected results* and results from transformed ontology as *returned results*. By employing *Pellet* reasoner,[5] fitness value also includes inferred axioms. TPs with higher fitness are better since they preserve more semantic properties of an original ontology. If TP does not contain constructs to be replaced in source OP, fitness value is zero.[6]

A set of forbidden language constructs is given as an input to the EA. The respective probabilities of a target OP are set to zero. This ensures that the forbidden language constructs are not present after the generated TP is applied to an input ontology. In order to reduce search space, input ontology is analyzed and, as a result, language constructs not present in the input ontology are zeroed in a source OP.

Future Work We plan to restrict generations of transformation patterns in terms of axioms present in given ontology wrt. source ontology pattern and in terms of some background knowledge about possible construct substitutes regarding target ontology pattern. Next, we also plan to test more sophisticated computation of fitness value, e.g., considering semantic ontology diff. In all, our work needs a substantial amount of experimentation (and parameters tuning) with highlighted planned features.

Although our work is still in its early phase, we think that this approach can make sense for an automatic transformation of complex axioms rather than transformation of simple axioms with one construct which can be easily done by manually tailored transformation patterns.

Acknowledgement This work was supported from the Czecho-Slovak bilateral project nos. 7AMB12SK020 (AIP) and SK-CZ-0208-11 (APVV). Ondřej Zamazal has been supported by the CSF grant no. 14-14076P.

References

1. Šváb-Zamazal O., Schlicht A., Stuckenschmidt H., Svátek V. Constructs Replacing and Complexity Downgrading via a Generic OWL Ontology. In: SOFSEM 2013.

[5] `http://pellet.owldl.com/`

[6] The equation for fitness measure, examples of its computation, examples for mutation/crossover operations etc. are at the supplementary web: `http://owl.vse.cz:8080/OWLED-2014/`