Evaluating the Generality of Disjunctive Model Faithful Acyclicity on OWL ontologies

Lukas Gerlach

Knowledge-Based Systems Group
Technische Universität Dresden, Germany

Abstract. The *chase* is a well-studied, sound and complete algorithm that is used in different variants as a basis for reasoning tasks over *(disjunctive) existential rules*. Since termination of the chase is undecidable, *acyclicity notions*, i.e. sufficient conditions for termination, like *model faithful acyclicity (MFA)* for the skolem chase and *restricted model faithful acyclicity (RMFA)* for the restricted chase are introduced. The recently developed acyclicity notion *disjunctive model faithful acyclicity (DMFA)* for the *disjunctive skolem chase* promises improvements for detecting termination over existing notions like MFA in theory. We further know that RMFA captures DFMA while RMFA itself is not sound for the disjunctive skolem chase.
We evaluate the generality of DMFA in practice compared to MFA and RMFA on rule sets that we obtain from real-world OWL ontologies in the Oxford ontology repository (OXFD) and the dataset of the OWL reasoner evaluation 2015 (ORE15). Our results show that DMFA achieves practical improvements over MFA that narrow the gap towards RMFA. Our findings motivate further research regarding the disjunctive skolem chase in general and the development of sufficient conditions for non-termination of the disjunctive skolem chase in particular.

Evaluation of the Generality of DMFA

We evaluate our previously defined notion of DMFA [4] on real-world OWL ontologies. Recall that we are particularly interested in the disjunctive skolem chase since, while terminating on less rule sets than the restricted chase, it promises to be more efficiently implementable in practice, for example using ASP solvers with lazy grounding. In general, this work can be considered an extension to our theoretical results and we assume the reader to be familiar with the definitions and results of [4] but we also point explicitly to important results at some points.

We use the Oxford ontology repository (OXFD)[1] containing 797 ontologies and the dataset of the OWL Reasoner Evaluation 2015 (ORE15)[2] [5,6], which also contains some ontologies from OXFD and features 1920 ontologies in total.

[1] OXFD - `https://www.cs.ox.ac.uk/isg/ontologies/`
[2] ORE15 - `https://zenodo.org/record/18578`

We perform the following steps for the evaluation:

1. Normalize the OWL ontologies.
2. Translate the normalized ontologies into sets of disjunctive existential rules.
3. Check if the obtained rules sets are MFA, DMFA, and RMFA, respectively.

The core part of our evaluation is the check itself (3). Since real-world datasets featuring existential rules rarely exist, translated OWL ontologies promise to produce results that are practically more relevant. Therefore, we use steps (1) and (2) to obtain suitable disjunctive existential rule sets for the check in (3). We tailor steps (1) and especially (2) towards this purpose to a large extent, e.g. we ignore all ABox-axioms and also some other axioms already in the translation. In general, we treat classes and properties as unary predicates and binary predicates, respectively.

Normalization and Translation into Disjunctive Existential Rules

We normalize the OWL ontologies using an existing tool.[3] The normalizer drops all axioms that do not carry logical meaning.[4] That is, we remove all `Declaration` axioms and `Annotation` axioms. Additionally, we drop all axioms featuring `Datatypes` or `DataProperties`. The normalizer flattens all remaining axioms such that we obtain only axioms of the form in Figure 1. By that we produce a *conservative extension* of the original ontology; that is, every model of the normalized ontology is also a model for the original ontology and every model of the original ontology can be extended to a model of the normalized ontology by providing suitable interpretations for the freshly introduced classes and properties.

For the translation, we use our own implementation, which is based on the Rulewerk library,[5] which uses OWLAPI[6] in turn. To be more precise, we use a fork of Rulewerk[7] where we add basic support for disjunctions in rules. However, we do not add support for reasoning with disjunctive rules in Rulewerk. For further implementation details, refer to the repository of the translation project.[8]

In the translation, we refine the normalized ontologies at first according to the following steps. We remove all ABox-axioms since we do not need to consider them for the DMFA-check. Additionally, we remove all tautological axioms, e.g. axioms of the form $\bot \sqsubseteq A$ or $A \sqsubseteq \top$. Since DMFA does not support equality, we omit all axioms of type (5) and (6). In later work, we may use an appropriate axiomatization as suggested in [3] to be able to treat equality properly.

[3] Ontology Normalizer - `https://github.com/dcarralma/OntologyNormalizer`
[4] Axiom Overview - `https://www.w3.org/TR/owl2-syntax/#Axioms`
[5] Rulewerk - `https://github.com/knowsys/rulewerk`
[6] OWLAPI - `https://github.com/owlcs/owlapi`
[7] Rulewerk Fork - `https://github.com/monsterkrampe/rulewerk/tree/feature/disjunctions-in-rules`
[8] Translation - `https://gitlab.com/m0nstR/owl-to-disjunctive-existential-rules-converter`

$A_1 \sqcap \cdots \sqcap A_n \sqsubseteq B$	$A_1(x) \wedge \cdots \wedge A_n(x) \rightarrow B(x)$	(1)
$A \sqsubseteq B_1 \sqcup \cdots \sqcup B_{n'}$	$A(x) \rightarrow B_1(x) \vee \cdots \vee B_{n'}(x)$	(2)
$A \sqsubseteq \forall R.B$	$A(x) \wedge R(x,y) \rightarrow B(y)$	(3)
$A \sqsubseteq \geq m R.B$	(see Example 1)	(4)
$A \sqsubseteq \leq m' R.B$	(requires equality; dropped)	(5)
$A \sqsubseteq \{ a_1, \ldots, a_l \}$	(requires equality; dropped)	(6)
$R_1 \circ \cdots \circ R_k \sqsubseteq S$	$R_1(x_1,x_2) \wedge \cdots \wedge R_k(x_k,x_{k+1}) \rightarrow S(x_1,x_{k+1})$	(7)
$R \sqcap S \sqsubseteq \bot$	$R(x,y) \wedge S(x,y) \rightarrow \bot$	(8)
$A \sqsubseteq \exists R.\mathrm{Self}$	$A(x) \rightarrow R(x,x)$	(9)
$\exists R.\mathrm{Self} \sqsubseteq A$	$R(x,x) \rightarrow A(x)$	(10)

Here, $A, A_1, \ldots, A_n, B, B_1, \ldots, B_{n'}$ are classes, $R, R_1, \ldots, R_k, S$ are (inverse) properties, $a_1, \ldots, a_l$ are individuals, and m, m' are integers ≥ 0.

Fig. 1. Normal Form and corresponding Disjunctive Existential Rules

We also ensure that each predicate only occurs with a single arity in the translated rule set by adding appropriate suffixes to class names and property names in the implementation but we do not take this into account here for brevity. This is necessary, since OWL Full allows to reuse the same name for a class and a predicate within an ontology.

We translate all remaining axioms into existential rules one by one as presented in Figure 1. Ideas like this have also been applied to similar normal forms [2]. The handling for axioms of type (4) is a little more involved and we illustrate it using the following example.

Example 1. For $A \sqsubseteq \geq 3R.B$, we derive the following rules:

	$R_1(x,y) \rightarrow R(x,y)$
$A(x) \rightarrow \exists z_1.(R_1(x,z_1) \wedge B(z_1))$	$R_2(x,y) \rightarrow R(x,y)$
$A(x) \rightarrow \exists z_2.(R_2(x,z_2) \wedge B(z_2))$	$R_3(x,y) \rightarrow R(x,y)$
$A(x) \rightarrow \exists z_3.(R_3(x,z_3) \wedge B(z_3))$	$R_1(x,y) \wedge R_2(x,y) \rightarrow \bot$
$R(x,y) \rightarrow R_1(x,y) \vee R_2(x,y) \vee R_3(x,y)$	$R_2(x,y) \wedge R_3(x,y) \rightarrow \bot$
	$R_1(x,y) \wedge R_3(x,y) \rightarrow \bot$

Here, R_1, R_2, and R_3 are fresh predicates that, intuitively speaking, form a partitioning of R. The necessity of the rule $R(x,y) \rightarrow R_1(x,y) \vee R_2(x,y) \vee R_3(x,y)$ can be object to discussion. If another translated axiom also derives facts for R, then the rule ensures that this is also reflected in R_1, R_2, or R_3, respectively. Though, omitting this rule does not seem to cause issues for our matter. It is only for the special case $m = 1$ that this would make a difference namely considering termination of the restricted chase. But for $m = 1$, we can simplify the translation using a single rule $A(x) \rightarrow \exists z.R(x,z) \wedge B(z)$ anyway. Still, to keep faithful to the intuition, we include the rule in the translation. ▲

Beside the axioms in Figure 1, the normalized ontologies may also contain SWRL rules and we usually ignore such ontologies. However, if an ontology only contains SWRL rules that are essentially Datalog rules containing only class and property atoms, then we translate these rules in the obvious way. Also note that only few of the ontologies that we consider contain SWRL rules at all.

In some rules we use $\bot$ to indicate inconsistencies. Also, $\bot$ and $\top$ may occur in normalized axioms that are not tautological. In practice, we represent $\bot$ and $\top$ by the special predicates `OWLNothing` and `bottomObjectProperty` or `OWLThing` and `topObjectProperty`, respectively, in spirit of the corresponding OWL axioms. Still, for our matter these are just plain unary and binary predicates. We also treat these predicates like any other predicate, i.e. without special semantic meaning. Since we remove tautological axioms, we can be sure that no fact is ever derived from `OWLNothing` or `bottomObjectProperty`. We axiomatize the semantics of $\top$ using auxiliary rules. For each non-special unary predicate P or binary predicate R, we add the rule $P(x) \to \texttt{OWLThing}(x)$ or $R(x, y) \to \texttt{OWLThing}(x) \wedge \texttt{OWLThing}(y)$, respectively. Additionally, we add a single rule $\texttt{OWLThing}(x) \wedge \texttt{OWLThing}(y) \to \texttt{topObjectProperty}(x, y)$.

Altogether, the translation already gives good hints where to expect differences for MFA, DMFA, and RMFA in the actual evaluation. The only axioms that introduce existential rules are those of type (4). For RMFA these may be blocked but for DMFA (and MFA) this is never the case. Furthermore, only axioms of type (2) and (4) introduce disjunctive rules and especially we never obtain rules that contain both disjunctions and existentially quantified variables. In particular, the disjunctive rules for (4) do not influence DMFA. Broadly speaking, this is because their head predicates only occur in other auxiliary rules in (4) and they do not lead to the derivation of new facts in this case. Therefore it does not matter if they are blocked in the DMFA-check. Hence, only rules for the axioms of type (2) may impact DMFA in comparison to MFA since blocking for these rules may prevent other rules from being applied later on.

Implementation of the DMFA-Check

The general outline of the DMFA implementation[9] is presented in Algorithm 1. As for the translation, the implementation is based on Rulewerk, which allows us to parse the rule sets obtained from the translation step and also offers bindings to VLog, which we use for Datalog reasoning. In essence, for a given rule set R, we compute $\mathsf{DMFA}(R)$ (see Definition 50 in [4]) step by step starting on the critical instance of R and checking for cyclic terms in each iteration.

To perform the DMFA-check, we alternate between the computation of the Datalog closure of the current fact set in *newFactsFromVlogDatalogClosure* and the application of all triggers for non-Datalog rules that are active and not blocked w.r.t. the latest Datalog closure result in *newFactsFromUnblockedNonDatalogRules*. Note that the latter function does not apply rules exhaustively. Both functions keep track of already computed facts internally such that we

[9] DMFA-Implementation `https://gitlab.com/m0nstR/dmfa-checker`

Algorithm 1: *dmfaCheck*

Input: Ruleset R
Output: Is R DMFA?

$\Delta F \coloneqq \mathit{criticalInstance}(R)$
$\mathit{vlogReasoner} \coloneqq \mathit{VlogReasoner}(R)$ `// uses VLog bindings of Rulewerk`
$\mathit{exisReasoner} \coloneqq \mathit{ExistentialReasoner}(R)$
while $\Delta F \neq \emptyset$ **do**
    `// keeps track of already computed facts internally`
    $\Delta F \coloneqq \Delta F \cup \mathit{vlogReasoner.newFactsFromVlogDatalogClosure}(\Delta F)$
    `// non-exhaustive application`
    `// keeps track of already computed facts internally`
    $\Delta F \coloneqq \mathit{exisReasoner.newFactsFromUnblockedNonDatalogRules}(\Delta F)$
    if $\mathit{hasCyclicTerm}(\Delta F)$ **then**
        return *False*
    end
end
return *True*

only need to pass newly derived facts to them. For *newFactsFromUnblockedNonDatalogRules*, we achieve a semi-naive implementation such that each possible application is done at most once. Note that this does not necessarily prevent the recomputation of some existing facts but we filter existing facts before returning the result.

For the Datalog closure in VLog, a similar approach does not seem to be feasible since the VLog bindings do not offer to reuse reasoning results if the underlying facts change. Inside *newFactsFromVlogDatalogClosure*, we pass only the initial facts and facts that are obtained from non-Datalog rules to VLog. By that, facts from Datalog rules are recomputed in each iteration but in practice this seems to be faster in some cases than passing already computed Datalog facts to VLog. From the VLog bindings, we then only query the facts that feature predicates that also occur in non-Datalog rules. We keep track of these resulting facts internally and filter existing ones out before returning the Datalog closure.

Theoretically, VLog also supports existential rules without disjunctions but it uses nulls instead of functional terms. We could derive function names from the nulls in VLog but the blocked check that is used in the DMFA-check also relies heavily on the structure of the functional terms, which we cannot obtain from VLog directly. Therefore, we implement the existential reasoning part ourselves in *newFactsFromUnblockedNonDatalogRules*.

We can also implement a check for RMFA instead of DMFA by altering the blocked check. Broadly speaking, we only need to replace a $\subseteq$-check by a $\models$-check. This is also not too hard to do in practice. For the theoretical definitions, refer to Definitions 47 and 59 in [4], respectively, as well as [1].

We also want to note that we vary from the formal definition a bit in the implementation of the blocked check since it needs to run fairly frequently during the DMFA/RMFA-check. For existential rules that contain disjunctions, we have

to run the blocked check for every individual substitution. If we check RMFA instead of DMFA, we even need to do this for non-disjunctive existential rules. For DMFA, non-disjunctive rules cannot be blocked. To simplify the blocked check for a trigger $\langle \rho, \theta \rangle$, we check if we already have $sk(H^k_\rho)\theta \subseteq F$ or $F \models \exists \vec{z}_k.H^k_\rho(\vec{x}_k, \vec{z}_k)$, respectively, where F is the set of facts that have been derived up until this point in the DMFA/RMFA-check and H^k_ρ denotes the k-th disjunct in the head of ρ. If this is not the case, then the trigger $\langle \rho, \theta \rangle$ cannot be blocked. Without going too much into detail here, this follows from a similar argument as for the proof of Theorem 49 in [4] for both DMFA and RMFA. Intuitively, for a trigger $\langle \rho, \theta \rangle$, the set of facts F that have been derived up until this point in the DMFA/RMFA-check subsumes the set of facts that is required to derive the terms in the body of ρ with θ applied. If $sk(H^k_\rho)\theta \subseteq F$ or $F \models \exists \vec{z}_k.H^k_\rho(\vec{x}_k, \vec{z}_k)$ is already false for F, then the corresponding check will also be false for the subsumed fact set used in the blocked check. Otherwise, we perform the blocked check as formally described for DMFA [4] or RMFA [1], respectively.

Note that for the Datalog closure in the blocked check, we do not use the VLog implementation but our own Datalog reasoner implementation, which is just a special case of the one that we already use for the existential rule reasoning except that we apply rules in the Datalog closure exhaustively. Since the fact sets are rather small in the blocked check, the overhead of calling VLog seems to be bigger for some rule sets than the arguably less performant implementation of ours.

Apart from computing Datalog closures, VLog also allows for checking MFA and RMFA directly but it has no support for disjunctions. We can still use the method of replacing disjunctions by conjunctions to run the native VLog checks on the rule sets. Since MFA is not defined for disjunctive existential rules, we would need to do this replacement anyway. This way of checking MFA for disjunctive existential rules is sound according to Proposition 37 in [4]. For RMFA however, this replacement may yield unsound results as shown in the following example.

Example 2. Consider the rule set R that consists of the following rules:

$$P(x, y) \to \exists z.P(y, z)$$
$$P(x, y) \to P(y, x) \vee P(x, x)$$

We have that R is not terminating w.r.t. the restricted chase, e.g. for the fact set $\{ P(a, b) \}$ if we prioritize applications of the first rule. If we replace disjunctions by conjunctions, the second rule becomes a Datalog rule so it is prioritized over the first one by definition [1] and the modified rule set is terminating. This also reflects in the RMFA check since the blocked check takes Datalog rules into account. Hence, the modified rule set is RMFA although the original rule set is not terminating w.r.t. the restricted chase. ▲

Evaluation Results

After normalizing and translating OXFD and ORE15, we obtain 789 and 1888 rule sets according to Figure 2, respectively, that we can run our actual checks

	initially	after Normalization	after Translation	at least one $\exists$-rule and $\vee$-rule	all checks finished
OXFD	797	791	789	137	103
ORE15	1920	1910	1888	613	514

Fig. 2. Number of Datasets

on. A minority of the ontologies was not successfully normalized or translated due to parsing errors, unsupported SWRL rules or resource limitations. For the evaluation, we are mainly interested in rule sets that contain disjunctions, since we know that MFA and DMFA exactly coincide for rule sets without disjunctions according to Proposition 57 in [4]. Additionally, we omit rule sets that do not feature existentially quantified variables since those rule sets are trivially MFA, DMFA, and RMFA. For each rule set we run the checks for MFA, DMFA, and RMFA one after the other with timeouts for each individual check. For example, if the check for MFA reaches its timeout or if an error occurs, e.g. if the program ran out of memory, the later checks for DMFA and RMFA are skipped. We take this shortcut here since our goal is to evaluate the generality of DMFA and not the performance of our implementation in the first place. Therefore, we only take those rule sets into account for which all checks produce a result. By that, we obtain results for 103 and 514 rule sets for OXFD and ORE15, respectively. The majority of unsuccessful checks happen for large rule sets with more than 1000 rules with existentially quantified variables. For very few rule sets, parsing errors occurred which are fixable by hand but we decided against this to make it easier to reproduce the results. For an overview of the number of datasets after each step, refer to Figure 2.

Before going into the main results of the evaluation, we want to briefly discuss the non-disjunctive rule sets as well. We know that RMFA is more general on (non-disjunctive) rule sets than MFA/DMFA. As a sanity check for the correctness of our implementation, we also run checks on some of the non-disjunctive rule sets. By that for OXFD, we find that out of 508 rule sets that we checked, 47 rule sets are not MFA/DMFA and 7 of these are RMFA. For ORE15, we find that out of 579 rule sets that we checked, 47 rule sets are not MFA/DMFA and 8 of these are RMFA. This is worthwhile to keep in mind as a magnitude when looking at the main evaluation results in the following.

For the main part of the evaluation that takes only the disjunctive rule sets into account, the numbers of rule sets that are MFA, DMFA, and RMFA, respectively, can be found in Figures 3 and 4. The results are grouped by the number of rules with disjunctions (columns) and the number of rules with existentially quantified variables (rows). Each cell contains the number of rule set that are MFA, DMFA, and RMFA in that order separated by slashes as well as the total number of rule sets that fit in this cell in parentheses. For example in Figure 3, the top-left cell states that of the 15 rule sets with 1-9 rules with disjunctions and 1-9 rules with existentially quantified variables, 14 rule sets where MFA, DMFA,

#∃ \ #∨	1-9	10-19	20+
1-9	14 / 14 / 14 (15)	3 / 4 / 4 (4)	0 / 0 / 0 (0)
10-99	2 / 2 / 3 (13)	**3 / 9 / 9 (11)**	2 / 2 / 2 (12)
100-999	0 / 2 / 2 (7)	3 / 3 / 3 (3)	2 / 2 / 3 (32)
1000+	1 / 1 / 1 (1)	0 / 0 / 0 (0)	1 / 1 / 1 (5)
31 / 40 / 42 (103)			

Fig. 3. OXFD Evaluation Results #MFA/#DMFA/#RMFA (#total)

and RMFA, respectively. We also marked the cells with the biggest difference of MFA and DMFA in bold. Below the table, we sum up the results from the individual cells.

For OXFD, we see in Figure 3 that DMFA marks exactly 9 more rule sets as terminating than MFA. Also, RMFA only marks 2 more rule sets as terminating than DMFA. To go more into detail, 6 of the rule sets that are DMFA but not MFA result from BioPAX[10] ontologies and the 3 others result from OBO foundry[11] ontologies. We have to note here that the rule sets obtained from the BioPAX ontologies are very similar or to a large extent even identical except for fresh class names introduced during the normalization. In the original dataset, the ontologies only seem to differ in the used ABox-axioms, which are removed during the normalization and translation. Hence, we essentially count the same rule set multiple times so we have to be careful in generalizing our findings here. However, the use of multiple instances of the BioPAX ontologies in OXFD may also hint towards the practical importance of these particular ontologies.

For ORE15 in Figure 4, the difference between MFA and DMFA and between DMFA and RMFA is 18 each, so DMFA is right in the middle between MFA and RMFA in this case. In total, we can see that the vast majority of rule sets is not even RMFA in this dataset. Similar to OXFD, 11 of the 18 rule sets that are DMFA but not MFA are again BioPAX ontologies of which some are essentially identical for the same reasons as for OXFD. Also note that some of the BioPAX ontologies from OXFD reoccur in ORE15. As for OXFD, we have to be careful in generalizing our results here. However, since the ORE15 dataset is bigger and the rule sets that are not MFA but DMFA are more diverse, the numbers for ORE15 seem to give a better reflection of the overall generality of DMFA.

In the bigger picture, for increasing numbers of disjunctive rules and rules with existentially quantified variables, we can see that much fewer rule sets are marked as terminating even for more sophisticated acyclicity notions like RMFA. This trend is most significant for an increased number of rules with existentially quantified variables. Compared to the non-disjunctive rule sets, we

[10] BioPAX - `http://www.biopax.org/`
[11] OBO Foundry - `http://www.obofoundry.org/`

#∃ \ #∨	1-9	10-19	20+
1-9	32 / 32 / 33 (40)	4 / 4 / 4 (4)	4 / 4 / 4 (10)
10-99	33 / 36 / 41 (82)	**8 / 20 / 28 (55)**	2 / 3 / 4 (25)
100-999	3 / 4 / 4 (182)	1 / 2 / 3 (17)	1 / 1 / 3 (73)
1000+	1 / 1 / 1 (7)	0 / 0 / 0 (6)	0 / 0 / 0 (13)
	89 / 107 / 125 (514)		

Fig. 4. ORE15 Evaluation Results #MFA/#DMFA/#RMFA (#total)

also see that the percentage of rule sets that are MFA, DMFA, or RMFA is significantly smaller for the disjunctive part. One may expect that an increased number of disjunctions increases the difference between MFA and DMFA since more rules can potentially be blocked. We can observe this for some cells but we cannot see a significant trend here. Still, it would be interesting to see if such a trend manifests in bigger datasets.

Beside the numbers that we see in the evaluation, we also want to investigate the structure of specific rule sets that are not MFA but DMFA or not DMFA but RMFA, respectively. For rule sets that are not MFA but DMFA, we pick one of the BioPAX ontologies from OXFD as an example and present a slightly simplified version of the structure that produces a cycle in the following example.

Example 3. Consider the following rules:

$$\begin{aligned} \mathit{Evidence}(x) &\to \exists z.\mathit{ConfidenceProp}(x,z) \\ \mathit{ConfidenceProp}(x,y) &\to \mathit{Confidence}(y) \\ \mathit{Confidence}(x) &\to \mathit{Xref}(x) \\ \mathit{Xref}(x) &\to \mathit{Evidence}(x) \vee \mathit{Confidence}(x) \\ \mathit{Evidence}(x) \wedge \mathit{Confidence}(x) &\to \texttt{OWLNothing}(x) \end{aligned}$$

It seems like *Evidence* and *Confidence* are supposed to form a partitioning of *Xref*. Looking at the original ontology[12] this is not quite true but it suffices as an intuition. Every *Evidence* is connected to a *Confidence*, which should itself not be an *Evidence* according to the intuition. For DMFA this works since the forth rule is blocked in this case and will not derive *Evidence* for the existing *Confidence*. For MFA however, every *Confidence* is also an *Evidence* which leads to a cycle. ▲

We can expect similar behavior for similar structures in other rule sets as well. However, the case that two members of a disjoint union are connected by an existential rule may be less common.

[12] `http://krr-nas.cs.ox.ac.uk/ontologies/UID/00007.owl`

For rule sets that are not DMFA but RMFA, we also have a concrete example[13] that we want to discuss in the following.

Example 4. Consider the following rules:

$$\begin{aligned} \mathit{Sibling}(x) &\rightarrow \exists z.\mathit{hasSibling}(z,x) \wedge \mathit{Person}(z) \\ \mathit{Person}(x) \wedge \mathit{hasSibling}(x,y) &\rightarrow \mathit{Sibling}(y) \\ \mathit{Sibling}(x) &\rightarrow \mathit{Person}(x) \\ \mathit{hasSibling}(x,y) &\rightarrow \mathit{hasSibling}(y,x) \end{aligned}$$

The intuitive meaning of the rules from top to bottom is as follows. For every *Sibling* s there is a *Person* that has s as its sibling. If a *Person* has a sibling s' then s' is indeed a *Sibling*. Additionally, every *Sibling* is also a *Person*. Note that the first three rules so far do not introduce cycles. Only with the forth rule, stating the symmetry of the *hasSibling* relation, every *Person* introduced by the first rule is now turned into a *Sibling* by the second rule. This does not only lead to the fact that this rule set is not DMFA but it is even non-terminating w.r.t. the disjunctive skolem chase. On the other hand, the restricted chase is "smart" enough not to introduce a new *Person* in rule one once the symmetry is satisfied and RMFA also catches this. ▲

It is a valuable insight that RMFA cannot benefit from the disjunctions in Example 4. Instead, the generality of RMFA comes solely from the existential rules here. We also see that the rule set from Example 4 is non-terminating w.r.t. the disjunctive skolem chase and we think that this is likely also the case for many of the other rule sets that are not DMFA but RMFA because the structures that may introduce cycles seem to be rather limited by the translation. It could be interesting to investigate other methods of translation and how they affect termination of different chase variants as well as acyclicity notions.

Conclusion

The evaluation results for OXFD and ORE15 show that DMFA can improve upon MFA on rule sets featuring disjunctions not only in theory but also in practice. For these rule sets, we achieve to narrow the gap between MFA and RMFA. Still, we do not know how tight DMFA really is for the disjunctive skolem chase. We know that DMFA cannot surpass its upper bound RMFA but RMFA itself is not sound for the disjunctive skolem chase so it may mark rule sets as terminating that in fact do not terminate w.r.t. the disjunctive skolem chase. On the other hand, note that there also exist rule sets that are not RMFA but terminate w.r.t. the disjunctive skolem chase. However, for rule sets with a higher number of disjunctive and existential rules, we see that only very few of them are in fact RMFA, which still suggests that a significate number of these are non-terminating. Our concrete observations in Example 4 strengthen this conjecture by showing that we encounter rule sets in the given datasets that are RMFA but do in fact not

[13] `http://krr-nas.cs.ox.ac.uk/ontologies/UID/00114.owl`

terminate w.r.t. the disjunctive skolem chase. Still, we cannot be sure about this without evaluating sufficient conditions for non-termination.

Since we can hope for efficient implementations of the disjunctive skolem chase in practice, e.g. using ASP solvers with lazy grounding, our results further motivate the usage of the disjunctive skolem chase in practice and encourage research in this direction. Further research topics include a performance centric evaluation of DMFA, e.g. by investigating the empirical validity of some of our implementation decisions in more detail, as well as other translation methods that may impact different acyclicity notions or even termination of some chase variants. In particular, the influence of equality in the translation may be a good starting point possibly by using an appropriate axiomatization for equality. Our work especially motivates the investigation of cyclicity notions, i.e. sufficient conditions for non-termination, for the disjunctive skolem chase. By that we can hope to tighten the space for rule sets of which we do not know if they are terminating w.r.t. the disjunctive skolem chase or not.

Acknowledgements

I want to thank Prof. Dr. Markus Krötzsch for the possibility of conducting the evaluation of DMFA in the Knowledge-Based Systems Research Group at the TU Dresden as a follow up project of the theoretical considerations. Special thanks to both David Carral, Ph.D. and Prof. Dr. Markus Krötzsch for the supervision of this project in general and for very helpful discussions and remarks in particular.

References

1. Carral, D., Dragoste, I., Krötzsch, M.: Restricted chase (non)termination for existential rules with disjunctions. In: Sierra, C. (ed.) Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017. pp. 922–928. ijcai.org (2017)
2. Carral, D., Krötzsch, M.: Rewriting the description logic ALCHIQ to disjunctive existential rules. In: Bessiere, C. (ed.) Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020. pp. 1777–1783. ijcai.org (2020), scheduled for July 2020, Yokohama, Japan, postponed due to the Corona pandemic.
3. Carral, D., Urbani, J.: Checking chase termination over ontologies of existential rules with equality. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020. pp. 2758–2765. AAAI Press (2020)
4. Gerlach, L.: Don't repeat yourself: Termination of the skolem chase on disjunctive existential rules (2020), (student thesis available at `https://iccl.inf.tu-dresden.de/web/Thema3509`)
5. Parsia, B., Matentzoglu, N., Gonçalves, R.S., Glimm, B., Steigmiller, A.: The OWL reasoner evaluation (ORE) 2015 resources. In: Groth, P., Simperl, E., Gray, A.J.G., Sabou, M., Krötzsch, M., Lécué, F., Flöck, F., Gil, Y. (eds.) The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part II. Lecture Notes in Computer Science, vol. 9982, pp. 159–167 (2016)
6. Parsia, B., Matentzoglu, N., Gonçalves, R.S., Glimm, B., Steigmiller, A.: The OWL reasoner evaluation (ORE) 2015 competition report. J. Autom. Reason. 59(4), 455–482 (2017)