EPiC Series in Computing
Volume 100, 2024, Pages 426–444
Proceedings of 25th Conference on Logic for Programming, Artificial Intelligence and Reasoning

Fuzzy Datalog$^\exists$ over Arbitrary t-Norms

Matthias Lanzinger[3,1], Stefano Sferrazza[1,3], Przemysław A. Wałęga[1], and Georg Gottlob[2,1]

[1] University of Oxford
[2] University of Calabria
[3] TU Wien

Abstract

One of the main challenges in the area of Neuro-Symbolic AI is to perform logical reasoning in the presence of both neural and symbolic data. This requires combining heterogeneous data sources such as knowledge graphs, neural model predictions, structured databases, crowd-sourced data, and many more. To allow for such reasoning, we generalise the standard rule-based language Datalog with existential rules (commonly referred to as tuple-generating dependencies) to the fuzzy setting, by allowing for arbitrary t-norms in the place of classical conjunctions in rule bodies. The resulting formalism allows us to perform reasoning about data associated with degrees of uncertainty while preserving computational complexity results and the applicability of reasoning techniques established for the standard Datalog setting. In particular, we provide fuzzy extensions of Datalog chases which produce fuzzy universal models and we exploit them to show that in important fragments of the language, reasoning has the same complexity as in the classical setting.

1 Introduction

We currently see a rapid growth of Artificial Intelligence and its usage in large-scale applications, such as image and speech recognition, knowledge graphs completion, or recommendation generation. Such systems produce huge amounts of data, whose facts are associated with degrees of truth—expressing the level of confidence in the truth of the datum. Reasoning about such data gives rise to new challenges for data management. Specifically, there is a growing demand for logical reasoning methods capable of integrating precise and uncertain data gathered from heterogeneous sources. Developing efficient approaches for this task would allow us to make a significant step towards a tight integration of symbolic and sub-symbolic AI.

This research direction is currently intensively studied within the areas of Neural-Symbolic AI [23, 24] and Statistical-Relational AI [17] which, in the last years, gave rise to numerous formalisms aiming to integrate various aspects of logical reasoning with neural models. A number of approaches are based on combining logic programming languages with probabilistic and neural predicates; representative examples in this class are DeepProbLog [35], SLASH [40], NeurASP [42], and Generative Datalog [3]. There are also approaches, like Logic Tensor Networks (LTN) [33, 39] or its extension LYRICS [36], which propose to adapt logical semantics to the neural setting by interpreting terms with tensors and connectives with t-norms. On the other hand, there is a long-standing research on fuzzy logics [28] and their applications to logic programming [2, 18, 19, 29, 31, 37] and description logics [34, 41, 10], among others. More recently, the fuzzy setting is studied for complex multi-adjoint [16, 37] and Prolog-derived semantics based on fuzzy similarity of constants and fuzzy unification procedures [29].

Although the recent progress on logical formalisms for reasoning about neural and uncertain data is undeniable, current approaches still do not allow to fully address the grand challenge of integrating neural data with logical reasoning [6, 4]. Indeed, current methods are often of high computational complexity (reasoning is often undecidable and sometimes complexity is not even analysed), require completely new, often exotic, reasoning procedures tailored to the introduced formalisms, or impose significant restrictions on the allowed forms of uncertainty as well as on their interaction within logical reasoning.

We address these difficulties by introducing an extension t-Datalog$^\exists$ of the standard rule-based language Datalog (with existential rules) to the setting where data can be associated with degrees of truth and rules are equipped with a wide range of connectives (interpreted by arbitrary t-norms) operating on these degrees. To illustrate the reasoning capabilities of t-Datalog$^\exists$ consider the example from Figure 1, where the task is to determine a common hypernym of objects presented in images *img1* and *img2*. To this end, we apply the CNN image classifier EfficientNet [35], which provides us with predicted labels for the subject of the image and truth degrees of these predictions. This allows us to produce fuzzy facts with *neural predicates*; the highest truth degrees 0.800 and 0.900 are associated with $\mathit{NeuralLabel}(\mathit{img1}, \mathit{tiger\_shark})$ and $\mathit{NeuralLabel}(\mathit{img2}, \mathit{tench})$, respectively. We also use a lexical database WordNet[1] [38] which contains, among many others, precise facts about hypernyms; for example we obtain facts $\mathit{Hypernym}(\mathit{tiger\_shark}, \mathit{fish})$ and $\mathit{Hypernym}(\mathit{tench}, \mathit{fish})$. To perform reasoning based on a combination of neural data from EfficientNet and precise facts from WordNet we use a t-Datalog$^\exists$ program consisting of the following rules, where conjunctions in rule bodies are replaced with operators corresponding to t-norms:

$$\mathit{NeuralLabel}(x, y) \rightarrow \mathit{Class}(x, y), \tag{r_1}$$

$$\mathit{Class}(x, y) \odot_{\text{Ł}} \mathit{Hypernym}(y, z) \rightarrow \mathit{Class}(x, z), \tag{r_2}$$

$$\mathit{Class}(x, z) \odot_{\mathsf{prod}} \mathit{Class}(y, z) \rightarrow \mathit{CommonClass}(x, y, z). \tag{r_3}$$

Rule (r_1) introduces a binary predicate *Class* which holds for image labels predicted by EfficientNet via *NeuralLabel*, Rule (r_2) exploits knowledge about *Hypernym* from WordNet to assign image objects to classes, and Rule (r_3) derives common classes for pairs of images. Note that t-Datalog$^\exists$ allows for using various t-norms in different rules; in particular, we use Łukasiewicz t-norm $\odot_{\text{Ł}}$ and the product t-norm $\odot_{\mathsf{prod}}$, which operate on truth degrees as follows: $a \odot_{\text{Ł}} b = \max\{0, a + b - 1\}$ and $a \odot_{\mathsf{prod}} b = a \cdot b$. Thus, Rule ($r_2$) allows us to derive $\mathit{Class}(\mathit{img1}, \mathit{fish})$ with truth $0.800 \odot_{\text{Ł}} 1 = 0.800$ and $\mathit{Class}(\mathit{img2}, \mathit{fish})$ with truth $0.900 \odot_{\text{Ł}} 1 = 0.900$ (facts about *Hypernym* are precise and so, they have a truth degree 1). Then, Rule (r_3) derives $\mathit{CommonClass}(img1, img2, fish)$ with truth $0.800 \odot_{\mathsf{prod}} 0.900 = 0.720$. Note that we can derive also $\mathit{CommonClass}(\mathit{img1}, \mathit{img2}, \mathit{tiger\_shark})$ with a low truth degree of 0.016, since EfficientNet misclassified image $img2$ as a *tiger_shark* with truth 0.020.

Our formalism t-Datalog$^\exists$ is, therefore, a natural extension of Datalog$^\exists$ (Datalog extended with existential rules [5], also known as tuple-generating dependencies [7] and studied under the name of Datalog$^\pm$ [14, 25]) to the fuzzy setting, where conjunctions are replaced with t-norms. It is worth emphasising that the class of t-Datalog$^\exists$ programs is very broad; we allow for existential quantification in rule heads (which we did not use in the exemplary program

[1] http://wordnetweb.princeton.edu/

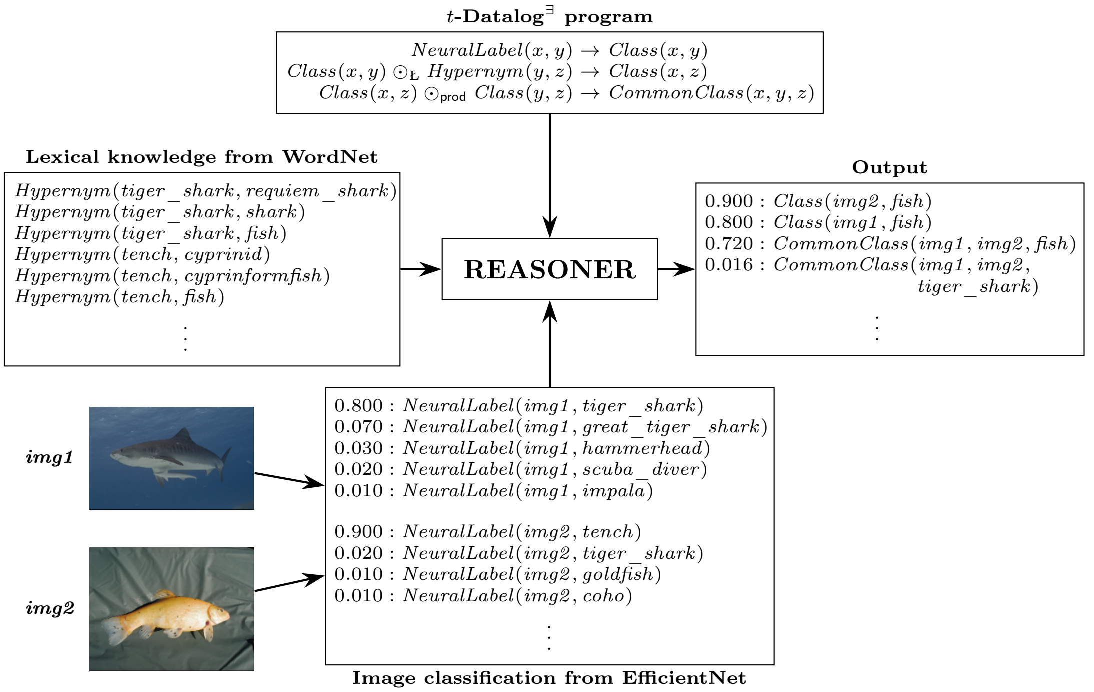

Figure 1: An application of t-Datalog$^\exists$ to derive common classes of objects in input images *img1* and *img2*; reasoning is performed based on image classifications from EfficientNet (with uncertainty degrees) and lexical knowledge from WordNet (precise information).

to simplify presentation), any arity predicates, recursion, and arbitrary t-norms (in contrast to many other approaches, e.g., our previous research tailored specifically to the Łukasiewicz t-norm [31]). In particular, no restriction on the choice of t-norms allows us to model a range of interactions between degrees of truth. This flexibility is also important from the practical perspective, as the choice of t-norms has a significant impact on the performance of a system [21].

The main advantage, distinguishing t-Datalog$^\exists$ from the related formalisms, is that t-Datalog$^\exists$ does not only allow us to perform complex logical reasoning about uncertain data gathered from heterogeneous data sources, but it also allows us to apply well-studied Datalog$^\exists$ reasoning mechanisms. In particular, we show in the paper how to adapt the (semi-oblivious and restricted) chase procedures developed for Datalog$^\exists$ and we prove that complexity of reasoning in various fragments of t-Datalog$^\exists$ is the same as in the corresponding fragments of Datalog$^\exists$. Hence, we obtain a proper extension of Datalog$^\exists$ which allows us to perform complex reasoning about degrees of turth with no negative impact on the computational complexity and with the possibility to use standard chase procedures.

The main contributions of this paper are as follows[2] :

- We introduce t-Datalog$^\exists$ (Section 2) as a fuzzy extension of Datalog$^\exists$ allowing for arbitrary t-norms instead of standard Boolean conjunction in rule bodies. Both syntax and semantics are defined by natural extensions of Datalog$^\exists$ to the fuzzy setting. As a result, we lay

[2] Preliminary ideas for t-Datalog (i.e., without existential quantification) were previously presented as an extended abstract [32].

foundations for fuzzy extensions of the Datalog$^\pm$ family of ontology languages (obtained by imposing various restrictions on Datalog$^\exists$) suitable for neuro-symbolic applications.

- We propose (Section 3.1) a fuzzy version of the chase and we show (Section 3.2) that, similarly as in Datalog$^\exists$, application of a finite fuzzy chase results in a fuzzy universal model, which can be used to decide entailment. We observe, however, that reasoning with fuzzy chases introduces new challenges, which disallows us to directly translate results on termination and complexity from Datalog$^\exists$.
- We introduce (Section 4) a new type of *truth-greedy* fuzzy chases for t-Datalog$^\exists$. We show their relation to standard chases for Datalog$^\exists$, which allows us to exploit termination and complexity results for Datalog$^\exists$. In particular, we show P-completeness for entailment in t-Datalog and in weakly acyclic t-Datalog$^\exists$, matching the complexity of Datalog and weakly acyclic Datalog$^\exists$.
- We introduce (Section 5) an extension of t-Datalog with fuzzy negations and any other unary operators (e.g., threshold operators) interpreted as functions $[0,1] \longrightarrow [0,1]$. We show that adding such operators does not increase complexity of reasoning if the input program is stratifiable (and does not use existential quantification), namely it remains P-complete in data complexity.

2 Datalog$^\exists$ over t-norms

In this section, we introduce syntax and semantics of t-Datalog$^\exists$, as well as present the basic reasoning problem and notation which we will use in the paper.

Signature and domain. We fix a signature σ (i.e. a set of predicate symbols, each of a fixed, but arbitrary arity) and countable sets Dom and Nulls of *object elements* and *nulls*, resepectively, with $\mathsf{Dom} \cap \mathsf{Nulls} = \emptyset$; we let $\mathsf{Dom}_\mathsf{N} = \mathsf{Dom} \cup \mathsf{Nulls}$. We will use GAtoms and $\mathsf{GAtoms}_\mathsf{N}$ for the sets of all ground atoms with constants in Dom and in Dom_N, respectively.

Fuzzy dataset. A *fuzzy dataset* is a partial function $\mathcal{D} : \mathsf{GAtoms} \longrightarrow (0,1]$ assigning real numbers from the interval $(0,1]$—treated as *truth degrees*—to a *finite* number of atoms in GAtoms. Note that a fuzzy dataset does not assign truth degrees to atoms with nulls and that it never assigns 0 to any atom, which is in line with the standard definition of a dataset listing facts which need to hold true (but not mentioning which facts need to be false).

t-norms. A *t-norm* is any commutative, monotone, and associative function of the form $\odot\colon [0,1]\times[0,1] \longrightarrow [0,1]$, with 1 being the identity element [28]. It follows from the definition that if a and b are Boolean (i.e., 0 or 1), then $a \odot b$ coincides with the value of the standard conjunction $a \wedge b$, for any t-norm $\odot$. Thus, t-norms provide a generalisation of the standard conjunction to the fuzzy setting. Commonly used t-norms in fuzzy logics include the *minimum t-norm* (also known as Gödel t-norm) $a \odot_{\min} b = \min\{a,b\}$, *Łukasiewicz t-norm* $a \odot_{Ł} b = \max\{0, a+b-1\}$, and the *real product t-norm* $a \odot_{\mathsf{prod}} b = a \cdot b$. However, t-norms can be significantly more complex, for example, all functions of the form $f_p(a,b) = (a^p + b^p - 1)^{\frac{1}{p}}$, for $p < 0$, are t-norms (part of the Schweizer-Sklar family of t-norms [43]). The following known observation will be particularly important for our results.

Proposition 1. *For every t-norm $\odot$ and all $a, b \in [0,1]$ it holds that $a \odot b \leq a \odot_{\min} b$.*

For an overview of t-norms and their properties, refer to the work of Klement et al. [30].

t-Datalog$^\exists$ programs. A t-Datalog$^\exists$ *program* Π is a finite set of *rules* r, of the form

$$R_1(\mathbf{x_1}) \odot_r \cdots \odot_r R_\ell(\mathbf{x}_\ell) \to \exists \mathbf{z}\, S(\mathbf{x_h}, \mathbf{z}),$$

where $\odot_r$ is any t-norm[3] (each rule can use a different t-norm), $S, R_1, \dots, R_\ell$ are predicate symbols in the signature and $\mathbf{x_1}, \dots, \mathbf{x}_\ell, \mathbf{x_h}, \mathbf{z}$ are (possibly empty) sequences of variables of arities matching the predicate symbols. The left-hand side of the implication $\to$ in a rule r is the *body*, $\text{body}(r)$, and the right-hand side is its *head*, $\text{head}(r)$. Variables in $\mathbf{x_1}, \dots, \mathbf{x}_\ell$ are called *body variables*, those in $\mathbf{x_h}$ *frontier variables*, $\text{fr}(r)$, and those in $\mathbf{z}$ are *existential variables*. We assume that in each rule the set of body variables contains each frontier variable, but does not contain any existential variable. A program with no existential variables is called a t-Datalog program. We use $\text{vars}(\phi)$ to refer to the set of all variables in an expression ϕ (e.g., in a rule or in a rule body). A grounding of a rule r is any function $\rho : \text{vars}(\text{body}(r)) \longrightarrow \mathsf{Dom_N}$ assigning domain constants and nulls to body variables.

Semantics. A *fuzzy interpretation* is a function $\mathcal{I} \colon \mathsf{GAtoms_N} \longrightarrow [0,1]$, assigning truth degrees to ground atoms (also to atoms with nulls). The function is extended to complex expressions ϕ and ϕ' inductively as follows:

$$\begin{aligned} \mathcal{I}(\phi \odot \phi') &= \odot(\mathcal{I}(\phi), \mathcal{I}(\phi')), \\ \mathcal{I}(\phi \to \phi') &= \min\{1, 1 - \mathcal{I}(\phi) + \mathcal{I}(\phi')\}, \\ \mathcal{I}\big(\exists \mathbf{z}\, S(\mathbf{a}, \mathbf{z})\big) &= \sup\left\{\mathcal{I}\big(S(\mathbf{a}, \mathbf{b})\big) \mid \mathbf{b} \in \mathsf{Dom_N}^{|\mathbf{z}|}\right\}, \end{aligned}$$

where $\odot$ is any t-norm and $\mathbf{a}$ is a sequence of constants in $\mathsf{Dom_N}$. Note that we use the same Łukasiewicz semantics for implication $\to$ in all rules.

A rule r is *K-satisfied* by a fuzzy interpretation $\mathcal{I}$, for a rational number $K \in [0,1]$, if $\mathcal{I}(\rho(r)) \geq K$ for every grounding ρ of r. Thus, $\phi \to \phi'$ is K-satisfied if $\mathcal{I}(\rho(\phi')) - \mathcal{I}(\rho(\phi)) \geq K - 1$. In particular, 1-satisfiability requires that $\mathcal{I}(\rho(\phi')) - \mathcal{I}(\rho(\phi)) \geq 0$, that is, the truth degree of the head is at least as large as the truth degree of the body. On the other hand, $\phi \to \phi'$ is trivially 0-satisfied by any $\mathcal{I}$, since $\mathcal{I}(\rho(\phi')) - \mathcal{I}(\rho(\phi)) \geq -1$ is always true. A fuzzy interpretation $\mathcal{I}$ is a *K-fuzzy model* of a program Π if all rules in Π are K-satisfied by $\mathcal{I}$. For a fuzzy dataset $\mathcal{D}$, we let $\mathcal{I}_\mathcal{D}$ be its minimal fuzzy interpretation defined such that $\mathcal{I}_\mathcal{D}(A) = \mathcal{D}(A)$ if A is in the domain of $\mathcal{D}$ and $\mathcal{I}_\mathcal{D}(A) = 0$ otherwise. Then, $\mathcal{I}$ is a *fuzzy model* of $\mathcal{D}$ if $\mathcal{I}(A) \geq \mathcal{I}_\mathcal{D}(A)$ for every ground atom A.

Reasoning and complexity. The basic reasoning problem we consider is K-entailment, for any $K \in [0,1]$, which is to check whether in every K-fuzzy model of a program Π and a fuzzy dataset $\mathcal{D}$, the truth degree of a goal ground atom G is not smaller than a target value c. Hence, the problem is defined as follows:

K-Entailment

Input: A t-Datalog$^\exists$ program Π, a fuzzy dataset $\mathcal{D}$, a ground atom $G \in \mathsf{GAtoms}$, and $c \in [0,1]$.

Output: 'Yes' if and only if $\mathcal{I}(G) \geq c$, for all K-fuzzy models $\mathcal{I}$ of Π and $\mathcal{D}$.

[3] We slightly abuse notation by using the same symbol, $\odot_r$, for a connective in a rule and for a t-norm which interprets this connective.

If the output is 'yes', we say that G is (c, K)-*entailed* by Π and $\mathcal{D}$, written as $(\Pi, \mathcal{D}) \models^c_K G$.

Clearly, checking K-entailment requires applying t-norms and, although computing most of the standard t-norms is easy (e.g., all the t-norms mentioned in this paper), one can introduce computationally-demanding t-norms. For this reason, we will abstract away from the complexity of computing t-norms. In particular, when studying computational complexity of K-Entailment, we will treat the time required to compute t-norms and the memory used to store truth degrees as constant. Moreover, we will focus on the *data complexity* of K-Entailment—which is with respect to the size of $\mathcal{D}$ only—and is particularly important for data-intensive applications, like the one from Figure 1. Furthermore, for the sake of simplicity, we only explicitly analyse atomic entailment in this paper. Recall that entailment of a conjunctive query ϕ can be expressed by adding a rule $\phi \to \mathit{Goal}$ and checking entailment of Goal. Therefore, complexity results in our paper transfer also to conjunctive query entailment.

We can observe that K-Entailment extends the standard notion of entailment in Datalog$^\exists$. In particular, if $c = K = 1$ and the only truth degree assigned by a fuzzy dataset is 1, then K-Entailment coincides with the standard entailment in Datalog$^\exists$, as shown below.

Theorem 2. *Let Π be a t-Datalog$^\exists$ program, $\mathcal{D}$ a fuzzy dataset, and $G \in \mathsf{GAtoms}$. If the only truth degree assigned by $\mathcal{D}$ is 1 (i.e., $\mathcal{D}(A) = 1$ whenever $\mathcal{D}(A)$ is defined), then the following are equivalent:*

1. $(\Pi, \mathcal{D}) \models^1_1 G$,
2. *G is entailed by the Datalog$^\exists$ counterparts Π' and $\mathcal{D}'$ of Π and $\mathcal{D}$ (namely Π' is obtained by replacing t-norms in Π with conjunctions and $\mathcal{D}' = \{A \mid \mathcal{D}(A) = 1\}$).*

Proof. Assume that $(\Pi, \mathcal{D}) \models^1_1 G$ and let $S \subseteq \mathsf{GAtoms_N}$ be a Datalog$^\exists$ model of Π' and $\mathcal{D}'$. We will show that $G \in S$. We define a fuzzy interpretation $\mathcal{I}$ such that for any $A \in \mathsf{GAtoms_N}$ we have $\mathcal{I}(A) = 1$ if $A \in S$, and $\mathcal{I}(A) = 0$ if $A \notin S$. Hence, $\mathcal{I}$ is a fuzzy model of $\mathcal{D}$. We claim that $\mathcal{I}$ is also a 1-fuzzy model of Π. For this, it suffices to show that each rule of Π' is satisfied in S if and only if its counterpart in Π is 1-satisfied in $\mathcal{I}$. This holds since the semantics of any t-norm coincides with the semantics of a conjunction and the semantics of $\to$ in t-Datalog$^\exists$ coincides with its semantics in Datalog$^\exists$, whenever a fuzzy interpretation assigns only Boolean truth degrees to all atoms. As $(\Pi, \mathcal{D}) \models^1_1 G$ and $\mathcal{I}$ is a 1-fuzzy model of Π and $\mathcal{D}$, we obtain that $\mathcal{I}(G) = 1$, and so, $G \in S$, as required.

For the opposite direction assume that Π' and $\mathcal{D}'$ entail G, and let $\mathcal{I}$ be a 1-fuzzy model of Π and $\mathcal{D}$. We construct a 'crisp' version $\mathcal{I}^c$ of $\mathcal{I}$ by setting $\mathcal{I}^c(A) = 1$ if $\mathcal{I}(A)$, and $\mathcal{I}^c(A) = 0$ if $\mathcal{I}(A) < 1$, for any $A \in \mathsf{GAtoms_N}$. Since $\mathcal{D}$ assigns only 1 as a truth degree, $\mathcal{I}^c$ is a fuzzy model of $\mathcal{D}$. We claim that $\mathcal{I}^c$ is also a 1-fuzzy model of Π. Towards a contradiction suppose that some ground rule $\phi \to \phi'$ of Π is 1-satisfied by $\mathcal{I}$, but not by $\mathcal{I}^c$; that is $\mathcal{I}(\phi) \leq \mathcal{I}(\phi')$ and $\mathcal{I}^c(\phi) > \mathcal{I}^c(\phi')$. Since over the Boolean values any t-norm behaves like a conjunction, we obtain that $\mathcal{I}^c(\phi) = 1$ and $\mathcal{I}^c(\phi') = 0$. By the monotonicity of t-norms $\mathcal{I}(\phi) \geq \mathcal{I}^c(\phi)$, so $\mathcal{I}(\phi) = 1$. Moreover $\mathcal{I}(\phi') < 1$ because $\mathcal{I}(\phi') = 1$ would imply that $\mathcal{I}^c(\phi') = 1$. Hence $\mathcal{I}(\phi) \not\leq \mathcal{I}(\phi')$, rising a contradiction. Thus $\mathcal{I}^c$ is a 1-fuzzy model of Π and $\mathcal{D}$. Since $\mathcal{I}^c$ is a 'crisp' model of Π and $\mathcal{D}$, the fact that Π' and $\mathcal{D}'$ entail G implies that $\mathcal{I}^c(G) = 1$. Thus, $\mathcal{I}(G) = 1$, as required. □

3 Chasing t-Datalog$^\exists$

In this section, we define fuzzy chases and fuzzy universal models. We will show that each finite fuzzy chase produces a fuzzy universal model, which can be used to check entailment.

As described in the previous section, the notion of satisfaction in t-Datalog$^\exists$ is parameterised with $K \in [0,1]$ (and so are also parameterised notions of a model and entailment), which determines how much larger the truth degree of a head of rule needs to be than the truth degree of the body so that we treat the rule as satisfied. It turns out, however, that the specific choice of the value for K does not impact our technical results. Therefore, for the sake of simplification, we will assume in the rest of the paper that $K = 1$ and we will not mention K; for example instead of K-ENTAILMENT and $\models^c_K$ we will simply refer to ENTAILMENT and $\models^c$, respectively. In Section 6 we will briefly discuss how our results generalise to arbitrary values of K.

3.1 Fuzzy Triggers and Chases

We start by defining notions of fuzzy semi-oblivious and restricted chases in t-Datalog$^\exists$, by lifting the notions used in standard Datalog$^\exists$ [12]. Our definitions are based on a fuzzy counterpart of a trigger, defined next.

A *fuzzy trigger* is a pair (r, ρ), where r is a rule and ρ is a grounding of r. The *result of applying a trigger* (r, ρ) to a fuzzy interpretation $\mathcal{I}$ is a fuzzy interpretation $\mathcal{I}'$ obtained by updating the truth degree of the head of $\rho(r)$ (with existential variables replaced by nulls) to the value $\mathcal{I}(\rho(\text{body}(r)))$, that is, to the truth degree of the body. Formally, we let the *head* $\text{H}(r, \rho)$ *of a fuzzy trigger* (r, ρ) be the ground atom obtained from the head of $\rho(r)$ by deleting existential quantifier and replacing each existential variable x with a null $\mathbf{N}^x_{r,\rho_{|\text{fr}(r)}} \in \mathsf{Nulls}$, where the null is determined by x, r, and the restriction of ρ to frontier variables $\text{fr}(r)$. To simplify notation and make it more intuitive, in what follows we will write $\mathsf{tr}_\mathcal{I}(r, \rho)$ instead of $\mathcal{I}(\rho(\text{body}(r)))$, for the truth degree corresponding to a trigger application. Hence, we formally define the result of applying (r, ρ) to $\mathcal{I}$ as the following fuzzy interpretation $\mathcal{I}'$:

$$\mathcal{I}'(A) = \begin{cases} \mathsf{tr}_\mathcal{I}(r, \rho), & \text{if } A = \text{H}(r, \rho), \\ \mathcal{I}(A), & \text{if } A \in \mathsf{GAtoms}_\mathsf{N} \setminus \{\text{H}(r, \rho)\}. \end{cases}$$

Example 3. Let r be the following rule r

$$\mathit{NeuralLabel}(x, y) \odot_Ł \mathit{NeuralLabel}(u, w) \rightarrow \exists z\, \mathit{CommonClass}(x, u, z)$$

and let ρ be its grounding with $x \mapsto \mathit{img1}$, $y \mapsto \mathit{tiger\_shark}$, $u \mapsto \mathit{img2}$, and $w \mapsto \mathit{tench}$, so the head $\text{H}(r, \rho)$ of the trigger (r, ρ) is $\mathit{CommonClass}(\mathit{img1}, \mathit{img2}, \mathbf{N}^z_{r,\rho_{|\text{fr}(r)}})$. To illustrate application of (r, ρ) assume that $\mathcal{I}$ assigns $\mathit{NeuralLabel}(\mathit{img1}, \mathit{tiger\_shark}) \mapsto 0.8$ and $\mathit{NeuralLabel}(\mathit{img2}, \mathit{tench}) \mapsto 0.9$. Hence $\mathsf{tr}_\mathcal{I}(r, \rho) = 0.8 \odot_Ł 0.9 = 0.7$, and so, the result of applying (r, ρ) to $\mathcal{I}$ is a fuzzy interpretation $\mathcal{I}'$ with $\mathit{CommonClass}(\mathit{img1}, \mathit{img2}, \mathbf{N}^z_{r,\rho_{|\text{fr}(r)}}) \mapsto 0.7$.

Depending on the type of allowed triggers we will obtain different types of chase procedures. In particular, the semi-oblivious chase will allow for so-active triggers and the restrictive chase for r-active triggers only. We say that:

- a trigger (r, ρ) is so-*active* in a fuzzy interpretation $\mathcal{I}$ if $\mathsf{tr}_\mathcal{I}(r, \rho) > \mathcal{I}(\text{H}(r, \rho))$, that is, an application of the trigger will increase the current truth degree of $\text{H}(r, \rho)$,
- a trigger (r, ρ) is r-*active* in a fuzzy interpretation $\mathcal{I}$ if $\mathsf{tr}_\mathcal{I}(r, \rho) > \mathcal{I}(\text{H}')$ for all H' obtained by replacing nulls in $\text{H}(r, \rho)$ with elements from Dom_N.

Note that each r-active trigger is also so-active, because $\mathsf{tr}_\mathcal{I}(r, \rho) > \mathcal{I}(\text{H}')$ for all H' described above, implies, in particular, that $\mathsf{tr}_\mathcal{I}(r, \rho) > \mathcal{I}(\text{H}(r, \rho))$.

A *semi-oblivious chase* (so-chase) for a program Π and a fuzzy dataset $\mathcal{D}$ is a (finite or infinite) sequence of triggers $(r_1, \rho_1), (r_2, \rho_2), \dots$ with rules r_i in Π such that there exists a sequence of fuzzy interpretations $\mathcal{I}_0, \mathcal{I}_1, \mathcal{I}_2, \dots$ with the following properties:

(i) $\mathcal{I}_0 = \mathcal{I}_{\mathcal{D}}$,
(ii) each (r_i, ρ_i) is so-active in $\mathcal{I}_{i-1}$,
(iii) each $\mathcal{I}_i$ (with $i > 0$) is the result of applying (r_i, ρ_i) to $\mathcal{I}_{i-1}$,
(iv) for each $\mathcal{I}_i$ and every so-active trigger in $\mathcal{I}_i$, there exists finite $j > i$ such that this trigger is not so-active in $\mathcal{I}_j$ (the *fairness* condition).

Observe that each fuzzy chase has a unique sequence $\mathcal{I}_0, \mathcal{I}_1, \mathcal{I}_2, \dots$ of fuzzy interpretations, which we will call *corresponding* to the chase; we will also call $\mathcal{I}_i$ the *ith interpretation in the chase*. A *restricted chase* (r-chase) is defined analogously, by using the notion of a r-active trigger instead of an so-active trigger. Note that both types of chases can be either finite or infinite. If a chase is finite, we call the last interpretation $\mathcal{I}_n$ the *result of applying the chase*.

Example 4. Let $\Pi = \{r\}$ for the rule r from Example 3 and let $\mathcal{D}$ be the fuzzy dataset corresponding to $\mathcal{I}$ from Example 3 (i.e., $\mathcal{I}_{\mathcal{D}} = \mathcal{I}$). There are four active triggers in $\mathcal{I}_{\mathcal{D}}$ (as there are four ways we can assign *img1* and *img2* to x and y). Each order of their applications gives rise to a different chase. For example we can obtain a chase, say $(r, \rho_1), (r, \rho_2), (r, \rho_3), (r, \rho_4)$, with corresponding interpretations $\mathcal{I}_0, \mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3, \mathcal{I}_4$ such that

$\mathcal{I}_1$ assigns $\mathit{CommonClass}(\mathit{img1}, \mathit{img2}, \mathbf{N}^z_{r,\rho_1 | \mathrm{fr}(r)}) \mapsto 0.7$ (since $0.8 \odot_{Ł} 0.9 = 0.7$),
$\mathcal{I}_2$ assigns $\mathit{CommonClass}(\mathit{img2}, \mathit{img1}, \mathbf{N}^z_{r,\rho_2 | \mathrm{fr}(r)}) \mapsto 0.7$ (since $0.9 \odot_{Ł} 0.8 = 0.7$),
$\mathcal{I}_3$ assigns $\mathit{CommonClass}(\mathit{img1}, \mathit{img1}, \mathbf{N}^z_{r,\rho_3 | \mathrm{fr}(r)}) \mapsto 0.6$ (since $0.8 \odot_{Ł} 0.8 = 0.6$),
$\mathcal{I}_4$ assigns $\mathit{CommonClass}(\mathit{img2}, \mathit{img2}, \mathbf{N}^z_{r,\rho_4 | \mathrm{fr}(r)}) \mapsto 0.8$ (since $0.9 \odot_{Ł} 0.9 = 0.8$).

Note that $(r, \rho_1), (r, \rho_2), (r, \rho_3), (r, \rho_4)$ is both an so-chase and an r-chase for Π and $\mathcal{D}$. It is still an so-chase if we add $\mathit{CommonClass}(\mathit{img1}, \mathit{img2}, \mathit{fish}) \mapsto 0.8$ to $\mathcal{D}$, but it is not an r-chase any more because $\mathsf{tr}_{\mathcal{I}_0}(r, \rho_1) = 0.7 \not\geq 0.8 = \mathcal{I}_0(\mathit{CommonClass}(\mathit{img1}, \mathit{img2}, \mathit{fish}))$, and so, (r, ρ_1) is not r-active in $\mathcal{I}_0$.

3.2 Fuzzy Universal Models

The core property of chases in Datalog$^{\exists}$, which makes them a crucial tool for checking entailment, is that each finite chase results in a *universal model* that represents (modulo homomorphisms) all models of a given program and dataset. In this section, we show that an analogous result can be provided for fuzzy chases in t-Datalog$^{\exists}$. We start by introducing a notion of a homomorphism which is tailored to our fuzzy setting. As defined below, a homomorphism is a function $h : \mathsf{Dom}_{\mathsf{N}} \longrightarrow \mathsf{Dom}_{\mathsf{N}}$, but we will often use its extension to atoms and rule bodies; for example we will write $h(A)$ to refer to the atom A with constants replaced according to h.

Definition 5. A *non-decreasing homomorphism* from a fuzzy interpretation $\mathcal{I}$ to a fuzzy interpretation $\mathcal{I}'$ is any function $h : \mathsf{Dom}_{\mathsf{N}} \longrightarrow \mathsf{Dom}_{\mathsf{N}}$ such that:

(i) $h(a) = a$, for every $a \in \mathsf{Dom}$, and
(ii) $\mathcal{I}(A) \leq \mathcal{I}'(h(A))$, for every $A \in \mathsf{GAtoms}_{\mathsf{N}}$.

We will write $\mathcal{I} \preccurlyeq \mathcal{I}'$ if there is a non-decreasing homomorphism from $\mathcal{I}$ to $\mathcal{I}'$.

Similarly as in Datalog$^{\exists}$, we define a universal model of Π and $\mathcal{D}$ as a model which can be homomorphically mapped to any model of Π and $\mathcal{D}$. However, instead of the standard homomorphism, we use the above-defined non-decreasing homomorphism.

Definition 6. A *fuzzy universal model* of a t-Datalog$^\exists$ program Π and a fuzzy dataset $\mathcal{D}$ is any fuzzy model $\mathcal{I}$ of Π and $\mathcal{D}$ such that $\mathcal{I} \preccurlyeq \mathcal{I}'$ for every fuzzy model $\mathcal{I}'$ of Π and $\mathcal{D}$.

In general, a fuzzy universal model may not be unique. However, if a program does not have existential variables, we can show that there exists a unique fuzzy universal model. Note that this is analogous to the standard, non-fuzzy setting.

Theorem 7. *Each pair of a t-Datalog program Π and a fuzzy dataset $\mathcal{D}$ has a unique fuzzy universal model.*

Proof. Consider the set S of all fuzzy models of Π and $\mathcal{D}$. We claim that the fuzzy interpretation $\mathcal{I}$ with $\mathcal{I}(A) = \inf\{\mathcal{I}'(A) \mid \mathcal{I}' \in S\}$, for each $A \in \mathsf{GAtoms_N}$, is the unique fuzzy universal model of Π and $\mathcal{D}$. By the construction, $\mathcal{I}$ is a fuzzy model of $\mathcal{D}$. To show that it is a fuzzy model of Π, we observe that for any rule $\phi \to \phi'$ in Π, any of its groundings ρ, and any $\mathcal{I}' \in S$, we have $\mathcal{I}'(\rho(\phi)) \leq \mathcal{I}'(\rho(\phi'))$. Thus, we have also $\mathcal{I}(\rho(\phi)) \leq \mathcal{I}(\rho(\phi'))$. Consequently, $\mathcal{I}$ is a fuzzy model of Π and $\mathcal{D}$ (i.e, $\mathcal{I} \in S$). To prove that $\mathcal{I}$ is a fuzzy universal model, we need to show that $\mathcal{I} \preccurlyeq \mathcal{I}'$, for any $\mathcal{I}' \in S$. This, however, is witnessed by simply letting h be the identity function on $\mathsf{Dom_N}$. Indeed, such h satisfies both Conditions (i) and (ii) from Definition 5. Finally, we observe that, by the construction, no $\mathcal{I}' \in S \setminus \{\mathcal{I}\}$ satisfies $\mathcal{I}' \preccurlyeq \mathcal{I}$, so no $\mathcal{I}' \in S$ different from $\mathcal{I}$ can be a fuzzy universal model of Π and $\mathcal{D}$. □

We observe that each fuzzy universal model can be used to check entailment, which follows directly from our definitions.

Proposition 8. *Let $\mathcal{I}$ be a fuzzy universal model of a t-Datalog$^\exists$ program Π and a fuzzy dataset $\mathcal{D}$, let $c \in [0,1]$, and let $G \in \mathsf{GAtoms}$. Then $\mathcal{I}(G) \geq c$ if and only if $(\Pi, \mathcal{D}) \models^c G$.*

The crucial property connecting fuzzy chases and universal models, is that every finite fuzzy chase (semi-oblivious or restricted) needs to result in a fuzzy universal model.

Theorem 9. *Let $\star \in \{\mathsf{so}, \mathsf{r}\}$. The result of applying a $\star$-chase for a t-Datalog$^\exists$ program Π and a fuzzy dataset $\mathcal{D}$ is a fuzzy universal model of Π and $\mathcal{D}$.*

Proof. Consider first $\star = \mathsf{so}$. Let $(r_1, \rho_1), (r_2, \rho_2), \dots, (r_n, \rho_n)$ be a finite so-chase for Π and $\mathcal{D}$, and let $\mathcal{I}_\mathcal{D} = \mathcal{I}_0, \mathcal{I}_1, \dots \mathcal{I}_n$ be corresponding fuzzy interpretations. To show that $\mathcal{I}_n$ is a fuzzy universal model for Π and $\mathcal{D}$, we fix an arbitrary fuzzy model $\mathcal{I}'$ of Π and $\mathcal{D}$; we will show inductively on i that $\mathcal{I}_i \preccurlyeq \mathcal{I}'$.

For the basis of induction, we need to show that there is a non-decreasing fuzzy homomorphism from $\mathcal{I}_\mathcal{D}$ to $\mathcal{I}'$. Since $\mathcal{I}'$ is a fuzzy model of $\mathcal{D}$, we have $\mathcal{I}_\mathcal{D}(A) \leq \mathcal{I}'(A)$ for every $A \in \mathsf{GAtoms_N}$. Thus the identity function on $\mathsf{Dom_N}$ is a non-decreasing fuzzy homomorphism witnessing $\mathcal{I}_\mathcal{D} \preccurlyeq \mathcal{I}'$.

For the inductive step assume that h is a non-decreasing fuzzy homomorphism from $\mathcal{I}_{i-1}$ to $\mathcal{I}'$; we will show how to construct a non-decreasing fuzzy homomorphism h' from $\mathcal{I}_i$ to $\mathcal{I}'$. Let us write *body* as a shorthand for $\rho_i(\mathrm{body}(r_i))$. Since h is a non-decreasing fuzzy homomorphism from $\mathcal{I}_{i-1}$ to $\mathcal{I}'$ and t-norms are monotone, $\mathcal{I}_{i-1}(\mathit{body}) \leq \mathcal{I}'(h(\mathit{body}))$. As $\mathcal{I}'$ is a fuzzy model of Π, it needs to satisfy r_i. Thus, there exists a ground atom H, obtained by replacing each existential variable x in $\mathrm{head}(r_i)$ with some $a_x \in \mathsf{Dom_N}$, such that $\mathcal{I}'(h(\mathit{body})) \leq \mathcal{I}'(H)$. We use this H to define h' as follows:

$$h'(a) = \begin{cases} a_x, & \text{if } a = \mathbf{N}^x_{r_i, \rho_{i|\,\mathrm{fr}(r_i)}} \text{ for some } x \text{ (i.e., } a \text{ is a null in } \mathrm{H}(r_i, \rho_i)), \\ h(a), & \text{for all other } a \in \mathsf{Dom_N}. \end{cases}$$

It remains to show that h' is a non-decreasing fuzzy homomorphism from $\mathcal{I}_i$ to $\mathcal{I}'$. Condition (i) of Definition 5 holds by the construction of h'. For Condition (ii) it suffices to show that

$\mathcal{I}_i(\mathrm{H}(r_i,\rho_i)) \leq \mathcal{I}'(h'(\mathrm{H}(r_i,\rho_i)))$, as for all $A \in \mathsf{GAtoms_N}$ other than $\mathrm{H}(r_i,\rho_i)$, the inequality holds by the inductive assumption. We observe that the following hold:

$$\mathcal{I}_i(\mathrm{H}(r_i,\rho_i)) = \mathcal{I}_{i-1}(\mathit{body}) \leq \mathcal{I}'(h(\mathit{body})) \leq \mathcal{I}'(H) = \mathcal{I}'(h'(\mathrm{H}(r_i,\rho_i))).$$

The first equality holds by the definition of a trigger application, the next two inequalities are already shown, and the last equality holds by the construction of h'. Thus h' witnesses $\mathcal{I}_i \preccurlyeq \mathcal{I}'$.

Finally, we observe that each finite r-chase is a prefix of some so-chase, so the argumentation above also proves the theorem for $\star = \mathsf{r}$. □

Theorem 9, together with Proposition 8, provides us with a mechanism for checking entailment, which aligns directly with the standard methods for deciding entailment in Datalog$^\exists$. There are however two main difficulties that need to be addressed. First, as in the classical setting, fuzzy chases are not always finite. Second, unlike in the classical setting, a fuzzy chase can update the truth degree of the same ground atom *multiple times*, as illustrated in the following example.

Example 10. Consider a program Π with Rules (r_1)–(r_3), and a fuzzy dataset $\mathcal{D}$ assigning $\mathit{NeuralLabel}(\mathit{img}, c_1) \mapsto 0.9$, $\mathit{Class}(\mathit{img}, c_1) \mapsto 0.6$, and $\mathit{Hypernym}(c_1, c_2) \mapsto 1$. Let the first trigger in a fuzzy chase for Π and $\mathcal{D}$ be $(r_2, \{x \mapsto \mathit{img}, y \mapsto c_1, z \mapsto c_2\})$, which results in $\mathit{Class}(\mathit{img}, c_2) \mapsto 0.6$. Let the second trigger be $(r_1, \{x \mapsto \mathit{img}, y \mapsto c_1\})$, which assigns $\mathit{Class}(\mathit{img}, c_1) \mapsto 0.9$. This, however, makes $(r_2, \{x \mapsto \mathit{img}, y \mapsto c_1, z \mapsto c_2\})$ active again and its application now yields $\mathit{Class}(\mathit{img}, c_2) \mapsto 0.9$. Note that it holds in both so- and r-chases.

The example above shows that even in the case of a t-Datalog program (with no existential quantification), a trigger can be *reactivated*, and so, the same trigger can occur multiple times in a fuzzy chase. This, in turn, may potentially lead to exponentially long or even infinite chases, which never happens in standard Datalog. Clearly, if existential quantification is present in a t-Datalog$^\exists$ program, reasoning with chases becomes even more challenging. In the next section, we will show how to overcome these difficulties. In particular, we will show in which cases the finiteness of a chase is guaranteed, and so, chasing can be used as a decision procedure for entailment checking.

4 Truth-Greedy Chases

As we have shown, reasoning is conceptually harder than in the classical setting, as a fuzzy chase may modify multiple times a truth degree of the same atom. To address this difficulty, we will introduce truth-greedy chases, which allow us to overcome the above-mentioned issue. This, in particular, will allow us to show that reasoning in weakly acyclic programs in t-Datalog$^\exists$ is tractable for data complexity, and so, no harder than in Datalog$^\exists$.

Let $\star \in \{\mathsf{so}, \mathsf{r}\}$. We say that a $\star$-chase $(r_1, \rho_1), (r_2, \rho_2), \dots$ with corresponding fuzzy interpretations $\mathcal{I}_0, \mathcal{I}_1, \mathcal{I}_2, \dots$ is *truth-greedy* if each of its triggers (r_i, ρ_i) has a maximal truth degree among all $\star$-active triggers in $\mathcal{I}_{i-1}$, that is, there is no $\star$-active trigger (r', ρ') such that $\mathsf{tr}_{\mathcal{I}}(r_i, \rho_i) < \mathsf{tr}_{\mathcal{I}}(r', \rho')$. In other words, in the ith step of a truth-greedy chase, we need to apply a trigger whose results assigns a maximal truth degree among active triggers.

As we show next, truth degrees of triggers in a truth-greedy chase are non-increasing. Note that this does not follow directly from the definition of a truth-greedy trigger, because application of a trigger can activate triggers which were not active before. In general, such triggers may have higher truth degrees. However, we will show that due to monotonicity of t-norms none of the newly activated triggers can have a truth degree higher than previously applied triggers.

Proposition 11. *Let* $\star \in \{\mathsf{so}, \mathsf{r}\}$ *and let* $(r_1, \rho_1), (r_2, \rho_2), \dots$ *be a truth-greedy* $\star$*-chase with corresponding interpretations* $\mathcal{I}_0, \mathcal{I}_1, \mathcal{I}_2, \dots$*. For all* $i \geq 1$ *it holds that* $\mathsf{tr}_{\mathcal{I}_{i-1}}(r_i, \rho_i) \geq \mathsf{tr}_{\mathcal{I}_i}(r_{i+1}, \rho_{i+1})$*.*

Proof. Suppose towards a contradiction that $\mathsf{tr}_{\mathcal{I}_{i-1}}(r_i, \rho_i) < \mathsf{tr}_{\mathcal{I}_i}(r_{i+1}, \rho_{i+1})$, for some $i \geq 1$. Since the chase is truth-greedy, $\mathsf{tr}_{\mathcal{I}_{i-1}}(r_{i+1}, \rho_{i+1}) \leq \mathsf{tr}_{\mathcal{I}_{i-1}}(r_i, \rho_i)$. Therefore $\mathsf{tr}_{\mathcal{I}_{i-1}}(r_{i+1}, \rho_{i+1}) < \mathsf{tr}_{\mathcal{I}_i}(r_{i+1}, \rho_{i+1})$, or equivalently $\mathcal{I}_{i-1}(\rho_{i+1}(\text{body}(r_{i+1}))) < \mathcal{I}_i(\rho_{i+1}(\text{body}(r_{i+1})))$. By the definition of a trigger application, the only difference between $\mathcal{I}_{i-1}$ and $\mathcal{I}_i$ is that the truth degree of $\text{H}(r_i, \rho_i)$ is strictly increased in $\mathcal{I}_i$ to the value of $\mathcal{I}_{i-1}(\rho_i(\text{body}(r_i)))$. Therefore, by the fact that $\mathcal{I}_{i-1}(\rho_{i+1}(\text{body}(r_{i+1}))) < \mathcal{I}_i(\rho_{i+1}(\text{body}(r_{i+1})))$, the body $\rho_{i+1}(\text{body}(r_{i+1}))$ needs to mention the atom $\text{H}(r_i, \rho_i)$, and so, by Proposition 1, the truth degree of this body in $\mathcal{I}_i$ is no larger than the truth degree of $\text{H}(r_i, \rho_i)$, that is, $\mathcal{I}_i(\rho_{i+1}(\text{body}(r_{i+1}))) \leq \mathcal{I}_{i-1}(\rho_i(\text{body}(r_i)))$. This, however, contradicts the assumption $\mathsf{tr}_{\mathcal{I}_{i-1}}(r_i, \rho_i) < \mathsf{tr}_{\mathcal{I}_i}(r_{i+1}, \rho_{i+1})$. □

As a consequence of Proposition 11 and the definitions of so- and r-active triggers, we obtain that a truth-greedy chase cannot mention two triggers with the same head.

Corollary 12. *Let* $\star \in \{\mathsf{so}, \mathsf{r}\}$ *and let* $(r_1, \rho_1), (r_2, \rho_2), \dots$ *be a truth-greedy* $\star$*-chase. It holds that* $\text{H}(r_i, \rho_i) \neq \text{H}(r_j, \rho_j)$ *whenever* $i \neq j$*.*

To exploit truth-greedy chases for efficient reasoning, we will relate fuzzy chases in t-Datalog$^\exists$ to standard chases in Datalog$^\exists$. In formal terms, we will relate a fuzzy chase of a t-Datalog$^\exists$ program Π and a fuzzy dataset $\mathcal{D}$ to a fuzzy chase of Π and $\mathcal{D}^{\mathsf{crisp}}$, where $\mathcal{D}^{\mathsf{crisp}}$ is a 'crispified' version of $\mathcal{D}$ obtained by setting $\mathcal{D}^{\mathsf{crisp}}(A) = 1$ whenever $\mathcal{D}(A)$ is defined (recall that, by the definition, it implies that $\mathcal{D}(A) > 0$). Note that a fuzzy chase for a 'crispified' dataset, corresponds to a standard chase in Datalog$^\exists$. As we show next, if the application of standard chase procedures to a crispified dataset $\mathcal{D}^{\mathsf{crisp}}$ always terminates (e.g., when a program has no existential variables, it is non-recursive, or weakly-acyclic), then for each fuzzy chase s there is a standard chase s^{crisp} such that s assigns truth degrees to no more atoms than s^{crisp}. Formally, for a chase s with corresponding interpretations $\mathcal{I}_0, \mathcal{I}_1, \dots$ we let $\mathsf{atoms}(s)$ be the set of all $A \in \mathsf{GAtoms}_{\mathsf{N}}$ such that $\mathcal{I}_i(A) \neq 0$, for some i. Hence, our result claims that $\mathsf{atoms}(s) \subseteq \mathsf{atoms}(s^{\mathsf{crisp}})$, shown below.

Lemma 13. *Let* $\star \in \{\mathsf{so}, \mathsf{r}\}$*, let* Π *be a* t*-Datalog*$^\exists$ *program, and let* $\mathcal{D}$ *be a fuzzy dataset. Assume that every sequence of* $\star$*-active trigger applications to* Π *and* $\mathcal{D}^{\mathsf{crisp}}$ *is finite. Then, for every* $\star$*-chase* s *of* Π *and* $\mathcal{D}$ *there exists a* $\star$*-chase* s^{crisp} *of* Π *and* $\mathcal{D}^{\mathsf{crisp}}$ *with* $\mathsf{atoms}(s) \subseteq \mathsf{atoms}(s^{\mathsf{crisp}})$*.*

Proof. Let s be a $\star$-chase $(r_1, \rho_1), (r_2, \rho_2), \dots$ with corresponding interpretations $\mathcal{I}_0, \mathcal{I}_1, \mathcal{I}_2, \dots$. We will show inductively on $i \geq 1$, that for each prefix $s_i = (r_1, \rho_1), \dots, (r_i, \rho_i)$ of s, there exists a finite sequence $s'_i = (r'_1, \rho'_1), \dots, (r'_j, \rho'_j)$ of $j \leq i$ $\star$-active trigger applications to Π and $\mathcal{D}^{\mathsf{crisp}}$ such that $\mathsf{atoms}(s_i) = \mathsf{atoms}(s'_i)$. By the assumption that every sequence of $\star$-active trigger applications to Π and $\mathcal{D}^{\mathsf{crisp}}$ is finite, this implies the existence of the required s^{crisp}.

In the basis of the induction we have $s_1 = (r_1, \rho_1)$, hence (r_1, ρ_1) is $\star$-active in $\mathcal{I}_{\mathcal{D}}$, and so, $\rho_1(\text{body}(r_1)) > 0$. Thus, for each A mentioned in $\rho_1(\text{body}(r_1))$ we have $\mathcal{D}(A) > 0$ and therefore $\mathcal{D}^{\mathsf{crisp}}(A) = 1$. If $\mathcal{D}^{\mathsf{crisp}}(\text{H}(r_1, \rho_1)) \neq 1$, then (r_1, ρ_1) is $\star$-active in $\mathcal{I}_{\mathcal{D}^{\mathsf{crisp}}}$ and we let $s'_1 = (r_1, \rho_1)$; otherwise we let s'_1 be the empty sequence. In both cases $\mathsf{atoms}(s_1) = \mathsf{atoms}(s'_1)$, as required.

In the inductive step we fix $i \in \mathbb{N}$ and assume that the claim is witnessed for $s_i = (r_1, \rho_1), \dots, (r_i, \rho_i)$ by $s'_i = (r'_1, \rho'_1), \dots, (r'_j, \rho'_j)$. Since (r_{i+1}, ρ_{i+1}) is $\star$-active in $\mathcal{I}_i$, for each atom A in the body $\rho_{i+1}(\text{body}(r_{i+1}))$, we have $\mathcal{I}_i(A) > 0$, and so, $A \in \mathsf{atoms}(s_i)$. Thus, by the inductive assumption, $A \in \mathsf{atoms}(s'_i)$, that is, $\mathcal{I}'_j(A) = 1$ for $\mathcal{I}'_0, \dots, \mathcal{I}'_j$ being interpretations corresponding to s'_i. Similarly as in the basis, if $\mathcal{I}'_j(\text{H}(r_{i+1}, \rho_{i+1})) \neq 1$, then (r_{i+1}, ρ_{i+1}) is $\star$-active in $\mathcal{I}'_j$ and we let $s'_{i+1} = (r'_1, \rho'_1), \dots, (r'_j, \rho'_j), (r_{i+1}, \rho_{i+1})$; otherwise we let $s'_{i+1} = s'_i$. In both cases $\mathsf{atoms}(s_{i+1}) = \mathsf{atoms}(s'_{i+1})$. □

It is worth observing that checking the assumption of Lemma 13, that is, if all $\star$-trigger applications to Π and $\mathcal{D}^{\mathsf{crisp}}$ are finite, we can use results established for Datalog$^\exists$. For example, it is known that for a fixed program and a dataset, all semi-oblivious chases are finite if there exists some finite semi-oblivious chase [27]. Since semi-oblivious chases in Datalog$^\exists$ coincide with so-chases in t-Datalog$^\exists$ for Π and $\mathcal{D}^{\mathsf{crisp}}$, checking if the assumption of Lemma 13 is satisfied, reduces to checking if some so-chase of Π and $\mathcal{D}^{\mathsf{crisp}}$ is finite.

Note that Lemma 13 and Corollary 12 allow us to determine for which pairs of a program Π and a dataset $\mathcal{D}$, the truth-greedy chase is guaranteed to terminate. Indeed, this is the case whenever the standard chase applied to $\mathcal{D}^{\mathsf{crisp}}$ is guaranteed to terminate. Furthermore, if we know the lengths of standard chases, we can bound the length of fuzzy truth-greedy chases.

For example, let us consider *weakly acyclic* programs (a syntactic restriction that inhibits the role of nulls in recursion [20]), which constitutes one of the most prominent fragments of Datalog$^\exists$ with terminating restricted chase [20]. Since standard chases for weakly acyclic programs are known to be polynomially long in the size of a dataset, we can show that truth-greedy chases are also polynomially long. Moreover, we can show that entailment in weakly acyclic t-Datalog$^\exists$ programs is P-complete for data complexity, that is, of the same computational complexity as in the non-fuzzy weakly acyclic Datalog$^\exists$ programs.

Theorem 14. *ENTAILMENT for weakly acyclic t-Datalog$^\exists$ programs is* P*-complete in data complexity.*

Proof. The lower bound follows from P-hardness of entailment in weakly-acyclic Datalog$^\exists$ [15] and Theorem 2. For the upper bound assume that we want to check if $(\Pi, \mathcal{D}) \models^c G$, for a weakly acyclic t-Datalog$^\exists$ program Π, fuzzy dataset $\mathcal{D}$, $G \in \mathsf{GAtoms_N}$, and $c \in [0,1]$. By Proposition 8 and Theorem 9 it suffices to construct a (finite) r-chase for Π and $\mathcal{D}$, and to check if $\mathcal{I}(G) \geq c$ in the resulting fuzzy interpretation $\mathcal{I}$ of this chase. Since Π is weakly-acyclic, by the result for Datalog$^\exists$ established by Fagin et al. [20], every sequence of r-active trigger applications to Π and $\mathcal{D}^{\mathsf{crisp}}$ is of polynomial length in the size of $\mathcal{D}^{\mathsf{crisp}}$. Hence, by Lemma 13, $|\mathsf{atoms}(s)|$ is polynomial, for every r-chase s of Π and $\mathcal{D}$. Note that this does not mean that all r-chases have polynomial lengths. However, by Corollary 12, we obtain that all truth-greedy r-chases have polynomial lengths.

It remains to argue that constructing a truth-greedy r-chase of Π and $\mathcal{D}$ is feasible in polynomial time (in the size of $\mathcal{D}$). To construct the chase, in every step we compute all r-active triggers (r, ρ), choose one of the triggers with maximal truth degree of $\rho(\text{body}(r))$, and apply it to construct a next fuzzy interpretation. This can be done in logarithmic space because there are polynomially many triggers to consider (in particular, there are polynomially many groundings ρ to consider, as they assign to variables only those constants, for which some atom in the current interpretation has a non-zero truth degree), so we can inspect all of them keeping in memory only a single trigger with the highest truth degree of $\rho(\text{body}(r))$. Applying the computed trigger is also feasible in logarithmic space. Since a truth-greedy chase has a polynomial length, we can compute the final interpretation and check if the truth degree of G in this interpretation is at least c, in polynomial time. □

Since each program with no existential variables is weakly acyclic, we obtain as a corollary that entailment in t-Datalog (t-Datalog$^\exists$ with no existential variables) is also in P for data complexity. Matching lower bound follows from Theorem 2 and P-hardness of entailment in Datalog for data complexity.

Corollary 15. *ENTAILMENT for t-Datalog programs is* P*-complete in data complexity.*

5 Adding Negation and Other Unary Operators

In this section, we will show how to extend t-Datalog with unary operators (also studied under the name of *negators* [22]) without negative impact on the complexity of reasoning. Unary operators are interpreted in our fuzzy setting as functions mapping a truth degree of a ground atom into another truth degree. A flagship unary operator is negation, which in the fuzzy setting can be interpreted in various ways. However, there are also many other unary operators worth considering. For instance, threshold operators Δ_T, for $T \in [0,1]$ which assign truth degree 1 to atoms which have truth degree at least T. Below we present semantics of two versions of fuzzy negation and of the threshold operator:

$$\mathcal{I}(\neg A) = 1 - \mathcal{I}(A), \qquad \mathcal{I}(\sim A) = \begin{cases} 1, & \text{if } \mathcal{I}(A) = 0, \\ 0, & \text{otherwise,} \end{cases} \qquad \mathcal{I}(\Delta_T A) = \begin{cases} 1, & \text{if } \mathcal{I}(A) \geq T, \\ 0, & \text{otherwise.} \end{cases}$$

Formally, we let a unary operator be any computable function $U : [0,1] \longrightarrow [0,1]$. A t-Datalog$_U$ program is defined similarly as a t-Datalog program, but body atoms in rules can be preceded by arbitrary unary operators.[4] Hence, rules of a t-Datalog$_U$ program are of the form

$$U_1 R_1(\mathbf{x_1}) \odot_r \cdots \odot_r U_\ell R_\ell(\mathbf{x}_\ell) \to S(\mathbf{x_h}, \mathbf{z}),$$

where each U_i is a unary operator or is empty. As usual in Datalog with negation [1], we define a notion of a stratified program as follows. We let a *stratification* of a t-Datalog$_U$ program Π be any function σ mapping predicates mentioned in Π to positive integers such that for each rule $r \in \Pi$ and all predicates P, P^+, and P^- mentioned, respectively, in the head, in body atoms not using unary operators, and in body atoms using unary operators of r, it holds that $\sigma(P^+) \leq \sigma(P)$ and $\sigma(P^-) < \sigma(P)$. We will treat a stratification σ as a partition of Π into $\Pi_1, \ldots, \Pi_n$ such that n is the maximal value assigned by σ and Π_i consists of all rules in Π with heads P such that $\sigma(P) = i$. A program is *stratifiable* (or stratified) if it has some stratification.

Semantics of unary operators is straightforward, namely $\mathcal{I}(UA) = U(\mathcal{I}(A))$ for each fuzzy interpretation $\mathcal{I}$ and unary operator U. The definition of entailment, however, is more complex; we will adapt the procedural semantics used in the non-fuzzy setting [1]. To this end, we start by considering *semi-positive* t-Datalog$_U$ programs, in which unary operators appear only in front of extensional predicates (i.e., predicates which do not occur rule heads). We can use the argumentation from the proof of Theorem 7 to show that semi-positive t-Datalog$_U$ programs, similarly to t-Datalog programs, have the unique fuzzy model property.

Proposition 16. *Each pair of a semi-positive t-Datalog$_U$ program Π and a fuzzy dataset $\mathcal{D}$ has a unique fuzzy universal model.*

Next, we exploit Proposition 16, to define semantics for stratifiable programs. Given a stratification $\Pi_1, \ldots, \Pi_n$ of Π and a fuzzy dataset $\mathcal{D}$ we let $\mathcal{I}_0, \mathcal{I}_1, \ldots, \mathcal{I}_n$ be a sequence of fuzzy interpretations such that $\mathcal{I}_0 = \mathcal{I}_\mathcal{D}$ and each $\mathcal{I}_i$ with $i > 1$ is the fuzzy universal model of Π_i and (the dataset representation of) $\mathcal{I}_{i-1}$. By the definition, each Π_i is semi-positive, so $\mathcal{I}_i$ is well defined by Proposition 16. We call $\mathcal{I}_n$ the result of applying $\Pi_1, \ldots, \Pi_n$ to $\mathcal{D}$. Similarly, as in the non-fuzzy setting, we can show that each stratification results in the same interpretation.

Proposition 17. *Let Π be a stratifiable t-Datalog$_U$ program and $\mathcal{D}$ a fuzzy dataset. Applying any stratification of Π to $\mathcal{D}$ results in the same fuzzy interpretation.*

[4] Similarly as in the case of t-norms we slightly abuse notation by using the same symbol for a unary operator and a function interpreting this operator.

Establishing Proposition 17 allows us to introduce a standard definition of entailment in stratified programs. We say that an atom G is *c-entailed*, for any $c \in [0, 1]$, by a stratifiable Π and $\mathcal{D}$ if $\mathcal{I}_n(G) \geq c$, where $\mathcal{I}_n$ is the result of applying any stratification of Π to $\mathcal{D}$. We define the Entailment problem analogously as in the case of t-Datalog$^\exists$, namely, given a t-Datalog$_U$ program Π, a fuzzy dataset $\mathcal{D}$, $G \in \mathsf{GAtoms}$, and $c \in [0, 1]$, the problem is to check if Π and $\mathcal{D}$ c-entail G.

The main result of this section is that adding arbitrary unary operators to t-Datalog does not increase the complexity of entailment checking, as long as the input program is stratifiable. In other words, entailment checking for stratifiable t-Datalog$_U$ programs is no harder than entailment checking for t-Datalog programs. As in the case of t-norms, our complexity analysis does not take into account the complexity of computing application of unary operators. Although such computations are usually of low complexity (e.g., in the case of $\neg$, $\sim$, and Δ_T introduced at the beginning of this section), but one can introduce much more complex unary operators.

Theorem 18. *Entailment for stratifiable t-Datalog$_U$ programs is* P*-complete in data complexity.*

Proof sketch. The lower bound is inherited from Datalog. For the upper bound, we compute any stratification $\Pi_1, \dots, \Pi_n$ of the input program. Then we compute the (unique) universal fuzzy models $\mathcal{I}_1, \dots, \mathcal{I}_n$ corresponding to application of the stratification. To compute each $\mathcal{I}_{i+1}$ from $\mathcal{I}_i$ and Π_{i+1} we apply the truth-greedy chase. By our results on truth-greedy chases in Section 4, this procedure allows us to construct $\mathcal{I}_n$ in polynomial time with respect to $\mathcal{D}$. Finally, checking if $\mathcal{I}_n(G) \geq c$, for input G and c, is clearly feasible in polynomial time. □

6 Discussion and Future Work

Summary. We have introduced t-Datalog$^\exists$, an extension of Datalog$^\exists$ for reasoning about uncertain information, which can be used for neuro-symbolic reasoning. By allowing for arbitrary t-norms in the place of conjunctions in standard Datalog programs, t-Datalog$^\exists$ provides us with a highly flexible fuzzy formalism. We have established a chase-based reasoning technique applicable to the fuzzy setting of t-Datalog$^\exists$, which we used for computational complexity analysis. For example, the complexity of entailment in weakly-acyclic t-Datalog$^\exists$ matches that of Datalog$^\exists$, namely it is P-complete for data complexity. Our fuzzy chase procedure is worst-case optimal for such reasoning. Moreover, we showed that t-Datalog can be extended with arbitrary (fuzzy) unary operators without a negative impact on the complexity. Our development has purposefully followed that of Datalog$^\exists$ in order to leverage the wide range of results known in the classical case. The obtained results illustrate the advantages of this approach and lay the foundation to richer fuzzy ontology languages in which reasoning can be performed efficiently, akin to the Datalog$^\pm$ family of languages [26].

Discussion. It is worth discussing some generalisations of the presented work. Recall that to simplify the presentation, we have focused on the proofs on K-satisfiability with $K = 1$. However, our techniques are not specific to this case, and our results can be extended to arbitrary $K \in [0, 1]$. Furthermore, observe that the presented syntax of t-Datalog$^\exists$ requires that each rule mentions at most one t-norm. For example, we do not allow for a rule of the form $P(x) \odot_{Ł} Q(x) \odot_{\mathsf{prod}} R(x) \rightarrow S(x)$, which mentions two different t-norms. We have imposed such a restriction in order to obtain associativity of operators in rule bodies, which is the case in standard Datalog and logic programming. However, if we fix the order of t-norms application in rule bodies, we can use multiple t-norms within a single rule and it would not impact our technical results. Furthermore, we have only explicitly considered atomic fact entailment and data complexity measure in the paper, but our techniques are also applicable to more complex queries and allow for performing analysis of other measures of computational complexity (e.g., combined or program complexity).

Finally, in our presentation we defined the semantics of $\rightarrow$ as the residuum of the Łukasiewicz t-norm. With respect to 1-satisfiability, every residuum of a t-norm is equivalent: under any of them a rule would be 1-satisfied exactly when the truth degree of the head is at least the truth degree of the body. With respect to 1-fuzzy models all these alternative semantics for $\rightarrow$ are the same and our results hold unchanged in those cases. For $K < 1$, individual analysis may be required to confirm whether our results still hold for alternative semantics for $\rightarrow$.

Future Work. Our research on t-Datalog$^\exists$ is motivated by the need for practical and powerful reasoning formalism for heterogeneous data sources. We have intentionally chosen to align the development of t-Datalog$^\exists$ closely with Datalog$^\exists$, to exploit results known for the latter. Another great benefit of this approach is that it opens a way for implementing reasoning procedures for t-Datalog$^\exists$ by extending existing Datalog$^\exists$ reasoning systems. In particular, developing implementations on top of established chase-based reasoners, such as Vadalog [8], form an attractive opportunity for future work.

The results obtained in the paper also introduce interesting theoretical research directions. Notice that in the presented results we focus on programs with finite Datalog$^\exists$ chase which, as we have shown, allows us to construct finite chases in t-Datalog$^\exists$. However, there are prominent fragments of Datalog$^\exists$, such as (*weakly*) *guarded* [13] or *warded* [9] programs, where entailment is decidable even though the chase is generally not finite. Our truth-greedy chase technique does not provide us with an appropriate tool for reasoning in such cases. For instance, consider the behaviour of a truth-greedy chase applied to a guarded program as described below.

Example 19. Consider a fuzzy dataset $\mathcal{D}$ with $\mathcal{D}(R(a,b)) = 1$ and $\mathcal{D}(A) < 1$ for all other atoms A for which $\mathcal{D}(A)$ is defined. Moreover, consider a guarded t-Datalog$^\exists$ program Π which, among others, contains a rule r of the form $R(x,y) \rightarrow \exists z\, R(y,z)$. A truth-greedy chase of Π and $\mathcal{D}$ applies this single rule infinitely many times before applying any other rule in Π. Since other rules are not applied, it is possible that some triggers remain active infinitely. Hence the fairness condition—Item (iv) in the definition of a chase—does not hold.

The question of how to decide entailment in t-Datalog$^\exists$ with infinite chases presents an interesting research challenge for future work. Bogwardt et al. [11] showed that reasoning in fuzzy variants of the description logic $\mathcal{EL}$ with Łukasiewicz semantics is decidable. While $\mathcal{EL}$ is closely related to guarded Datalog$^\exists$, the undecidability proof in [11] relies on conjunction in rule heads. In contrast to the Boolean case, this is not equivalent to single atoms in the head in the fuzzy setting and their undecidability argument does not imply undecidability of guarded t-Datalog$^\exists$.

Decidability in guarded and warded Datalog$^\exists$, in particular, follows from the fact that the infinite part of the chase only repeats certain patterns. In future we will aim to exploit this idea in the fuzzy setting of t-Datalog$^\exists$, to introduce adequate fuzzy chase procedures and establish computational complexity results for guarded and warded fragments of t-Datalog$^\exists$.

Acknowledgements

Matthias Lanzinger acknowledges support by the Royal Society “RAISON DATA” project (Reference No. RP\R1\201074) and by the Vienna Science and Technology Fund (WWTF) [10.47379/ICT2201, 10.47379/VRG18013, 10.47379/NXT22018]. Przemysław A Wałęga was supported bu the EPSRC projects OASIS (EP/S032347/1), ConCuR (EP/V050869/1) and UK FIRES (EP/S019111/1), as well as SIRIUS Centre for Scalable Data Access and Samsung Research UK. Stefano Sferrazza was supported by the Vienna Science and Technology Fund (WWTF) [10.47379/VRG18013]. Georg Gottlob acknowledges support by the NextGenerationEU NRRP MUR program "FAIR - Future AI Research (PE00000013)". For the purpose of Open Access, the authors have applied a CC BY public copyright licence to any Author Accepted Manuscript (AAM) version arising from this submission.

References

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[2] Ágnes Achs and Attila Kiss. Fuzzy extension of Datalog. *Acta Cybernetica*, 12(2):153–166, 1995.

[3] Mario Alviano, Matthias Lanzinger, Michael Morak, and Andreas Pieris. Generative Datalog with stable negation. In *Proceedings of PODS*, pages 21–32, 2023.

[4] Kay R Amel. From shallow to deep interactions between knowledge representation, reasoning and machine learning. In *Proceedings of SUM*, pages 16–18, 2019.

[5] Jean-François Baget, Michel Leclère, Marie-Laure Mugnier, and Eric Salvat. On rules with existential variables: Walking the decidability line. *Artificial Intelligence*, 175(9-10):1620–1654, 2011.

[6] Sébastien Bardin, Somesh Jha, and Vijay Ganesh. Machine learning and logical reasoning: The new frontier (Dagstuhl Seminar 22291). In *Dagstuhl Reports*, 2023.

[7] Catriel Beeri and Moshe Y Vardi. The implication problem for data dependencies. In *Automata, Languages and Programming: Eighth Colloquium Acre (Akko)*, pages 73–85, 1981.

[8] Luigi Bellomarini, Davide Benedetto, Georg Gottlob, and Emanuel Sallinger. Vadalog: A modern architecture for automated reasoning with large knowledge graphs. *Information Systems*, 105:101528, 2022.

[9] Gerald Berger, Georg Gottlob, Andreas Pieris, and Emanuel Sallinger. The space-efficient core of vadalog. *ACM Trans. Database Syst.*, 47(1):1:1–1:46, 2022.

[10] Fernando Bobillo and Umberto Straccia. fuzzyDL: An expressive fuzzy description logic reasoner. In *Proceedings of IEEE International Conference on Fuzzy Systems*, pages 923–930, 2008.

[11] Stefan Borgwardt, Marco Cerami, and Rafael Peñaloza. The complexity of fuzzy EL under the łukasiewicz t-norm. *Int. J. Approx. Reason.*, 91:179–201, 2017.

[12] Marco Calautti and Andreas Pieris. Semi-oblivious chase termination: The sticky case. *Theory of Computing Systems*, 65(1):84–121, 2021.

[13] Andrea Calì, Georg Gottlob, and Michael Kifer. Taming the infinite chase: Query answering under expressive relational constraints. *Journal of Artificial Intelligence Research*, 48:115–174, 2013.

[14] Andrea Calì, Georg Gottlob, and Thomas Lukasiewicz. A general Datalog-based framework for tractable query answering over ontologies. In *Proceedings of PODS*, pages 77–86, 2009.

[15] Andrea Calì, Georg Gottlob, and Andreas Pieris. Query answering under non-guarded rules in datalog+/-. In *Proceedings of Web Reasoning and Rule Systems*, pages 1–17, 2010.

[16] Maria Eugenia Cornejo, David Lobo, and Jesús Medina. Syntax and semantics of multi-adjoint normal logic programming. *Fuzzy Sets and Systems*, 345:41–62, 2018.

[17] Luc De Raedt, Sebastijan Dumančić, Robin Manhaeve, and Giuseppe Marra. From statistical relational to neuro-symbolic artificial intelligence. *arXiv preprint arXiv:2003.08316*, 2020.

[18] Rafee Ebrahim. Fuzzy logic programming. *Fuzzy Sets and Systems*, 117(2):215–230, 2001.

[19] Patrik Eklund and Frank Klawonn. Neural fuzzy logic programming. *IEEE Transactions on Neural Networks*, 3(5):815–818, 1992.

[20] Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, and Lucian Popa. Data exchange: semantics and query answering. *Theoretical Computer Science*, 336(1):89–124, 2005.

[21] Fahimeh Farahbod and Mahdi Eftekhari. Comparison of different t-norm operators in classification problems. *arXiv preprint arXiv:1208.1955*, 2012.

[22] János Fodor and Ronald R Yager. Fuzzy set-theoretic operators and quantifiers. In *Fundamentals of Fuzzy Sets*, pages 125–193. 2000.

[23] Artur d'Avila Garcez, Marco Gori, Luis C Lamb, Luciano Serafini, Michael Spranger, and Son N Tran. Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning. *arXiv preprint arXiv:1905.06088*, 2019.

[24] Artur d'Avila Garcez and Luis C Lamb. Neurosymbolic AI: The 3rd wave. *Artificial Intelligence Review*, pages 1–20, 2023.

[25] Georg Gottlob, Thomas Lukasiewicz, and Andreas Pieris. Datalog+/-: Questions and answers. In *Proceedings of KR*, 2014.

[26] Georg Gottlob, Thomas Lukasiewicz, and Andreas Pieris. Datalog+/-: Questions and answers. In *Proceedings KR*, 2014.

[27] Gösta Grahne and Adrian Onet. Anatomy of the chase. *Fundamenta Informaticae*, 157(3):221–270, 2018.

[28] Petr Hájek. *Metamathematics of Fuzzy Logic*, volume 4 of *Trends in Logic*. Kluwer, 1998.

[29] Pascual Julián Iranzo and Fernando Sáenz-Pérez. A fuzzy Datalog deductive database system. *IEEE Transactions on Fuzzy Systems*, 26(5):2634–2648, 2018.

[30] Erich-Peter Klement, Radko Mesiar, and Endre Pap. *Triangular Norms*, volume 8 of *Trends in Logic*. Springer, 2000.

[31] Matthias Lanzinger, Stefano Sferrazza, and Georg Gottlob. MV-Datalog+-: Effective rule-based reasoning with uncertain observations, 2022.

[32] Matthias Lanzinger, Stefano Sferrazza, and Georg Gottlob. New perspectives for fuzzy Datalog (extended abstract). In *Proceedings of Datalog 2.0*, pages 42–47, 2022.

[33] FBK Luciano Serafini, Samy Badreddine, AI Sony, and Japan Ivan Donadello. Logic tensor networks: Theory and applications. *Neuro-Symbolic Artificial Intelligence: The State of the Art*, 342:370, 2022.

[34] Thomas Lukasiewicz and Umberto Straccia. Managing uncertainty and vagueness in description logics for the semantic web. *Journal of Web Semantics*, 6(4):291–308, 2008.

[35] Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Neural probabilistic logic programming in DeepProbLog. *Artificial Intelligence*, 298:103504, 2021.

[36] Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, and Marco Gori. Lyrics: A general interface layer to integrate logic inference and deep learning. In *Proceedings of ECML PKDD*, pages 283–298, 2019.

[37] Jesús Medina, Manuel Ojeda-Aciego, and Peter Vojtás. A procedural semantics for multi-adjoint logic programming. In *Proceedings of EPIA*, pages 290–297, 2001.

[38] George A Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

[39] Luciano Serafini and Artur S d'Avila Garcez. Learning and reasoning with logic tensor networks. In *Proceedings of AIxIA*, pages 334–348, 2016.

[40] Arseny Skryagin, Wolfgang Stammer, Daniel Ochs, Devendra Singh Dhami, and Kristian Kersting. Neural-probabilistic answer set programming. In *Proceedings of KR*, pages 463–473, 2022.

[41] Umberto Straccia. Reasoning within fuzzy description logics. *Journal of Artificial Intelligence Research*, 14:137–166, 2001.

[42] Zhun Yang, Adam Ishay, and Joohyung Lee. NeurASP: Embracing neural networks into answer set programming. *arXiv preprint arXiv:2307.07700*, 2023.

[43] Xiaohong Zhang, Huacan He, and Yang Xu. A fuzzy logic system based on Schweizer-Sklar t-norm. *Science in China Series F: Information Sciences*, 49(2):175–188, 2006.

A Details for Section 5

We follow [1] in the following definitions as well as the line or argumentation. We will use a slightly different notation that in the main body of the paper, which is more convenient for proofs.

The *extensional predicates* ($\mathsf{edb}(\Pi)$) of a t-Datalog$_U$ program Π are those relation symbols that only occur in rule bodies of Π. The *intensional predicates* ($\mathsf{idb}(\Pi)$) are those relation symbols that occur in the head of some rule of Π. We know by Lemma 13 that every truth-greedy chase for t-Datalog$^\exists$ programs (and thus also for semi-positive t-Datalog$_U$ programs) is finite. We write $\mathsf{tgc}(\Pi, \mathcal{D})$ for the result of applying a truth-greedy chase[5] to Π and $\mathcal{D}$. Since $\mathsf{tgc}(\Pi, \mathcal{D})$ is always finite, we will treat $\mathsf{tgc}(\Pi, \mathcal{D})$ as a fuzzy dataset. As above, for $G \in \mathsf{GAtoms}$, we write $G \in \mathcal{D}$ to mean that $\mathcal{D}(G)$ is defined.

The notion of semipositive programs is lifted from Datalog$^\neg$. By convention we assume everywhere in the following that $\mathcal{D}$ is defined only for extensional predicates.

Proposition 20 (cf. Theorem 15.2.2 [1])**.** *Let Π be a semipositive program and $\mathcal{D}$ a fuzzy dataset. Then there is a unique fuzzy universal model of Π and $\mathcal{D}$.*

Proof sketch. For each unary operator U and $R \in \mathsf{edb}(\Pi)$, let R^U be a new relation with $\mathcal{D}'(R^U(\mathbf{c})) = U(\mathcal{D}(R(\mathbf{c})))$ for every ground atom where $\mathcal{D}(R(\mathbf{c}))$ is defined. Let $\mathcal{D}'(G) = \mathcal{D}(G)$ for all other $G \in \mathcal{D}$. Let Π' be the t-Datalog$^\exists$ program obtained by replacing every instance of $U\,R$ in the body of a rule with R^U. The unique fuzzy universal model $\mathcal{I}$ of Π' and $\mathcal{D}'$ (recall Theorem 7) is the desired unique fuzzy universal model of Π and $\mathcal{D}$. □

Accordingly, we use the construction of Π' and $\mathcal{D}'$ in the proof sketch of Proposition 20 also to define the semantics of a chase on semipositive programs: the result of applying the chase to semipositive program Π and dataset $\mathcal{D}$ is the result of applying the chase to Π' and $\mathcal{D}'$. We write $\Pi(\mathcal{D})$ for the unique fuzzy universal model of Π and $\mathcal{D}$ from Proposition 20.

A *stratification* of a t-Datalog$_U$ program Π is a sequence of t-Datalog$_U$ programs $\Pi_1, \dots, \Pi_n$ such that for some mapping $\zeta : \mathsf{idb}(\Pi) \to [n]$,

(i) $\{\Pi_1, \dots, \Pi_\ell\}$ is a partition of Π.
(ii) for each relation symbol $R \in \mathsf{idb}(\Pi)$, all rules with R in head are in $\Pi_{\zeta(R)}$.
(iii) If there is a rule with R in its head and $R' \in \mathsf{idb}(\Pi)$ in its body, then $\zeta(R') \leq \zeta(R)$.
(iv) If there is a rule with R in its head and $U\,R'$ in its body, for some unary operator U, then $\zeta(R') < \zeta(R)$.

A program is *stratifiable* if it has a stratification.

The semantics of a t-Datalog$_U$ program Π is defined in terms of a procedure. Let $\zeta = P_1, \dots, P_n$ be a stratification of Π. Note that every Π_i is semipositive. Define

$$\mathcal{D}_0 = \mathcal{D} \tag{1}$$

$$\mathcal{D}_i = \mathsf{tgc}(\Pi_i, \mathcal{D}_{i-1}) \tag{2}$$

We denote $\mathcal{D}_n$ obtained from this procedure as $\zeta(\Pi, \mathcal{D})$, and refer to it the semantics of a t-Datalog$_U$ program under ζ. We say that two stratifications of a t-Datalog$_U$ program are *equivalent* if they have the same semantics for all fuzzy datasets.

[5] Recall that so- and r-chase are equivalent without existential quantifiers.

For Datalog $^\neg$ it is well known that all stratifications lead to equivalent semantics [1]. The semantics of the formalism only affect a single key lemma, that we reprove below for our setting.

Lemma 21. *Let Π be a semipositive t-Datalog$_U$ program and ζ a stratification for Π. Then $\Pi(\mathcal{D}) = \zeta(\mathcal{D})$ for every fuzzy dataset $\mathcal{D}$.*

Proof. Recall that we the semantics of semipositive programs are the result of applying the truth-greedy chase on Π' and $\mathcal{D}'$ as defined in the proof of Proposition 20. Let s_i be the truth-greedy chase sequence of (2). It is enough to observe that their concatenation $s^* = s_1 s_2 \cdots s_n$ is a finite chase of Π' and $\mathcal{D}'$. By Theorem 9 and Theorem 7 this will produce the unique fuzzy universal model from Proposition 20.

To see that the concatenation is indeed a finite chase of Π' and $\mathcal{D}'$ we need to only check that there is no active trigger at the end of s^*. We argue via induction on the the concatenation of the first $i \leq n$ sequences $s^{(i)} = s_1 \cdots s_i$ that in the interpretation obtained from applying $s^{(i)}$ to Π' and $\mathcal{D}'$, no trigger for rules in $\Pi_1, \ldots, \Pi_i$ is active. First, the trigger cannot be for a rule in Π_i as it is the result of applying the chase on $\mathcal{D}_{i-1}$. Second, by inductive assumption no triggers from $\Pi_1, \ldots, \Pi_{i-1}$ were active after $s^{(i-1)}$. But by Item (iii) and Item (iv), the heads of Π_i cannot occur in the bodies of Π_j with $j < i$. Therefore, the addition of s_i also cannot make any trigger for rules in $\Pi_1, \ldots, \Pi_{i-1}$ active. □

Theorem 22. *Let Π be a stratifiable t-Datalog$_U$ program. All stratifications of Π are equivalent.*

Proof sketch. The proof of Theorem 15.2.10 in [1] holds also for our setting by simply replacing the key Lemma 15.2.9 there with Lemma 21. □

That is, we can use any stratification to compute the semantics of t-Datalog$_U$ programs, just as for stratified Datalog $^\neg$. Computing a stratification ζ is possible in polynomial time in the size of a dataset (see also Proposition 15.2.7 [1]) and thus computing the truth degree of $G \in \mathsf{GAtoms}$ in $\zeta(\mathcal{D})$ is also polynomial (cf. Corollary 15). Thus, entailment in t-Datalog$_U$ is in P for data complexity, as stated in Theorem 18.