Fuzzy logic abduction

Peter Vojtáš[1]

Abstract. Abduction is an important form of nonmonotonic reasoning allowing one to find explanations for certain symptoms or observations. We consider the situation when the application domain is described by a logic program (IF-THEN rules). To model uncertainty in human cognition and real world applications, we use fuzzy logic (i.e. many valued logic where axioms can have truth values (confidence factors) smaller than 1). Our connectives are tunable to fit real world data. We introduce a model of fuzzy logic programming abduction problem FLPAP which has many applications in diagnostical systems. We show soundness and completeness of declarative and procedural semantics of FLPAP. We show how can linear programming be applied in order to get minimal solutions wrt a cost function. We discuss the problem of approximative explanations in finitely valued logic and integer programming.

1 Introduction

This paper presents a model of abduction under uncertainty based on fuzzy logic programming. Abduction is an important form of nonmonotonic reasoning allowing one to find explanations and/or causes for certain symptoms or observations (introduced by C. S. Peirce in [8]).

We restrict ourselves to logic based abduction, especially to abduction applied to logic programming (see [3], [6]). To model uncertainty in human cognition and real world applications we offer the possibility to use fuzzy logic (see [5]).

2 Example

2.1 The application domain

The following illustrative example from the domain of motor vehicles is inspired by [2] and [3] (subscripts L, P, G denote connectives of Łukasiewicz, product and Gödel logic)). Consider the following **fuzzy logic program** P_{mv} (fuzzy IF-THEN rule base) in a prolog syntax with attached truth values (which can be also understood as a fuzzy set of rules, i.e. a partial mapping P_{mv} : Rules $\longrightarrow (0,1]$)

high_fuel_consumption $\longleftarrow_G$ rich_mixture$\wedge_L$ low_oil; 0.8.
overheating $\longleftarrow_P$ low_oil; 0.5.
noisy_behaviour $\longleftarrow_P$ rich_mixture; 0.8.
overheating $\longleftarrow_L$ low_water; 0.9.
noisy_behaviour $\longleftarrow_P$ low_oil; 1.

Denote the set of all propositional variables describing our universe of discourse Var_{mv}.

2.2 Observations

If we observe that our car is noisy, overheated and has a high fuel consumption, we would like to know why it is so. Let us call these variables **observation variables** and denote it $OV_{mv} = \{$ noisy_behaviour, overheated, high_fuel_consumption$\}$. The second parameter of our abduction problem are **observations** (sometimes called manifestations, symptoms, effects) represented by a fuzzy theory $OBS_{mv} : OV_{mv} \longrightarrow [0,1]$ consisting of facts=observation variables equipped with truth values

high_fuel_consumption; 0.25
overheating; 0.25.
noisy_behaviour; 0.5

2.3 Hypothesis and explanations

Possible explanations will be fuzzy subsets of the set of **hypothesis** $H_{mv} = \{$ rich_mixture, low_oil, low_water $\}$. This makes sense, because in the realm of a theory and of observations suffering from uncertainty, we can expect that certain level of confidence of hypothesis can be (under the presence of the theory) an explanation of these (uncertain) observations.

A fuzzy set $E : H_{mv} \longrightarrow [0,1]$ is an **explanation** of our fuzzy logic programming abduction problem (FLPAP) $\mathcal{A}_{mv} =< P_{mv}, OBS_{mv}, H_{mv} >$ given by the theory P_{mv}, observations OBS_{mv} and the set of hypothesis H_{mv}, if the theory $P_{mv} \cup E$ is consistent and $P_{mv} \cup E$ semantically implies all observations OBS_{mv}. In our case FLPAP $\mathcal{A}_{mv}$ has an infinite set of solutions, it contains especially the explanation which takes values constantly 1 (we do not have negation in our example). The fuzzy theory E_2 defined as follows $E_2($ rich_mixture $) = E_2($ low_oil $) = 0.7$ and $E_2($ low_water $) = 0$ is also an explanation of FLPAP $\mathcal{A}_{mv}$. Indeed, for an arbitrary truth assignment of all propositional variables $I : Var_{mv} \longrightarrow [0,1]$ holds: if I is a model of both P_{mv} and E_2 then we can calculate that for all $m \in OV_{mv}$ is $I(m) \geq OBS_{mv}(m)$. Because I is a model of E_2 we have e.g. $I($ low_oil $) \geq E_2($ low_oil $) = 0.7$ (using truth value functions for connectives) we get from the second rule that either

[1] Institute of Computer Science, Academy of Sciences of Czech republic, Pod vodárenskou věží 2, 18207 Prague, Czech republic, email: vojtas at uivt.cas.cz, Dpt. Comp. Sci. University of PJŠafárik Košice, Slovakia, partialy supported by COST 15 and VEGA 1/4375/97

$I(\text{ overheating }) \geq I(\text{low\_oil}) \geq 0.7$ or

$$0.5 \leq \frac{I(\text{ overheating })}{I(\text{ low\_oil })} \leq \frac{I(\text{ overheating })}{0.7}$$

hence in both cases is $I(\text{ overheating }) \geq 0.35 > 0.25 = OBS_{mv}(\text{ overheating })$ (for other observations similarly). □

3 Fuzzy logic programming

3.1 The language

Our **language** of propositional fuzzy logic consists of a finite set Var of propositional (sentential) variables and several rationality preserving and definable connectives: conjunctions $\&_1, \&_2, \ldots, \&_k$, disjunctions $\vee_1, \vee_2, \ldots, \vee_l$, implications $\rightarrow_1, \rightarrow_2, \ldots, \rightarrow_m$ and the negation $\neg$. For a connective c the corresponding semantical counterpart - the truth value function will be denoted by $\dot{c}$ (i.e. with a dot over the very connective, e.g. $\dot{\neg}(x) = 1 - x$). Conjunctions are assumed to be t-norms, disjunctions are t-conorms ([7]). We assume that for all implications in our language there is a generating t-norm T such that $\dot{\rightarrow}_T(u, v) = \sup\{w : T(u, w) \leq v\}$ (see e.g. [4], [5]). Note that T need not be a truth value of any connective in our language (e.g. in our example $\dot{\&}_P$ is generating $\rightarrow_P$, although $\&_P$ does not appear in our program).

3.2 Fuzzy theories

A **fuzzy theory** is a fuzzy set of propositions Th : Formulas $\longrightarrow [0, 1] \cap Q$ assigning each proposition a rational number. A single axiom will be denoted as a pair $(C; Th(C))$ (see [5]). **Semantical interpretations** of our language $I : Var \longrightarrow [0, 1]$ are assignments of variables with truth values (real numbers, [5]). A truth value assignment I is a **model of a fuzzy theory** Th if for all formulas C is $Th(C) \leq I(C)$. This definition says, that a theory and data fit together if the expert given truth value is a lower bound for truth values from data. Of course it is interesting to have the strongest theory fitting data (completeness problem). Another problem is whether our data are truth value functional (if not we have to use other approaches e.g. probabilistic, belief functions,..., see e.g. [1]).

3.3 Fuzzy logic programs

In general, a formula B is called a **body** if it is in disjunctive normal form. A **rule** is a proposition of form $A \longleftarrow B$, where A is a propositional variable and B is a body. A propositional variable is also called a **fact**. A **fuzzy logic program (FLP)** is a fuzzy theory P : Rules $\cup$ Facts $\longrightarrow (0, 1] \cap Q$ such that the domain of P is finite and consists of rules and facts. A propositional variable can be intended as a **query** (in this case it is denoted by A? (note, we do not adopt refutation here). For a fuzzy logic program P and a query A? a **correct answer** is a real number $x \in [0, 1]$ such that for all truth assignments (semantic interpretations) I the following holds: If I is a model of the fuzzy theory P then $I(A) \geq x$.

3.4 Procedural semantics of FLP

As far as our connectives are subject for tuning and/or learning we are in a logical theory in which we (possibly) do not have any logical axioms. Motivation for our procedural semantics is based on the "backwards" understanding of many-valued modus ponens (do not confuse it with so called fuzzy modus ponens which is an interpolation and/or extrapolation deduction rule for deduction by analogy). Many valued modus ponens is a sound rule ([4], [5], [9]) which looks as follows:

$$\begin{array}{rl} \text{From} & (B \longrightarrow_T A; y) \\ \text{and} & (B; x) \\ \text{infer} & (A; T(y, x)). \end{array}$$

In this case we say that the generating t-norm T evaluates modus ponens. The backward understanding argues as follows: Querying A, we look for an implication with A in the head, choose one of them, say $(B \longrightarrow_T A, y)$. From now on we know that the truth value of our answer will have value $T(y, b)$ for some b, where b will be one of (in our case (B, x)) computed truth values of B. We can write our backward modus ponens inference as

$$\begin{array}{rl} A? & \text{deduce by } (A \longleftarrow_T B; y) \text{ to} \\ T(y, B) & \text{deduce by } (B; x) \text{ to} \\ T(y, x) & \end{array}$$

3.5 Admissible rules

Our inference rules are in the form of a context free grammar, where X and Y are variables for words (possibly empty).

Rule 1. From XAY infer $XT(P(A \leftarrow_T B), B)Y$

Rule 2. From XAY infer $XP(A)Y$

Rule 3. From $XB_1\&B_2Y$ infer $X\dot{\&}(B_1, B_2)Y$

Rule4a. From $XB_1 \vee B_2Y$ infer $X \dot{\vee}(B_1, B_2)Y$

Rule4b. From $XB_1 \vee B_2Y$ infer $X \dot{\vee}(B_1, 0)Y$

Rule4c. From $XB_1 \vee B_2Y$ infer $X\dot{\vee}(0, B_2)Y$ (we need these because of possibility one member of disjunction fails and then there is no aggregation of confidences of single diagnosis)

Rule 5. From $X\neg BY$ infer $X1 - \max(b_1, \ldots, b_n)Y$ if the computational tree for P and B? is finite and $b_1, \ldots, b_n$ are calculated values of all successful branches (failed branches are interpreted to have value 0).

Rule 6. If there are in the word X no letters denoting propositional variables (the inference was successful), then the word X is a composition of truth value functions for connectives and evaluating t-norms with rational arguments, then from X infer the value of that expression.

For a fuzzy program P and a query A? we say that a rational number $q \in [0, 1]$ is a **computed answer** for P and A? if there is a sequence $G_0, G_1, \ldots, G_n$ (successful computation) such that $G_0 = A$, $G_n = q$ and for all $i < n$, G_{i+1} is inferred from G_i by one of admissible rules.

In [9] there is developed a fixpoint semantics for definite fuzzy logic programs (i.e. without negation and hence without Rule 5) and soundness and approximative completeness of procedural semantics is proved (for lower semicontinuous conjunctions, disjunctions and generating t-norms).

Rule 5 gives consistent results for our fuzzy logic programs, but it is out of the scope of this paper to discuss a semantics for it.

4 Fuzzy abduction

4.1 Definition of the problem

For two fuzzy theories T and S denote by $T \cup S$ the fuzzy theory defined by $(T \cup S)(A) = \max\{T(A), S(A)\}$.

Definition 1 *A* **fuzzy logic programming abduction problem (FLPAP)** *consists of a tuple $\mathcal{A} =< P, OBS, H >$, where $H \subseteq Var$ is the set of hypothesis, $OBS : OV \longrightarrow [0,1]$ is the fuzzy theory of observations (where OV is the set of observation variables such that $OV \cap H = \emptyset$) and P is a fuzzy logic program (where intended meaning is that observation variables should not be explained by themselves).*

Definition 2 *A fuzzy theory $E : H \longrightarrow [0,1]$ is a* **correct explanation** *to FLPAP $\mathcal{A} =< P, OBS, H >$ if*
(1) $P \cup E$ semantically implies OBS, i.e. for every semantical interpretation (truth assignment of variables) $I : Var \longrightarrow [0,1]$ the following holds: If I is a model of $P \cup E$ then I is a model of OBS.
(2) $P \cup E$ is consistent

It will be useful to represent solutions as a subset of many dimensional unit cube. Namely, if $H = \{h_1, \ldots, h_n\}$ is the set of hypothesis and $E : H \longrightarrow [0,1]$ is a solution, it is uniquely determined by its values $(E(h_1), \ldots, E(h_n)) \in [0,1]^n$. So having an element of n^{th} power of unit cube $e = (e_1, \ldots, e_n) \in [0,1]^n$ it represents the mapping $E_e(h_i) = e_i$. Denote by $\mathbb{SOL}_d(\mathcal{A})$ the set $\{e \in [0,1] : E_e$ is a correct explanation for FLPAP $\mathcal{A}\}$ (the subscript d remembers us the definition is a declarative one)

4.2 Existence of explanations

Eiter and Gottlob have shown in [3] that deciding $\mathbb{SOL}_d(\mathcal{A}) \neq \emptyset$ is complete for complexity classes at the second level of polynomial hierarchy, while the use of priorisation raises the complexity to the third level in certain cases (for arbitrary propositional theories). We show that for definite FLPAP $\mathcal{A}$ are both the existence and priorisation (minimality) in the first level (NP).

Continuation of example. Having our motor vehicle example from 2.1. and our fuzzy program P_{mv} and the query high_fuel_consumption? by fuzzy logic program computation using first program rule we get

$$\min(0.8, \max(0, \text{rich\_mixture} + \text{low\_oil} - 1))$$

Now similarly as in the two valued logic abductive logic programming ([6]) our procedure instead of failing in a proof when a selected subgoal fail to unify with the head of any rule, the subgoal is viewed as a hypothesis, that is, if we know truth value for low_mixture and low_oil (from an explanation) we have the computed answer for high_fuel_consumption. To fulfill $OBS_{mv}(\text{high\_fuel\_consumption}) \geq 0.25$ it should be $\min(0.8, \max(0, \text{rich\_mixture} + \text{low\_oil} - 1)) \geq 0.25$ hence $\text{rich\_mixture} + \text{low\_oil} \geq 1.25$ □

So our fuzzy logic programming abduction should run as a usual logic program with two exceptions

• it successfully ends without deducing variables which are in the set of hypothesis

• it is prompted by a query with threshold, which can serve as a cut (as we will see later).

Definition 3 *(Procedural semantics for FLPAP) Let $\mathcal{A} =< P, OBS, H >$ be a FLPAP and let $\mathcal{G} = (G_0, G_1, \ldots, G_l)$ be a sequence of words in the alphabet of fuzzy logic program computation. We say that the sequence $\mathcal{G}$ is a* **successful abduction** *for $\mathcal{A}$ and $m \in OV$ if*
$G_0 = m$,
G_l contains only variables from H and
for all $i < l$, G_{i+1} is inferred from G_i by one of admissible rules and for the constantly one interpretation $I_1 : Var \longrightarrow \{1\}$ is the truth value $I_1(G_{i+1}) \geq OBS(m)$ (this condition is to be understood as a cut, because it estimates the best possible computation)

For $H = \{h_1, \ldots, h_n\}$, the expression G_l can be understood as a function of n real variables $G_l : [0,1]^n \longrightarrow [0,1]$ and that is why we can denote it by $G_l = f_{\mathcal{G}}(h_1, \ldots, h_n)$.

Theorem 4 *(Existence od solutions) Let $\mathcal{A} =< P, OBS, H >$ be a FLPAP and for each $m \in OV$ there is a successful abduction for $\mathcal{A}$ and m. Then $\mathbb{SOL}_d(\mathcal{A}) \neq \emptyset$.*

Proof: Because of our cut, which in each step checks whether substituting truth value 1 still keeps $I(G^m_{i+1}) \geq OBS(m)$, especially for the last step, hence for all $m \in OV$ is $f_{\mathcal{G}^m}(1, \ldots, 1) \geq OBS(m)$ □

4.3 Correct explanations versus computed explanations

Our definition of abduction gives us the possibility to define computed explanations for FLPAP $\mathcal{A}$.

Definition 5 *A tuple $e = (e_1, \ldots, e_n)$ is a* **computed explanation** *for a FLPAP $\mathcal{A} =< P, OBS, H >$ if for every $m \in OV$ there is a successful abduction $\mathcal{G}^m$ for $\mathcal{A}$ and m such that $f_{\mathcal{G}^m}(e_1, \ldots, e_n) \geq OBS(m)$ The set of all computed explanations will be denoted by $\mathbb{SOL}_p(\mathcal{A})$.*

Subscript p resembles us to procedural character of this definition

Example continued. In our motor vehicle example we calculated in 4.2 that first rule of P_{mv} gives rich_mixture + low_oil ≥ 1.25, similarly second rule gives low_oil ≥ 0.5 and the third gives rich_mixture ≥ 0.625, hence the set (coordinates are ordered as (rich_mixture, low_oil, low_water))

$$\{(e_1, e_2, e_3) \in [0,1] : \quad \begin{aligned} & e_1 + e_2 \geq 1.25 \text{ and} \\ & e_1 \geq 0.625 \text{ and} \\ & e_2 \geq 0.5\} \end{aligned}$$

is a subset of $\mathbb{SOL}_p(\mathcal{A})$ □

Theorem 6 *(Soundness of fuzzy abduction) Assume $\mathcal{A}$ is a definite FLPAP, then $\mathbb{SOL}_p(\mathcal{A}) \subseteq \mathbb{SOL}_d(\mathcal{A})$ (that is, every computed explanation for $\mathcal{A}$ is also a correct explanation)*

Proof: Our inverse usage of modus ponens is sound and so is our threshold cut (because of monotonicity of t-norms and conorms). So the result follows by induction through the length of abduction. □

In the next completeness theorem we need the assumption that our logical program has a finite computational tree - according to abductions. This is in practical applications of abduction very often the case, because e.g. observations should not be explained by themselves (i.e. there is no recursion on rules defining them), and most of logic programs for abduction are layered. Moreover if t-norms are archimedian (also very often) then the iteration of t-norms ends below the observation value threshold, and hence is cut.

Theorem 7 *(Completeness of fuzzy abduction) Assume $\mathcal{A}$ is a definite FLPAP and the logical program has a finite computational tree according to abductions, then $\mathbb{SOL}_d(\mathcal{A}) \subseteq \mathbb{SOL}_p(\mathcal{A})$ (that is, every correct explanation for $\mathcal{A}$ is also a computed explanation)*

Proof: Let $e = (e_1, \dots, e_n)$ be a correct explanation for FLPAP $\mathcal{A} =< P, OBS, H >$. Recall that it uniquely determines an explanation $E_e : H \longrightarrow [0,1]$. Consider the fuzzy logic program P_e which consists of rules of the program P and of facts $h_i; e_i$. for all i. The fact that e is a correct explanation for $\mathcal{A}$ means exactly that for all $m \in OV$ is $OBS(m)$ a correct answer for the program P_e and query m?. Hence (by approximative completeness from [9]) for every $\epsilon > 0$ there is a successful prolog computation $\mathcal{G}^m_\epsilon$ such that the computed answer

$$q^m_\epsilon = f_{\mathcal{G}^m_\epsilon}(e_1, \dots, e_n) \geq OBS(m) - \epsilon.$$

Choose an infinite sequence $\epsilon_r \longrightarrow 0^+$. As far as there are only finitely many computations, there is between all $\mathcal{G}^m_{\epsilon_r}$'s a single computation $\mathcal{G}^m_\delta$ which does the job for all but finitely many ϵ_r's, hence

$$q_\delta = f_{\mathcal{G}^m_\delta}(e_1, \dots, e_n) \geq OBS(m) - \epsilon_r.$$

for all but finitely many r's hence $q_\delta \geq OBS(m)$. Hence if $OV = \{m_1, \dots, m_k\}$ then for every $i < k$ there is a computation $\mathcal{G}^k$ such that

$$q_k = f_{\mathcal{G}^k}(e_1, \dots, e_n) \geq OBS(m),$$

so for every k is $\mathcal{G}^k$ a witness for q_k being a computed answer for P_e and m_k. Hence e is a computed explanation. □
From now on we do not have to distinguish between two sets of solutions and we simply denote it $\mathbb{SOL}(\mathcal{A})$.

One can ask how difficult is to decide whether $e \in \mathbb{SOL}(\mathcal{A})$ or not. Now we see that it substantially depends on the complexity of logic programming computation and the complexity of functions evaluating the truth values for connectives and evaluating t-norms. So from a computational point of view, it makes sense to use simple connectives (e.g. linear in each coordinate (as product is) or even partly constant).

4.4 Cheapest explanation via linear programming

In real world applications one can be interested in cheapest/minimal solutions of the abduction problem.

Example continued. Continuing our example on motor vehicles assume, that to check (and fix) an explanation $(e_1, e_2, e_3) \in \mathbb{SOL}(\mathcal{A}_{mv})$ costs $2e_1 + e_2 + 0.1e_3$. The space of solutions $\mathbb{SOL}(\mathcal{A}_{mv})$ is bounded by linear surfaces in $[0,1]^3$ and is union of 4 convex bodies.

One of them we get using first three rules of the program P_{mv}. Applying a linear programming method for minimal solution to the set

$$\{(e_1, e_2, e_3) \in [0,1] : \quad \begin{array}{l} e_1 + e_2 \geq 1.25 \text{ and} \\ e_1 \geq 0.625 \text{ and} \\ e_2 \geq 0.5\} \end{array}$$

wrt our cost function we get in this convex body a minimal solution solution $(0.625, 0.625, 0)$ at cost of 1.875.

The cheapest solution of FLPAP $\mathcal{A}_{mv}$ is $e_{min} = (0.25, 1, 0.35)$ at cost of 1.535 (we get it from the convex body of all solutions to first, fourth and fifth program rule). □

As linear programming is lying in NP complexity class (even much lower) as prolog does, to find minimal solutions for a definite FLPAP (assuming connectives and t-norms are coordinatewise linear) does not increase the complexity and remains in NP.

4.5 Finite approximations

In many real world applications we work with approximations of data and measurements are interesting up to some precision. We can show that our results work also for connectives which are partly constant lower semicontinuous functions (we loose associativity of connectives), our fuzzy logic is finitely valued logic, fuzzy abduction is easy to implement and finding cheapest (minimal) solutions restricts to a problem of integer programming.

REFERENCES

[1] D. Dubois, J. Lang, H. Prade. Fuzzy sets in approximative reasoning, part2, Logical approaches. Fuzzy Sets and Systems 40 (1991) 203–244

[2] L. Console, D. Thesier Dupre, P. Torasso. On the relationship between abduction and deduction. J. Logic Comput. 1 (1991) 661–690

[3] T. Eiter, G. Gottlob. The complexity of logic based abduction. J. ACM 42(1995)3–42

[4] S. Gottwald. Fuzzy sets and fuzzy logic. Vieweg, Berlin, 1993

[5] P. Hájek. Metamathematics of fuzzy logic. Kluwer, Dodrecht, 1998

[6] A. C. Kakas, R. A. Kowalski, F. Toni. Abductive logic programming. J. Logic Computat. 2(1992)719–770

[7] R. Mesiar. Fuzzy sets and probability theory. Tatra Mt. Math. Publ. 1 (1992) 105–129

[8] C. S. Peirce Abduction and induction. In Philosophical writings of Peirce, chapter 11, J. Buchler ed. Dover, New York 1955

[9] P. Vojtáš. Fuzzy reasoning with tunable t-operators. J. Adv. Comp. Intelligence 2(1998)121–127