Set Functors and Generalised Terms

P. Eklund[*], **M. Ángeles Galán**[†]
Umeå University
Department of Computing Science
SE-901 87 Umeå, Sweden
{peklund,magalan}@cs.umu.se

M. Ojeda-Aciego[‡], **A. Valverde**[‡]
University of Málaga
Department of Applied Mathematics
E-29080 Málaga, Spain
{aciego,a_valverde}@ctima.uma.es

Abstract

In this paper we use techniques for monad compositions in order to provide a basis for categorical unification in the framework of generalised terms. In particular, we provide results for many-valued sets of terms, and show that this composition of set functors can be extended to a monad.

1 Introduction

Several heuristic approaches have been suggested to extend many-valued logic programming. However, the lack of a foundational base, is an obstacle for a wider acceptance of these models, and further, formal approaches typically build upon conventional terms. Restricting to finitely many truth values, using the framework suggested in [12], a many-valued predicate calculus was proposed in [9].

This paper is motivated by the use of categorical methods in many-valued logic programming. In particular, we generalise terms using compositions of monads.

We compose set functors with the term monad and show how such compositions can be extended to monads. Our motivation of using monads is given by the situation in the classical case where most general unifiers are coequalisers in the Kleisli category associated with the term monad [13].

[*]Corresponding author.

[†]Supported by the Swedish Research Council for Engineering Sciences.

[‡]Partially supported by the Spanish CICYT project TIC97-0579-C02-02.

For notations and results within category theory and universal algebra, we refer to [1, 2, 10, 11]. For a more detailed treatment of set functors used in this paper, also including many-valued sets, we refer to [4, 5, 6]. For a survey of many-valued logic, see e.g. [8].

2 Monads and Kleisli categories

A monad can be seen as the abstraction of the concept of adjoint functors and in a sense an abstraction of universal algebra. It is interesting to note that monads are useful not only in universal algebra, but it is also an important tool in topology when handling regularity, iteratedness and compactifications, and also in the study of toposes and related topics.

Let $\mathtt{C}$ be a category. A monad (or triple, or algebraic theory) over $\mathtt{C}$ is written as $\Phi = (\Phi, \eta, \mu)$, where $\Phi : \mathtt{C} \to \mathtt{C}$ is a (covariant) functor, and $\eta : id \to \Phi$ and $\mu : \Phi \circ \Phi \to \Phi$ are natural transformations for which $\mu \circ \Phi\mu = \mu \circ \mu\Phi$ and $\mu \circ \Phi\eta = \mu \circ \eta\Phi = id_\Phi$ hold. A monad defined in this way is said to be in monoid form.

Note that, for a natural transformation ξ, $(\xi\Phi)_X = \xi_{\Phi X}$ and $(\Phi\xi)_X = \Phi\xi_X$. It is useful to write η^Φ and μ^Φ if we need to distinguish between natural transformations in different monads.

A Kleisli category $\mathtt{C}_\Phi$ for a monad Φ over a category $\mathtt{C}$ is defined as follows: Objects in $\mathtt{C}_\Phi$ are the same as in $\mathtt{C}$, and the morphisms are defined as $hom_{\mathtt{C}_\Phi}(X, Y) = hom_{\mathtt{C}}(X, \Phi Y)$, that is morphisms $f \colon X \rightharpoonup Y$ in $\mathtt{C}_\Phi$ are simply morphisms $f \colon X \to \Phi Y$ in $\mathtt{C}$, with $\eta_X^\Phi \colon X \to \Phi X$ being the identity morphism.

Composition of morphisms is defined as

$$(X \xrightharpoonup{f} Y) \circ (Y \xrightharpoonup{g} Z) = X \xrightarrow{\mu^\Phi_Z \circ \Phi g \circ f} \Phi Z.$$

The Kleisli category is equivalent to the full subcategory of free Φ-algebras of the monad, and its definition makes it clear that the arrows are substitutions. Indeed, the categorical unification algorithm in [13] is based on the Kleisli category of the term monad.

A monad (Φ, η, μ) written as $(\Phi, \eta, \circ)$, where $\circ$ is the composition of morphisms in the corresponding Kleisli category, is said to be a monad in clone form. In fact, there is a one-to-one correspondence between monads, respectively, in monoid and clone forms [11].

3 Set functors

3.1 The power-set monad

Let L be a completely distributive lattice. For $L = \{0, 1\}$ we write $L = 2$. The covariant power-set functor L_{id} is obtained by $L_{id}X = L^X$, i.e. the set of mappings (or L-fuzzy sets) $A : X \to L$, and following [7], for a morphism $f\colon X \to Y$ in $\mathtt{Set}$, by defining

$$\begin{aligned} L_{id}f(A)(y) &= \bigvee_{x \in X} A(x) \wedge f^{-1}(\{y\})(x) \\ &= \bigvee_{f(x)=y} A(x). \end{aligned}$$

Further, define $\eta_X : X \to L_{id}X$ by

$$\eta_X(x)(x') = \begin{cases} 1 & \text{if } x = x' \\ 0 & \text{otherwise} \end{cases}$$

and $\mu_X\colon L_{id}L_{id}X \to L_{id}X$ by

$$\mu_X(\mathcal{A})(x) = \bigvee_{A \in L_{id}X} A(x) \wedge \mathcal{A}(A).$$

Proposition 3.1 ([11]) $\mathbf{L}_{id} = (L_{id}, \eta, \mu)$ *is a monad.*

Note that $\mathbf{2}_{id}$ is the usual covariant power-set monad $\mathbf{P} = (P, \eta, \mu)$, where PX is the set of subsets of X, $\eta_X(x) = \{x\}$ and $\mu_X(\mathcal{B}) = \bigcup \mathcal{B}$.

The problem of extending a functor to a monad is not a trivial one, and some strange situations may well arise as shown below. Note that the id^2 functor can be extended to a monad with $\eta_X(x) = (x, x)$ and $\mu_X((x_1, x_2), (x_3, x_4)) = (x_1, x_4)$. Similarly, id^n can be extended to a monad. In addition, the proper power-set functor P_0, where $P_0X = PX \setminus \{\emptyset\}$, as well as $id^2 \circ P_0$ can, respectively, be extended to a monad in a unique way. However, $P_0 \circ id^2$ cannot be made to a monad [4].

Remark 3.1 *Let* $\Phi = (\Phi, \eta^\Phi, \mu^\Phi)$ *and* $\Psi = (\Psi, \eta^\Psi, \mu^\Psi)$ *be monads over* $\mathtt{Set}$. *The composition* $\Phi \circ \Psi$ *cannot always be extended to a monad as we see in the case of* $P_0 \circ id^2$.

3.2 The term monad

It is useful to adopt a more functorial presentation of the set of terms, as opposed to using the conventional inductive definition of terms, where we bind ourselves to certain styles of proofs. Even if a purely functorial presentation might seem complicated, there are advantages when we define corresponding monads, and, further, a functorial presentation simplifies efforts to prove results concerning compositions of monads. Notations follow [6], which were adopted also in [4].

For a set A, the constant set functor $A_{\mathtt{Set}}$ is the covariant set functor which assigns sets X to A, and mappings f to the identity map id_A. The sum $\sum_{i \in I} \varphi_i$ of covariant set functors φ_i assigns to each set X the disjoint union $\bigcup_{i \in I}(\{i\} \times \varphi_i X)$, and to each morphism $X \xrightarrow{f} Y$ in $\mathtt{Set}$ the mapping $(i, m) \mapsto (i, \varphi_i f(m))$, where $(i, m) \in (\sum_{i \in I} \varphi_i)X$.

Let k be a cardinal number and $(\Omega_n)_{n \le k}$ be a family of sets. We will write $\Omega_n id^n$ instead of $(\Omega_n)_{\mathtt{Set}} \times id^n$. Note that $\sum_{n \le k} \Omega_n id^n X$ is the set of all triples $(n, \omega, (x_i)_{i \le n})$ with $n \le k$, $\omega \in \Omega_n$ and $(x_i)_{i \le n} \in X^n$.

A disjoint union $\Omega = \bigcup_{n \le k}\{n\} \times \Omega_n$ is an operator domain, and an Ω-algebra is a pair $(X, (s_{n\omega})_{(n,\omega) \in \Omega})$ where $s_{n\omega} : X^n \to X$ are n-ary operations. The $\sum_{n \le k} \Omega_n id^n$-morphisms between Ω-algebras are precisely the homomorphisms between the algebras.

The term functor can now be defined by transfinite induction. In fact, let $T_\Omega^0 = id$ and define

$$T_\Omega^\alpha = (\sum_{n\le k}\Omega_n id^n)\circ \bigcup_{\beta<\alpha} T_\Omega^\beta$$

for each positive ordinal α. Finally, let

$$T_\Omega = \bigcup_{\alpha<\bar{k}} T_\Omega^\alpha$$

where $\bar{k}$ is the least cardinal greater than k and $\aleph_0$. Clearly, $(n,\omega,(m_i)_{i\le n}) \in T_\Omega^\alpha X$, $\alpha \neq 0$, implies $m_i \in T_\Omega^{\beta_i}X$, $\beta_i < \alpha$.

Note that $(T_\Omega X, (\sigma_{n\omega})_{(n,\omega)\in\Omega})$ is an Ω-algebra, if we define $\sigma_{n\omega}((m_i)_{i\le n}) = (n,\omega,(m_i)_{i\le n})$ for $(n,\omega)\in\Omega$ and $m_i \in T_\Omega X$. Actually, this algebra is a freely generated algebra in the category of Ω-algebras, that is, for an Ω-algebra $B = (Y,(t_{n\omega})_{(n,\omega)\in\Omega})$, a morphism $X \xrightarrow{f} Y$ in $\mathtt{Set}$ can be extended by transfinite induction to a Ω-homomorphism

$$(T_\Omega X, (\sigma_{n\omega})_{(n,\omega)\in\Omega}) \xrightarrow{f^\star} (Y,(t_{n\omega})_{(n,\omega)\in\Omega})$$

called the Ω-extension of f associated to B, by

$$\begin{aligned} f^\star_{|T_\Omega^0 X} &= f \text{ for the base case, and}\\ f^\star(n,\omega,(m_i)_{i\le n}) &= t_{n\omega}((f^\star(m_i))_{i\le n}) \end{aligned}$$

for each positive ordinal α satisfying $\alpha < \bar{k}$ and $(n,\omega,(m_i)_{i\le n}) \in T_\Omega^\alpha X$.

A morphism $X \xrightarrow{f} Y$ in $\mathtt{Set}$ can also be extended to the corresponding Ω-homomorphism

$$(T_\Omega X, (\sigma_{n\omega})_{(n,\omega)\in\Omega}) \xrightarrow{T_\Omega f} (T_\Omega Y, (\tau_{n\omega})_{(n,\omega)\in\Omega}),$$

where $T_\Omega f$ is defined to be the Ω-extension of $X \xrightarrow{f} Y \hookrightarrow T_\Omega Y$ associated to $(T_\Omega Y, (\tau_{n\omega})_{(n,\omega)\in\Omega})$.

Remark 3.2 ([6]) *T_Ω is the least covariant set functor which has id and $(\sum_{n\le k}\Omega_n id^n)\circ T_\Omega$ as subfunctors.*

We can now extend T_Ω to a monad. Define $\eta_X^{T_\Omega}(x) = x$. Further, let $\mu_X^{T_\Omega} = id^\star_{T_\Omega X}$ be the Ω-extension of $id_{T_\Omega X}$ with respect to $(T_\Omega X, (\sigma_{n\omega})_{(n,\omega)\in\Omega})$.

Proposition 3.2 ([11]) *$\mathbf{T}_\Omega = (T_\Omega, \eta^{T_\Omega}, \mu^{T_\Omega})$ is a monad.*

4 Composition of monads

In the following we will show how the composition $L_{id}\circ T_\Omega$ can be extended to a monad. We will here write 2 and L instead of 2_{id} and L_{id}, and T instead of T_Ω. Our constructions will make use of the mapping $\sigma_X^L : TLX \to LTX$ defined as follows (note that we will also write σ instead of σ^L for brevity):

For the base case $\sigma_{X|T^0LX} = id_{LX}$. Further, for $l = (n,\omega,(l_i)_{i\le n}) \in T^\alpha LX$, $\alpha > 0$, $l_i \in T^{\beta_i}LX$, $\beta_i < \alpha$, let

$$\sigma_X(l)((n',\omega',(m_i)_{i\le n})) = \\ = \begin{cases} \bigwedge_{i\le n} \sigma_X(l_i)(m_i) & \text{if } n = n' \text{ and } \omega = \omega' \\ 0 & \text{otherwise} \end{cases}$$

Note that in the case of $\alpha > 0$, for $L = 2$ we have

$$\sigma_X(l) = \{(n,\omega,(m_i)_{i\le n}) \mid m_i \in \sigma_X(l_i)\}.$$

Note also that, for $l \in TLX$ and $m \in TX$ we have $\sigma_X(l)(m) = 0$, if $l \in T^\alpha LX$ and $m \notin T^\alpha X$, or if $l \notin T^\alpha LX$ and $m \in T^\alpha X$.

Lemma 4.1 *$\sigma : T\circ L \to L\circ T$ is a natural transformation.*

Proof: For any $l \in TLX$, and any $X \xrightarrow{f} Y$ in $\mathtt{Set}$, we need to show that $LTf \circ \sigma_X(l) = \sigma_Y \circ TLf(l)$. For $l \in T^0LX$, this is immediate. For $\alpha > 0$, we may write $l = (n,\omega,(l_i)_{i\le n})$, where $l_i \in T^{\beta_i}LX$, $\beta_i < \alpha$, for all $i \le n$. Let now $\overline{m} = (n,\omega,(\overline{m}_i)_{i\le n}) \in TY$. Then,

$$LTf(\sigma_X(l))(\overline{m}) = \\ = \bigvee_{Tf((n,\omega,(m_i)_{i\le n}))=\overline{m}} \bigwedge_{i\le n} \sigma_X(l_i)(m_i).$$

Further, by induction, we get

$$\begin{aligned} \sigma_Y(TLf(l))(\overline{m}) &= \bigwedge_{i\le n} \sigma_Y(TLf(l_i))(\overline{m}_i) \\ &= \bigwedge_{i\le n} LTf(\sigma_X(l_i))(\overline{m}_i) \\ &= \bigwedge_{i\le n} \bigvee_{Tf(m_i)=\overline{m}_i} \sigma_X(l_i)(m_i). \end{aligned}$$

By complete distributivity of L, we then obtain naturality of σ. ■

We will use this natural transformation σ in order to define the natural transformations η^{LT} and μ^{LT}, which provide LT with the structure of a monad.

Definition 4.1 *The natural transformations $\eta^{LT}\colon id \to LT$ and $\mu^{LT}\colon LTLT \to LT$ are defined as follows*

$$\eta^{LT} = \eta^L T \circ \eta^T \quad \mu_X^{LT} = L\mu_X^T \circ \mu_{TTX}^L \circ L\sigma_{TX}$$

Note that $\eta_X^{LT}(x) = \eta_{TX}^L(x)$, and in the case of $L = 2$ then $\eta_X^{2T}(x) = \{x\}$. Further for $R \in LT^\alpha LTX$, $\alpha > 0$, and $m \in TX$, note that

$$\mu_X^{LT}(R)(m) = \bigvee_{r \in TLTX} R(r) \wedge \sigma_{TX}(r)(m).$$

and also note that in the case $L = 2$, for $R = \{(n_j, \omega_j, (r_{ij})_{i \leq n_j}) \mid j \in J\} \in 2T^\alpha 2TX$, $\alpha > 0$, we have

$$\mu_X^{2T}(R) = \\ = \{(n_j, \omega_j, (m_{ij})_{i \leq n_j}) \mid j \in J, m_{ij} \in \sigma_{TX}(r_{ij})\}$$

The following technical lemma gives some conditions which guarantee the monad structure for the composition LT.

Lemma 4.2 *The following properties hold:*

(i) $\sigma_{TX} \circ T\eta_X^{LT} = \eta_{TTX}^L \circ \eta_{TX}^T$,

(ii) $L\mu_X^T \circ \sigma_{TX} \circ T\mu_X^{LT} = \mu_X^{LT} \circ L\mu_{LTX}^T \circ \sigma_{TLTX}$,

(iii) $\sigma_X \circ \eta_{LX}^T = L\eta_X^T$.

Note that (i) and (ii) in the case of $L = 2$ become

(i') $\sigma_{TX}(T\eta_X^{2T}(m)) = \{m\}$, for all $m \in TX$,

(ii') $2\mu_X^T \circ \sigma_{TX} \circ T\mu_X^{2T}(d) = \bigcup_{R \in \sigma_{T2TX}(d)} \sigma_{TX}(R)$.

Proof: (i) This holds trivially for $\alpha = 0$. In case of $\alpha > 0$, for $m = (n, \omega, (m_i)_{i \leq n}) \in TX$ and $m' = (n, \omega, (m'_i)_{i \leq n}) \in TX$, by induction, we get

$$\begin{aligned}
&\sigma_{TX}(T\eta_X^{LT}(m))(m') = \\
&= \sigma_{TX}((n, \omega, (T\eta_X^{LT}(m_i))_{i \leq n}))(m') \\
&= \bigwedge_{i \leq n} \sigma_{TX}(T\eta_X^{LT}(m_i))(m'_i) \\
&= \bigwedge_{i \leq n} \eta_{TX}^L(m_i)(m'_i).
\end{aligned}$$

Since $m = m'$ if and only if $m_i = m'_i$ for all $i \leq n$, we immediately get

$$\eta_{TX}^L(m)(m') = \bigwedge_{i \leq n} \eta_{TX}^L(m_i)(m'_i).$$

(ii) Again this holds trivially for $\alpha = 0$. In case of $\alpha > 0$, let $m = (n, \omega, (m_i)_{i \leq n}) \in TX$ and $d = (n, \omega, (d_i)_{i \leq n}) \in TLTLTX$. By induction and complete distributivity of L we then have

$$\begin{aligned}
&\sigma_{TX}(T\mu_X^{LT}(d))(m) = \bigwedge_{i \leq n} \sigma_{TX}(T\mu_X^{LT}(d_i))(m_i) = \\
&= \bigwedge_{i \leq n} \mu_X^{LT}(\sigma_{TLTX}(d_i))(m_i) \\
&= \bigwedge_{i \leq n} \bigvee_{r \in TLTX} \sigma_{TLTX}(d_i)(r) \wedge \sigma_{TX}(r)(m_i) \\
&= \bigvee_{(n,\omega,(r_i)) \in TLTX} \bigwedge_{i \leq n} \sigma_{TLTX}(d_i)(r_i) \wedge \sigma_{TX}(r_i)(m_i) \\
&= \bigvee_{r \in TLTX} \sigma_{TLTX}(d)(r) \wedge \sigma_{TX}(r)(m) \\
&= \mu_X^{LT}(\sigma_{TLTX}(d))(m).
\end{aligned}$$

(iii) By definition, as $\sigma_{X|T^0LX} = id_{LX}$. ■

Proposition 4.1 $(L_{id} \circ T_\Omega, \eta^{L_{id} \circ T_\Omega}, \mu^{L_{id} \circ T_\Omega})$*, denoted $\mathbf{L}_{id} \bullet \mathbf{T}_\Omega$, is a monad.*

Proof: We have

$$\begin{aligned}
&\mu_X^{LT} \circ LT\eta_X^{LT} = \\
&= L\mu_X^T \circ \mu_{TTX}^L \circ L\sigma_{TX} \circ LT\eta_X^{LT} \\
&= L\mu_X^T \circ \mu_{TTX}^L \circ L\eta_{TTX}^L \circ L\eta_{TX}^T \\
&= L\mu_X^T \circ L\eta_{TX}^T = id_{LTX}
\end{aligned}$$

and

$$\begin{aligned}
&\mu_X^{LT} \circ \eta_{LTX}^{LT} = \\
&= L\mu_X^T \circ \mu_{TTX}^L \circ L\sigma_{TX} \circ \eta_{TLTX}^L \circ \eta_{LTX}^T \\
&= L\mu_X^T \circ \mu_{TTX}^L \circ \eta_{LTTX}^L \circ \sigma_{TX} \circ \eta_{LTX}^T \\
&= L\mu_X^T \circ \sigma_{TX} \circ \eta_{LTX}^T \\
&= L\mu_X^T \circ L\eta_{TX}^T = Lid_{TX} = id_{LTX}
\end{aligned}$$

Further we have the associativity of μ^{LT}

$$\begin{aligned}
&\mu_X^{LT} \circ LT\mu_X^{LT} = \\
&= L\mu_X^T \circ \mu_{TTX}^L \circ L\sigma_{TX} \circ LT\mu_X^{LT} \\
&= \mu_{TX}^L \circ LL\mu_X^T \circ L\sigma_{TX} \circ LT\mu_X^{LT} \\
&= \mu_{TX}^L \circ L\mu_X^{LT} \circ LL\mu_{LTX}^T \circ L\sigma_{TLTX} \\
&= \mu_{TX}^L \circ LL\mu_X^T \circ L\mu_{TTX}^L \circ \\
&\qquad \circ LL\sigma_{TX} \circ LL\mu_{LTX}^T \circ L\sigma_{TLTX} \\
&= L\mu_X^T \circ \mu_{TTX}^L \circ L\mu_{TTX}^L \circ \\
&\qquad \circ LL\sigma_{TX} \circ LL\mu_{LTX}^T \circ L\sigma_{TLTX} \\
&= L\mu_X^T \circ \mu_{TTX}^L \circ \mu_{LTTX}^L \circ \\
&\qquad \circ LL\sigma_{TX} \circ LL\mu_{LTX}^T \circ L\sigma_{TLTX} \\
&= L\mu_X^T \circ \mu_{TTX}^L \circ L\sigma_{TX} \circ \\
&\qquad \circ \mu_{TLTX}^L \circ LL\mu_{LTX}^T \circ L\sigma_{TLTX} \\
&= L\mu_X^T \circ \mu_{TTX}^L \circ L\sigma_{TX} \circ \\
&\qquad \circ L\mu_{LTX}^T \circ \mu_{TTLTX}^L \circ L\sigma_{TLTX} \\
&= \mu_X^{LT} \circ L\mu_{LTX}^T \circ \mu_{TTLTX}^L \circ L\sigma_{TLTX} \\
&= \mu_X^{LT} \circ \mu_{LTX}^{LT}
\end{aligned}$$

■

Note that in the proof of the proposition above, the actual definition of the functors L and T has not been used, only universal properties and those stated in Lemma 4.2. We have actually proved the following theorem

Theorem 4.1 *Let $\Phi = (\Phi, \eta^\Phi, \mu^\Phi)$ and $\Psi = (\Psi, \eta^\Psi, \mu^\Psi)$ be monads and let $\sigma : \Psi \circ \Phi \to \Phi \circ \Psi$ be a natural transformation such that the following properties hold:*

(i) $\sigma_{\Psi X} \circ \Psi\eta_X^{\Phi\Psi} = \eta_{\Psi\Psi X}^{\Phi} \circ \eta_{\Psi X}^{\Psi}$,

(ii) $\Phi\mu_X^\Psi \circ \sigma_{\Psi X} \circ \Psi\mu_X^{\Phi\Psi} = \mu_X^{\Phi\Psi} \circ \Phi\mu_{\Phi\Psi X}^{\Psi} \circ \sigma_{\Psi\Phi\Psi X}$.

(iii) $\sigma_X \circ \eta_{\Phi X}^\Psi = \Phi\eta_X^\Psi$

Then $\Phi \bullet \Psi = (\Phi \circ \Psi, \eta^\Phi\Psi \circ \eta^\Psi, \Phi\mu^\Psi \circ \mu^\Phi\Psi\Psi \circ \Phi\sigma\Psi)$ is a monad.

5 Conclusions and further work

We have seen how set functors can be composed to providing monads. It is important to study more examples, i.e. including various types of double power-set monads. Further, it is interesting to investigate techniques for constructing new monads from given ones.

Within the scope of many-valued logic programming, it is important to further investigate the possibilities of using categorical approaches to unification, with variable substitutions for generalised terms being morphisms in $\mathtt{Set}_{\Phi\bullet\mathbf{T}_\Omega}$, i.e. variable substitutions are morphisms $X \xrightarrow{f} \Phi TY$ in `Set`. It is expected that this approach provides an appropriate formal framework for useful developments of generalised term based many-valued logic programming.

Acknowledgements

We are grateful to Werner Gähler and Aleš Pultr for discussions and helpful suggestions.

References

[1] J Adámek, H Herrlich, G Strecker, *Abstract and Concrete Categories*, John Wiley & Sons, 1990.

[2] M Barr, C Wells, *Toposes, Triples and Theories*, Springer, 1985.

[3] P Eklund, W Gähler, *Generalized Cauchy spaces*, Math. Nachr. **147** (1990), 219-233.

[4] P Eklund, W Gähler, *Fuzzy filter functors and convergence*, Applications of category theory to fuzzy subsets. (S E Rodabaugh, et al ed.), Theory and Decision Library B, Kluwer, 1992, 109-136.

[5] W Gähler, *A topological approach to structure theory*, Math. Nachr. **100** (1981), 93-144.

[6] W Gähler, *Monads and convergence*, Proc. Conference Generalized Functions, Convergences Structures, and Their Applications, Dubrovnik (Yugoslavia) 1987, Plenum Press, 1988, 29-46.

[7] J A Goguen, *L-fuzzy sets*, J. Math. Anal. Appl. **18** (1967), 145-174.

[8] P Hájek, *Metamathematics of Fuzzy Logic*, Kluwer Academic Publishers, 1998.

[9] F Klawonn, R Kruse, *A Łukasiewicz logic based Prolog*, Mathware & Soft Comp. **1** (1994), 5-29.

[10] J Loeckx, H-D Ehrich, M. Wolf, *Specification of Abstract Data Types*, John Wiley & Sons, 1996.

[11] E G Manes, *Algebraic Theories*, Springer, 1976.

[12] J Pavelka, *On fuzzy logic I–III*, Zeitschr. Math. Logik Grundl. Math. **25** (1979), 45-52, 119-134, 447-464.

[13] D E Rydeheard, R M Burstall, *A categorical unification algorithm*, Proc. Category Theory and Computer Programming, LNCS 240, Springer, 1986, 493-505.