Mathematical Institute
Slovak Academy of Sciences, Bratislava

Preprint 24/1996

Peter Vojtáš

Uncertain reasoning with floating connectives

August 7, 1996

PREPRINT SERIES

Uncertain reasoning with floating connectives

Peter Vojtáš[1,2]

[1] Department of Computer Science, College of Science, P. J. Šafárik University, Jesenná 5, 041 54 Košice, Slovakia
[2] Mathematical Institute, Slovak Academy of Sciences,* Jesenná 5, 041 54 Košice, Slovakia

Abstract. We consider a mathematical model for uncertain reasoning based on many-valued logic and logic programming. To handle some aspects of expert systems we discuss disjunctive logic programs (without negation). In order to fit our programs with the knowledge base we investigate logical systems with arbitrary (tunable/floating) fuzzy conjunctions (t-norms). We show the completeness of our semantics.

Key words: many-valued logic, expert systems, uncertainty, logic programming, t-norms

1 Motivation

Our motivation arises from expert systems (or high level knowledge base, database information technologies). In most systems the knowledge and inference are separated. Our main interest is the inference mechanism (often dubbed "deduction engine")

We choose an approach undertaken by D.Meritt ([Mr]), L.Sterling and E.Shapiro ([SS]) and by S.Ceri, G.Gottlob, L.Tanca ([CGT]). They show that Prolog and/or logic programming is a natural language for building expert systems.

There is yet another feature which we would like to stress in our motivation - the uncertainty. The expert's rules might be vague and the user might be unsure of answers to questions. This can be easily seen in medical diagnostic systems where the expert is not able to be definite about the relationship between symptoms and diseases. In fact, the doctor might offer multiple diagnoses ([Mr]). We consider also the case where there is more than one rule (i.e. more than one diagnostical procedure) which supports a given conclusion (diagnosis, therapy selection). We will sometimes couple together such procedures by a disjunction in the body of rules (disjunctive logic programming). In real systems this can increase the confidence of the final answer (see also N.J.Nilsson [Ni]).

Methods of handling uncertainty fall into two broad classes

* e-mail: vojtas@kosice.upjs.sk, tad/fax (+42 95)62 28 29 1
This work was supported by the grant 2/1224/96 of the Slovak Grant Agency for Science.

- those offering quantitative numerical representation of degrees of certainty in some way and

- those modeling uncertainty in a purely descriptive or qualitative way.

We carry out the quantitative approach here. I. Graham in [Gr] wrote: "One point to bear in mind when selecting an uncertainty management method is the fact that there are several kinds of uncertainty. In particular we must be careful to distinguish between uncertainty arising from natural variation and that from conceptual apprehension. Natural variation give rise to notions such as probability and possibility which can be handled by statistical and fuzzy methods (many-valued logic) respectively and have quite different mathematical properties. Conceptual error leads to vagueness, variations in belief, degrees of truth (i.e.provability), variations in degree of permission, and so on. In many real applications there will be a need to model more than one kind of uncertainty. The difficulty with systems which only offer one method is precisely that the designer may be tempted to model vagueness with probability or stochasticity with fuzzy sets or both with certainty factor, all of which are highly likely to lead to dysfunctional systems". So far I.Graham in [Gr]. We try to face this challenge by considering our many-valued connectives to be arbitrary t-norms and to be "floating" in the sense, that during the life span of the system the connectives can change in order to fit the knowledge base (however we do not consider this many-valued machine learning procedure here). Of course we do not exclude the possibility of tuning our system by changing confidence factors of rules and facts. But in large systems with many interrelations this can lead to contradictory requirements. This can be overdone by tuning connectives.

D.Dubois, J.Lang and H.Prade in [DLP] require a fuzzy logic programming language should be based on a real logical system (we refer to them for extensive references on the research in fuzzy logic programming). Our approach covers from the mathematical point of view many previous results and is based on a mathematical model of many-valued logic. Classical many-valued logical systems of Lukasiewicz and Gödel (see S.Gottwald's book [Go]) are developing both syntactical and semantical part of the logical system and are oriented to prove completeness results for 1-tautologies. Our situation requires to handle cases where an information (=axiom=fact or rule in Prolog) has some degree of certainty (confidence factor) between 0 and 1. This corresponds to case when the set of axioms is no more a crisp set of formulas (propositions resp.) but rather a fuzzy set of formulas. We are enough lucky that there were already developed tools in logic to handle this phenomena, namely Pavelka's propositional logic [P], Novak's fuzzy predicate calculi [No] (extending Pavelka's ideas) and a substantial simplification of both in works of P.Hájek ([H]) where RPL-Rational Pavelka logic and RQL-Rational quantification logic are introduced and developed. But for our purpose we had to go even behind this systems because already MYCIN ([SB]) deduction puts together many-valued connectives which have never been studied in a single axiomatic system. In [VPL], [VP1] and [VP2] we have already undertaken a research in this direction. We will summarize it here to be selfcontained and we will push our research a little bit forward considering disjunctive logic programs with floating connectives.

So far, in our motivation, we came to the point that we will investigate many-valued logic with arbitrary (possibly floating) connectives. Hence we will not look for an axiomatic system, we rather concentrate on structures (models) of many-valued predicate calculi. As in many-valued logic $A \longrightarrow B$ does not always take same truth values as $\neg A \vee B$ our rules (clauses) will be (on syntactical level) implications and facts will be atoms. Our deduction engine will be backward modus pones (instead of handling clauses under resolution and refutation mechanism of classical logic programming).

2 Introduction

Following P.Hájek ([H]) we consider formulas labeled by a rational number (φ, q) as axioms. A model of (φ, q) is a (many-valued/fuzzy) structure of the language where the truth value of φ (possibly a real number) is greater than or equal to q. Following S.Gottwald ([Go]) we denote (as symbols of our alphabet) conjunctions by et, disjunctions by vel, implications by seq and negations by non (with possible indexes). The corresponding truth values computations are defined by functions (signed by a dot over the symbol)

$$et^{\bullet} : [0,1] \times [0,1] \longrightarrow [0,1]$$
$$vel^{\bullet} : [0,1] \times [0,1] \longrightarrow [0,1]$$
$$seq^{\bullet} : [0,1] \times [0,1] \longrightarrow [0,1]$$
$$non^{\bullet} : [0,1] \longrightarrow [0,1]$$

The basic requirement for our system is to have the many-valued modus pones sound and giving best possible answers. This leads to a requirement (according to S.Gottwald) that there should be a pair of implication seq and a conjunction et such that:

($\Phi 0$) $et^{\bullet}$ is a t-norm (see e.g.[Me]), i.e. $et^{\bullet}$ prolongates the classical two valued conjunction, is associative, commutative, and $et^{\bullet}(1,x) = x, et^{\bullet}(0,x) = 0$.

and moreover seq is coupled with et by following axioms of S.Gottwald ([Go])

($\Phi 1$) $v \leq w$ implies $seq^{\bullet}(u,v) \leq seq^{\bullet}(u,w)$

($\Phi 2$) $et^{\bullet}(u, seq^{\bullet}(u,v)) \leq v$

($\Phi 3$) $v \leq seq^{\bullet}(u, et^{\bullet}(u,v))$

Having such et and seq the modus ponens

$$from\ (B,x)\ and\ (B\ seq\ A, y)\ infer\ (A, et^{\bullet}(x,y))$$

is sound and complete (i.e.gives best possible answers). The t-norm evaluating the use of modus ponens with seq will be very often denoted by $et^{\bullet}_{seq}$ (et_{seq}

need not be in our language). Notice moreover (for us it will be very important) that for a t-norm $et^{\bullet}$ there is an seq fulfilling $\Phi 1-3$ if and only if $et^{\bullet}$ is lower semicontinuous (as a function of two real variables), see [Go]. The following are basic connectives:

The Łukasiewicz connectives
$et_L^{\bullet} = \max(0, x+y-1)$
$seq_L^{\bullet} = \min(1, 1-x+y)$
$vel_L^{\bullet} = \min(1, x+y)$

The Gödel intuitionistic connectives
$et_G^{\bullet} = \min(x, y)$
$seq_G^{\bullet} = y$ if $x > y$ else 1
$vel_G^{\bullet} = \max(x, y)$

The product logic
$et_P^{\bullet} = x.y$
$seq_P^{\bullet} = \min(1, \frac{y}{x})$
$vel_P^{\bullet} = x + y - xy$

So far, as we will deal with definite programs only, negations are superfluous now. Note that et_L is the smallest conjunction and et_G the largest, vel_L is the largest disjunction and vel_G the smallest. Tupples (seq_L, et_L), (seq_P, et_P), (seq_G, et_G) fulfill $\Phi 1-3$.

The following observation (although very easy) is important for our approach - we understand the whole scale of t-norms laying between extremal ones as the space where our tuning/floating can range (and we would like to build up a feeling for this scale in order to find heuristics for eventual learning process). Looking from a probabilistic point of view max(0,x+y-1) is the minimal measure of a intersection of events with probability x and y, independent events meet in a set with measure x.y and min(x,y) represents the intersection of overlapping events. In this sense (loosely speaking) we can understand the scale of all possible many-valued logics representing the scale of notions, for example

Lukasiewicz logic	Product logic	Göedel logic
pessimist	realist	optimist
minimal lost	moderate outcome	maximal profit
competition	tolerance	cooperation
contradictory	independent	affirmative
expert vs.layman	two laymen	two experts
enemies	both selfish	friends

Leaving our connectives floating between these extremes we can replace probabilistic methods by logical (we win the functionality of our truth values).

3 Example. (L.Sterling, E.Shapiro, [SS])

A typical MYCIN rule ([SB]) looks as follows

IF	the gram stain of the organism is gram negative,
	the morphology of the organism is rod,
	the aerobicity of the organism is anaerobic
THEN	there is suggestive evidence (0.5) that
	the identity of the organism is Bacteroides.

There are two aspects to the above rule. First, the heuristic identification of a bacteria based on its gram stain, morphology, and aerobicity. Second, there is a certainty factor attached to the rule (in our case it is 0.5). Sterling and Shapiro are arguing that these two aspects should be separate. They suggest: the uncertainty is best handled by the following enhanced meta-interpreter (see [SS, p. 318])

% solve(Goal,Certainty)$\leftarrow$ certainty is our confidence that Goal is true.
solve(true,1).
solve((A,B),C)$\leftarrow$ solve(A,C1), solve(B,C2), minimum(C1,C2,C).
% (This means that the conjunction *et* in the body of rules is the many-valued
% Gödel's intuitionistic conjunction $\mathrm{et}^{\bullet}_{G}(x, y) = \min(x, y)$ (see [G,H3]).)
solve(A,C)$\leftarrow$ rule-cf(A$\leftarrow$B,C1), solve(B,C2), C:=C1$*$C2.
% (This means, that the implication seq = $\rightarrow$ is the adjoint implication to the
% many-valued conjunction $\mathrm{et}^{\bullet}_{P}(x, y) = x \cdot y$ (see [G,P,H3]).)

The heuristic knowledge is expressed in the Prolog rule:

identity(Organism,bacteroides)$\leftarrow$
gram-stain(Organism,gram-negative),
morphology(Organism,rod),
aerobicity(Organism,anaerobic).

So a logic program with uncertainties according to [SS] is a set of pairs $\langle$Clause; Factor$\rangle$ where Clause is a rule and Factor is a certainty factor c, $0 < c \leq 1$ (we will call it confidence factor and write Clause.cf = Factor). So, the above MYCIN-rule is of the form

$$\langle A \leftarrow B_1, B_2, B_3; 0.5\rangle \text{ and/or } A \leftarrow B_1, B_2, B_3.\ \mathrm{cf} = 0.5$$

Now assume that for an Organism = xy we have from some tests following ground facts:

gram-stain(xy,gram-negative). with-cf = 0.9
morphology(xy,rod). with-cf = 0.8
aerobicity(xy,anaerobic). with-cf = 0.7

Then the goal (consultation) ?identity(xy,bacteroides)? is answered YES with a confidence factor $= 0.35 = 0.5 \cdot \min(0.7, 0.8, 0.9)$.

So, the MYCIN assumes that the uncertainty of predicates in the body of the rule (symptoms) is affirmative/overlapping each other and the implication is the product one, describing the conditional probability of independent events. We can imagine that in many cases these assumptions are difficult to fulfill and so we would like to show that our fuzzy Prolog procedure gives correct answers and best possible one's in case arbitrary connectives are considered. So our results will give a theoretical background for any future application based on many-valued logic. In a special case the question is left open which connectives are best fitting with our knowledge base (behind this we can think about a machine learning algorithm which can search for optimal connectives).

Our rules will consist of several bodies coupled by disjunctions and implying the conclusion. From now on a body consists of disjunctions of conjunctions of positive atoms under several, possibly different, t-norms. Note that different t-norms need not be associative and commutative one to another - hence careful bracketing is necessary. Our rule expresses the dependence by an implication seq (the use of which is evaluated by $et^{\bullet}_{seq}$). The usual Prolog backward writing of rules we replace by backward writing qes of our implication seq. A typical rule looks like (we use a prefix notation in the body)

$$A \ qes \ vel([et_1(B_1, et_2(B_2, B_3))], [et_3(C_1, et_4(C_2, C_3))]).cf = 0.8$$

Consider a program P with rules (in propositional logic; the unification and substitution stuff is same as in classical prolog)

$A \ qes \ vel([et_1(B_1, et_2(B_2, B_3))], [et_3(C_1, et_4(C_2, C_3))]).cf = a$
$B_1 \ qes_1 \ et_3(D_1, D_2).cf = b_1$
$C_2 \ qes_2 \ et_2(E_1, E_2)$

and the database

$B_2.cf = b_2$, $B_3.cf = b_3$, $C_1.cf = c_1$, $C_3.cf = c_3$, $D_1.cf = d_1$, $D_2.cf = d_2$, $E_1.cf = e_1$, $E_2.cf = e_2$.

The question (consultation) A? leads to computation which resembles classical prolog, we have just to remember (via a bookkeeping) confidence factors for the final evaluation of the confidence of the answer (see also graded proofs of P. Hájek ([H])):

$$
\begin{aligned}
&et_{seq}(a, vel([et_1(B_1, et_2(B_2, B_3))], [et_3(C_1, et_4(C_2, C_3))]))\\
&et_{seq}(a, vel([et_1(et_{seq_1}(b_1, et_3(D_1, D_2)), et_2(B_2, B_3))], [et_3(C_1, et_4(C_2, C_3))]))\\
&et_{seq}(a, vel([et_1(et_{seq_1}(b_1, et_3(D_1, D_2)), et_2(B_2, B_3))],\\
&\qquad\qquad [et_3(C_1, et_4(et_{seq_2}(c_2, et_2(E_1, E_2)), C_3))]))\\
&\qquad\qquad\qquad\qquad \cdots\cdots\\
&et^{\bullet}_{seq}(a, vel^{\bullet}([et^{\bullet}_1(et^{\bullet}_{seq_1}(b_1, et^{\bullet}_3(d_1, d_2)), et^{\bullet}_2(b_2, b_3))],\\
&\qquad\qquad [et^{\bullet}_3(c_1, et^{\bullet}_4(et^{\bullet}_{seq_2}(c_2, et^{\bullet}_2(e_1, e_2)), c_3))]))
\end{aligned}
$$

what is the confidence factor of the answer YES to the query A?.

Note that instead of the head we have substituted the body of the rule as in usual prolog (unusual in our case are connectives which in classical prolog convention are replaced by commas). The only difference is, we keep the confidence factors of rules and facts in mind until the moment all atoms were replaced by the corresponding confidence factors (the classical resolution would end here successfully). In this moment all connectives are replaced by truth evaluation functions and the term is calculated.

Moreover let us note, that in the case our database of facts would be incomplete and one branch of the search tree below the *vel* would not succeed the value 0 for the confidence factor of that branch will be substituted.

The implementation of this procedure is under the progress by M. Lieskovský (see [Li], this volume).

4 Declarative semantics

We assume that our language L of first order predicate calculus contains several connective seq_i, et_i and vel_i. Herbrand universe U_L consist, as usual, of all ground terms (hence function symbols are interpreted crisp). Herbrand base B_L consists of all ground atoms (see e.g. [Ap], [Ll]). An n-ary predicate symbol should be interpreted as a fuzzy subset of U_L^n. Glueing together all fuzzy predicates we interpret them all at once by a mapping $f : B_L \longrightarrow [0,1]$. The truth value $f(\varphi)$ of a formula φ in the interpretation f is defined along the complexity of φ.

Definition 1. (See [H]) A fuzzy theory is a partial mapping T assigning formulas rational numbers from $(0,1]$.

A interpretation f is a model of T if for all formulas $\varphi \in dom(T)$ we have

$$f(\varphi) \geq T(\varphi).$$

Definition 2. A fuzzy theory P is called a floating fuzzy definite disjunctive program if $dom(P)$ is a finite set (up to variants taking same P-values) of rules (of the form 'head qes body') and facts (atoms).

We do not give here exact definition of computed answer (the exact one is in [Li], this volume). We just refer to our example, recalling that along a standard prolog computation we keep the confidence factors of rules and facts until after successful derivation we evaluate the final confidence of the answer.

Definition 3. A pair (r, Θ) is a computed answer for P and A if Θ is the substitution we fixed along a successful derivation and r is the value calculated from confidence factors, querying the program P by the goal A.

A pair (x, Θ) is correct answer for P and A if for arbitrary interpretation f which is a model of P we have

$$f(\forall(A)) \geq x.$$

Theorem 4. *(Soundness of the floating fuzzy definite disjunctive programs) Every computed answer for a fuzzy definite disjunctive program P and A is a correct answer.*

Proof. Is similar to that of [Ap], [Ll] and [VP2]. Take a goal A with a computation of length $k+1$ starting with a rule $P(A\ qes\ B) = a$. Suppose that the result holds for all computed answers due to computations of length $\leq k$. For each atom D from the body B there is a computation of length $\leq k$, hence computed answer $d \leq f(D)$ in every model of P. But then $f(B) \geq b$ computed answer for whole body, because conjunctions and disjunctions are monotone in both coordinates. Hence as f being a model of P means $f(A\ qes\ B) \geq P(A\ qes\ B) = a$ and by the soundness of modus ponens we get

$$f(A) \geq et^{\bullet}_{seq}(a, b).$$

5 Completeness of fuzzy programs

In [VP2] we have introduced a generalization of the classical fixpoint theory for fuzzy (seq, et_2)-definite programs. We extend this fixpoint theory here to floating fuzzy definite disjunctive programs. We consider the space

$$F_P = \{f : f \textit{ is a mapping from } B_P \textit{ into } [0,1]\}$$

of all fuzzy interpretations equipped with coordinatewise lattice ordering.

Definition 5. We define the operator $T_P : F_P \longrightarrow F_P$ by $T_P(f)(A) = \sup\{r :$ there is a rule $A\ qes_i\ B$ which is a ground instance of some $C \in dom(P)$ and $r = et^{\bullet}_{seq_i}(P(C), f(B))\}$

Theorem 6. *Assume each seq_i fulfils $\Phi 1-3$ with some $et^{\bullet}_{seq_i}$. Moreover assume all $et^{\bullet}_i$, $vel^{\bullet}_i$ are lower semicontinuous. Then*

(1) T_P is continuous (i.e. T_P respects joins of upward directed sets).

(2) f is a model of P iff $T_P(f) \leq f$ (hence the minimal fixpoint of T_P is a model of P).

Proof. (1) The continuity of T_P is straightforward, just recall that $\Phi 1-3$ guarantee the lower semicontinuity of $et^{\bullet}_{seq_i}$ (see S.Gottwald's book [Go]).

(2). Assume f is a model of P. For an $A\ qes_i\ B$ a ground instance of some $C \in dom(P)$ we would like to show that

$$f(A) \geq et^{\bullet}_{seq_i}(P(C), f(B)).$$

We have

$$f(A\ qes_i\ B) \geq f(C) \geq P(C)$$

the first inequality holds because $A\ qes_i\ B$ is a ground instance of C, the second, because f is a model of P. By ($\Phi 1 - 2$) we have

$$et^{\bullet}_{seq_i}(f(B), P(C)) \leq et^{\bullet}_{seq_i}(f(B), f(A\ qes_i\ B)) =$$

$$= et^{\bullet}_{seq_i}(f(B), seq^{\bullet}_i(f(B), f(A))) \leq f(A).$$

Now assume $T_P(f) \leq f$. It suffices for all ground instances $A\ qes_i\ B$ of C to show $f(A\ qes_i\ B) \geq P(C)$. But

$$f(A) \geq T_P(f)(A) \geq et^{\bullet}_{seq_i}(P(C), f(B))$$

gives (by $\Phi 1$ and the above)

$$f(A\ qes_i\ B) = seq^{\bullet}_i(f(B), f(A)) \geq$$

$$seq^{\bullet}_i(f(B), et^{\bullet}_{seq_i}(P(C), f(B))) \geq P(C).$$

The last inequality is $\Phi 3$.

Theorem 7. *(Completeness of floating fuzzy definite disjunctive programs) Assume each seq_i fulfills $\Phi 1 - 3$ with some t-norm $et^{\bullet}_{seq_i}$ and all $et^{\bullet}_i$,$vel^{\bullet}_i$ are lower semicontinuous. Then for every correct answer (x, Θ) for P and A and for every $\epsilon > 0$ there is a computed answer (r, ϑ) for P and A such that $x - \epsilon < r$ and $\Theta = \vartheta\gamma$ (for some γ).*

Proof. The previous theorem practically states the completeness for ground instances of atoms. To extend it for general case does not inflect confidence factors evaluation and using lifting lemma and Mgu lemma of [Ll] we get the full result.

6 Conclusion

We have presented here a mathematical model of disjunctive definite logic programming in many-valued logic with tunable/floating connectives. Although we do not have an axiomatic system we introduced declarative semantics and our operational semantics is based on backward modus ponens (no refutation and resolution) for rules being true implications (and not Horn clauses). The advantage of our approach for deduction engines of expert systems is that we keep functionality in calculating truth values (in opposite to probabilistic/stochastic approach). For systems with small depth of complexity of bodies we can simulate also probabilistic behaviour by choosing appropriate connectives. Our system is also good for large/complex systems with many interrelations when tuning confidence factors can lead to contradictory requirements. Our floating connectives offer a new free axis along which we can better corelate our axioms with the given/learned knowledge base. Moreover this fuzzy/many-valued logic is also good for simulating behaviour of systems where no independent/repeatable experiments can be made (e.g. in economy, banking, stocks market and sometimes also in medicine). Considering disjunctive programming enables us to feature real systems, where several affirmative (though uncertain) answers increase the total confidence. From the theoretical point of view we showed that fixpoint theory works also for many-valued/fuzzy structures with arbitrary lower semicontinuous t-norms/conjunctions for floating fuzzy definite disjunctive logic programming.

References

[Ap] Apt, K. R.: Logic programming. In: van Leeuwen, J.(Ed.): Handbook of Theoretical Computer Science, Vol. B, Formal methods and semantics, 493–574, Elsevier, 1990

[CGT] Ceri S., Gottlob G., Tanca L.:Logic Programming and databases. Springer Verlag, Berlin 1990

[DLP] Dubois D., Lang J., Prade H.: Fuzzy sets in approximate reasoning, Part 2: Logical approaches. Fuzzy Sets and Systems **40** (1991) 203–244

[Gr] Graham I. Fuzzy logic in commercial expert systems - results and prospects. Fuzzy sets and systems 40(1991)451–472

[Go] Gottwald, S.: Mehrwertige Logik. Akademie Verlag, Berlin, 1988

[H] Hájek, P.: Fuzzy logic and arithmetical hierarchy I, II. Fuzzy Sets and Systems (to appear)

[Li] Lieskovský M. Fuzzy WAM for expert systems, this volume

[Ll] Lloyd, J.W.: Foundation of Logic Programming. Springer Verlag, Berlin, 1987

[Me] Mesiar R. Fuzzy sets and probability theory. Tatra Mt. Math. Publ. 1 (1992) 105–123

[Mr] Meritt, D.: Building Expert Systems in Prolog. Springer Verlag, Berlin, 1989

[Ni] Nilsson N.J. principles of artificial intelligence. Springer Verlag, Berlin 1982

[No] Novák, V.: On the syntactico-semantical completeness of first-order fuzzy logic I, II. Kybernetika **26** (1990) 47–26, 134–152

[P] Pavelka, J.: On fuzzy logic I, II, III. Zeitschr. f. Math. Logik und Grundl. der Math. **25** (1979) 45–52, 119–134, 447–464

[SB] Shortliffe, E. H., Buchanan, B. G.: A model of inexact reasoning in medicine. Math. Biosci. **23** (1975) 351–379

[SS] Sterling, L., Shapiro, E.: The Art of Prolog. MIT Press, Cambridge MA, 1986

[VPL] Vojtáš, P., Paulík, L., Lieskovský, M: Expert systems and different logic systems. In: Žižka, J., Brazdil, P. (Eds.): Artificial Intelligence Techniques AIT'95, 233–239, Published by Tech. Univ., Brno, 1995

[VP1] Vojtáš P., Paulík, L.: Logical Programming in RPL and RQL. In: Bartošek, M., Staudek, J., Wiedermann, J. (Eds.): SOFSEM'95: Theory and Practice of Informatics, 487–492, Lecture Notes in Comp. Sci., 1012, Springer, Berlin, 1995

[VP2] Vojtáš P., Paulík, L.: Soundness and completeness of non-classical extended SLD resolution. In Proc.ELP'96 Leipzig, eds. R.Dyckhoff et all., Lecture Notes in Computer Science 1050, Springer Verlag 1996, 289–301

This preprint series was founded in 1996. Its purpose is to present manuscripts of submitted or unpublished papers and reports of fellows of the Mathematical Institute of the Slovak Academy of Sciences in Bratislava and its Košice branch. The authors are fully responsible for the content of the preprints. Postal addresses:

Mathematical Institute
Slovak Academy of Sciences
Štefánikova 49
SK–814 73 Bratislava
SLOVAKIA

Tel./Fax (+421 7) 5249 7316

Mathematical Institute
Slovak Academy of Sciences
Grešákova 6
SK–040 01 Košice
SLOVAKIA

Tel./Fax (+421 95) 6228291

PostScript files of preprints are available at WWW address:
`http://www.mat.savba.sk/preprints`

Latest preprints of the series:

24/1996 *Peter Vojtáš*: Uncertain reasoning with floating connectives

23/1996 *Ivan Korec*: Definability of addition from multiplication and neighbourhood relation and some related results

22/1996 *Ivan Korec*: Open problems more or less related to generalized Pascal triangles

21/1996 *Sylvia Pulmannová*: On connections among some orthomodular structures

20/1996 *Pekka Lahti, Sylvia Pulmannová*: Coexistent observables and effects in quantum mechanics

19/1996 *Ivan Korec*: Definability of arithmetical operations from Pascal's triangle modulo n and arithmetical functions

18/1996 *Ivan Korec*: Undecidability and uniform definability in classes of structures related to Pascal's triangles modulo n

17/1996 *Beloslav Riečan*: Probability theory on some ordered structures

16/1996 *Beloslav Riečan*: Weak observables in MV algebras

15/1996 *Anatolij Dvurečenskij, Hee Sik Kim*: On connections between BCK-algebras and difference posets

14/1996 *Anatolij Dvurečenskij, Karl Svozil*: Product of partition logics, orthoalgebras and automata

13/1996 *Robin L. Hudson, Sylvia Pulmannová*: Chaotic expansion of elements of the universal enveloping algebra of a Lie algebra associated with a quantum stochastic calculus

12/1996 *Sylvia Pulmannová*: Congruences in partial abelian semigroups

11/1996 *Karol Nemoga, Štefan Schwarz*: An explicit description of the set of all normal bases generators of a finite field

10/1996 *Ján Jakubík*: Subdirect product decompositions of MV-algebras

9/1996 *Ivan Korec*: Reversibility in generalized Pascal triangles and binary reversibility in one-dimensional cellular automata

8/1996 *Jozef Bobok, Milan Kuchta*: X-minimal patterns and generalization of Sharkovskiĭ's theorem

7/1996 *Jarmila Hedlíková, Sylvia Pulmannová*: Generalized difference posets and orthoalgebras

6/1996 *Ján Borsík, Roman Frič*: Pointwise convergence fails to be strict

5/1996 *Miroslav Ploščica*: Affine completions of distributive lattices

4/1996 *Ján Haluška*: Equal temperament and Pythagorean tuning: a geometrical interpretation in the plane

3/1996 *Winfried Just, Peter Vojtáš*: On matrix rapid filters

(continued on inside back cover)

2/1996 *Anatolij Dvurerečenskij*: Fuzzy set representations of some quantum structures

1/1996 *Miloslav Duchoň, Beloslav Riečan*: On the product of semigroup valued measures