národní
úložiště
šedé
literatury

Fuzzy Turing Machines Revised

Wiedermann, Jiří
2001

Dostupný z http://www.nusl.cz/ntk/nusl-33990

Dílo je chráněno podle autorského zákona č. 121/2000 Sb.

Tento dokument byl stažen z Národního úložiště šedé literatury (NUŠL).

Datum stažení: 22.07.2024

Další dokumenty můžete najít prostřednictvím vyhledávacího rozhraní nusl.cz .

Institute of Computer Science
Academy of Sciences of the Czech Republic

Fuzzy Turing Machines Revised

Jiří Wiedermann

Technical report No. 838

July 2001

Pod Vodárenskou věží 2, 182 07 Prague 8, phone: +4202 6605 3520, fax: (+4202) 858 57 89,
e-mail:jiri.wiedermann@cs.cas.cz

Institute of Computer Science
Academy of Sciences of the Czech Republic

Fuzzy Turing Machines Revised

Jiří Wiedermann[1]

Technical report No. 838

July 2001

Abstract:

Fuzzy Turing machines and fuzzy languages were introduced by Zadeh, Lee and Santos in nineteen seventies. Unfortunately, it appears that from computability point of view their model is too powerful — its nondeterministic version accepts non–recursively enumerable fuzzy languages. Moreover, from the viewpoint of the modern fuzzy logic theory the model is too restrictive since it is defined only for a specific t-norm (Gödel norm). Therefore we propose a generalization of the original model that is based on rigorous mathematical fundamentals of fuzzy logic. Its acceptance criterion is modified so that the resulting model obeys the Church–Turing Theses.

Keywords:
fuzzy computations, fuzzy Turing machines, fuzzy languages

[1]This research was partially supported by GA ČR grant No. 201/00/1489.

1 Introduction

In this paper we will concentrate to the investigation of computational power of fuzzy computations. This, of course, depends much on the precise definition of the respective computations. Unfortunately, the formal notion of fuzzy computations seems to be quite fuzzy indeed. Attempts to define it have been around since the dawn of the notion of fuzzy algorithms by Zadeh in late nineteen seventies (cf.[6]). In those days fuzzy variants of Turing machines, Markov algorithms, and finite automata (cf. [5],[6]) have been proposed and also fundamentals of fuzzy language theory have been established [3]. Surprisingly, despite of a rich application potential of fuzzy computing (e.g. as embodied in fuzzy control systems) no sufficient attention has been paid to the investigation of recursive–theoretical limits of fuzzy computations.

In order to do so one needs an agreed-upon machine model whose computations will serve as a paradigmatic example of fuzzy computing. No doubts that any attempt to define such a model should start at the same level as in the classical computability (or complexity) theory, i.e. within the classical framework of Turing machine theory. In doing so the above mentioned earlier approaches enhanced by the recent progress in developing the sound (meta)mathematical foundations of fuzzy logic (cf. [1]) should be taken into account.

The benefits from having a formal, generally accepted Turing-machine-like model of fuzzy computations are obvious: such a model would enable a systematic study of the power and efficiency of fuzzy computing, would allow its comparison with standard models of computing and, last but not least, would enable a transfer of known results between the domain of crisp and fuzzy computing.

The aim of this paper is to propose a candidate model for computability and complexity investigations of fuzzy computations and to bring first results along the respective lines. Similarly as earlier approaches (cf. [5] or [6]) this model is based on a generalization of a classic notion of a (nondeterministic) Turing machine. Moreover, it is designed so as to be compatible with recent developments aiming at rigorous mathematical foundations of fuzzy logic. Contrary to the earlier approaches its acceptance criterion is adjusted so as the resulting model obeys the Church-Turing thesis. Thanks to this fact the model is more general and more 'practical' than its forerunners. It covers a full spectrum of specific t–norms some of which were treated separately in early days of fuzzy computing. Thanks to its modified acceptance criterion the computational power of this model does not exceed that of the classical models what opens the way of its realization by the latter models (e.g. by neural nets).

In Section 2 we will define the basic variant of a fuzzy Turing machine — viz. the one that makes use of a fuzzy instruction set. It accepts its inputs with various truth degrees which also depend on the choice of the underlying t–norm. This norm presents the link to the respective fuzzy predicate calculus. The classical notion of acceptance by a fuzzy Turing machine is considered first. Due to our more general definition of a fuzzy Turing machine a proof that the respective definition of acceptance is sound is needed. In Section 4 a result showing that the languages accepted by nondeterministic variants of fuzzy Turing machines with the classical acceptance criterion are not recursively enumerable ones. This is in contrast to deterministic variants which accept languages which are recursively enumerable. In Section 5 we propose a modified acceptance criterion that leads to fuzzy Turing machines that obey the Church-Turing thesis while retaining their generality and relation to crisp machines. In Section 6 the merit of the presented results is discussed.

For easier understanding of fuzzy logics background the acquaintance with e.g. the second chapter of the monograph [1] is to be recommended. For fundamentals of computability and formal language theory cf. [2].

2 Fuzzy Turing machines

Inspired by the earlier proposals of fuzzy Turing machines (especially those from [5] and [6]) we define our basic variant of a fuzzy Turing machine. The general design idea is that the machine accepts words of a (fuzzy) language and at the same time it determines the membership degree of each accepted word in the respective language. The machine will be obtained from the standard nondeterministic Turing machine by fuzzification of its 'instruction set'. The membership degree of each accepted word is derived from so-called truth degrees associated with each computational path. These truth degrees

are computed by a special truth function called t–norm. This t–norm determines the kind of fuzzy propositional calculus in which the respective computations can also be described (cf. [1]). Specific variants of fuzzy Turing machines, among them also the one defined originally by Santos [5], can be obtained by choosing a particular t–norm.

Next we proceed to the definition of a fuzzy Turing machine. Prior to giving the respective definition we introduce the definition of a *t–norm* (cf. [1]).

Definition 1 *A* t–norm *is a binary operation* $*$ *on [0,1] (i.e. t: [0,1]*$^2 \to$ *[0,1]) satisfying the following conditions:*

1. $*$ *is commutative and associative, i.e for all* $x, y, z \in [0,1]$ *we have* $x * y = y * x$ *and* $(x * y) * z = x * (y * z)$;
2. $*$ *is non–decreasing in both arguments, i.e.* $x_1 \leq x_2$ *implies* $x_1 * y \leq x_2 * y$ *and* $y_1 \leq y_2$ *implies* $x * y_1 \leq x * y_2$;
3. *for all* $x \in [0,1]$ *we have* $1 * x = x$ *and* $0 * x = 0$.

(Note that we do not require a continuity of the *t*–norm.) For the sake of simplicity, we will introduce only the definition of a single–tape fuzzy Turing machine since we will be mainly interested in its principal computational abilities and less in its effectiveness.

Definition 2 *A* nondeterministic single–tape fuzzy Turing machine *(Fuzzy-NTM) is a ten–tuple* $\mathcal{F} = (S, T, I, \Delta, b, q_0, q_f, M, \mu, *)$ *where:*

1. S *is the finite set of* states*;*
2. T *is the finite set of* tape symbols*, to be printed on a tape that has a left-most cell but is unbounded to the right;*
3. I *is the set of* input symbols*;* $I \subseteq T$;
4. Δ *is the next-move relation which is a subset of* $S \times T \times S \times T \times \{-1, 0, 1\}$*. For each possible move of* $\mathcal{F}$ *there is an element* $\delta \in \Delta$ *with* $\delta = (s_1, t_1, s_2, t_2, d)$*. That is, if the current state is* s_1 *and the tape symbol scanned by the machine's head is* t_1*,* $\mathcal{F}$ *will enter the new state* s_2*, the new tape symbol* t_2 *will rewrite the previous symbol* t_1*, and the tape head will move in direction* d*. (In the previous relation symbol* -1 (1) *denotes a move by one cell to the left (right) and* 0 *denotes no move.)*
5. b*, in* $T - I$*, is the* blank*;*
6. q_0 *is the* initial state*;*
7. q_f *is the* final*, or* accepting *state;*
8. M *is a finite subset of the real interval [0,1], of cardinality* $|M| > 0$;
9. $\mu : \Delta \to M$ *is a function that to each move* δ *assigns the truth degree* $\mu(\delta)$ *of its membership in* Δ
10. $*$ *is a t–norm.*

Note that the membership degree of $\delta \in \Delta$ equals the truth degree of the proposition "δ *is an element of* Δ". For $\delta = (s_1, t_1, s_2, t_2, d) \in \Delta$ we will define a predicate $\Delta(s_1, t_1, s_2, t_2, d)$ and we will say that the truth degree of $\Delta(s_1, t_1, s_2, t_2, h)$ equals α if and only if $\mu(\delta) = \alpha$.

The notion of computation is defined as usual with the help of instantaneous descriptions (IDs). An *instantaneous description* Q_t of $\mathcal{F}$ working on input w at time $t > 0$ is a unique description of machine's tape, of its state and of the position of the machine's head after performing its t-th move on input w. If Q_t and Q_{t+1} are two IDs we will write $Q_t \vdash^{\alpha} Q_{t+1}$ and say that Q_{t+1} is *reachable in one step* from Q_t with truth degree α if and only if there is a possible move in Δ, with truth

degree α, leading from Q_t to Q_{t+1}. On input w the machine starts its computation in the respective *initial* ID Q_0. This is an ID describing the tape holding a string of n input symbols (the so–called *input string*, or *input word*), one symbol per cell starting with the leftmost cell. All cells to the right of the input string are blank. The head is scanning the leftmost cell and the current state is q_0. From the initial ID the computation proceeds to IDs that are reachable in one step from Q_0, etc. If $Q_0 \vdash^{\alpha_0} Q_1 \vdash^{\alpha_1} Q_2 \ldots \vdash^{\alpha_{k-1}} Q_k$, with $\alpha_0, \ldots, \alpha_{k-1} \in M$ we say that Q_k is *reachable* from Q_0.

Now we establish a relation between the truth degrees of individual moves and those of achieving individual IDs. In order to do so note that, intuitively, within the propositional calculus belonging to the respective t–norm, for a particular move $\delta = (s_1, t_1, s_2, t_2, h) \in \Delta$ its membership degree $\alpha = \mu(\delta)$ can be interpreted as the truth degree of the proposition "*at each time t, when the machine is in state s_1, its head is scanning the i-th cell with symbol t_1, the machine will subsequently enter state s_2, rewrite the scanned symbol by t_2 and move its head in direction h*". Then, this truth degree can be extended to any ID of $\mathcal{F}$ reachable from its initial ID as follows.

Let Q_t be reachable from Q_0 via a computational path $Q_0 \vdash^{\alpha_0} Q_1 \vdash^{\alpha_1} Q_2 \ldots \vdash^{\alpha_{t-1}} Q_t$, with $\alpha_0, \ldots, \alpha_{t-1} \in M$, let $D((Q_0, Q_1, \ldots, Q_t))$ denote the truth degree of the proposition *"after t steps, starting from Q_0 and proceeding along the computational path $Q_0 \vdash^{\alpha_0} Q_1 \vdash^{\alpha_1} Q_2 \ldots \vdash^{\alpha_{t-1}} Q_t$, the instantaneous description of $\mathcal{F}$ is Q_t"*. The corresponding *evaluation function* D is defined as

$$D((Q_0, Q_1, \ldots, Q_t)) = \begin{cases} 1, & t = 0 \\ D((Q_0, Q_1, \ldots, Q_{t-1})) * \alpha_{t-1}, & t > 0 \end{cases} \qquad (*)$$

Thus, the above mentioned truth degree is being 'adjusted' along any computational path with the help of the respective t–norm which acts as the truth function of the (strong) conjunction & (cf. [1]). Due to nondeterminism it may happen that Q_t is reachable from Q_0 via different computational paths. Therefore, the 'path independent' truth degree $d(Q_t)$ of the proposition "*after t steps, starting from Q_0 the ID of $\mathcal{F}$ is Q_t*" is defined as

$$d(Q_t) = \max\{D((Q_0, Q_1, \ldots, Q_t))\}$$

where the maximum is taken over all computational paths leading from Q_0 to Q_t.

A sequence $Q_0, Q_1, \ldots, Q_q$ of IDs is called an *accepting sequence of IDs* of $\mathcal{F}$ on input w, if and only if Q_0 is an initial ID, $Q_{i-1} \vdash^{\alpha_i} Q_i$ for $1 \leq i \leq q$, and Q_q is an accepting ID (i.e. such ID that contains the final state q_f). If an accepting ID Q_q is reachable from q_0 on input w we say that w is accepted with truth degree $d(Q_q)$.

3 The classical acceptance criterion

Now we are in a position to define the acceptance criterion for a fuzzy Turing machine. We will first consider the original criterion from the earliest papers by Lee, Santos and Zadeh from nineteen seventies (cf. [3], [5], or [6]). For the further purposes we will refer to this criterion as to the classical acceptance criterion.

A Fuzzy-NTM works as a (fuzzy) language acceptor as follows. The tape symbols of the machine include the alphabet of the language, called the input symbols, a special symbol *blank*, denoted b, and perhaps other symbols.

Definition 3 (The clasical acceptance criterion) *Let $\mathcal{F} = (S, T, I, \Delta, b, q_0, q_f, M, \mu, *)$ be a Fuzzy-NTM. The input string w is* accepted with truth degree $e(w)$ *by $\mathcal{F}$ if and only if*

- *there exists an accepting ID reachable from the initial ID Q_0 on input w;*
- *$e(w) = \max_Q\{d(Q) | Q$ is an accepting ID reachable from $Q_0\}$*

Next we show that the previous definition is sound — i.e. that the maximum $e(w)$ of truth degrees of accepting IDs over all accepting paths (if there is at least one) always exists.[2] Therefore consider

[2]Note that for the case of Gödel norm (as studied in [5] and [6]) such a proof is not needed since in this case $e(w) \in M$, i.e. the maximum is taken over a finite set.

the commutative ordered semigroup $G = \langle [0,1], *, \leq \rangle$, where $*$ is a t-norm. Let $M = \{\alpha_1 < \alpha_2 < \ldots < \alpha_k\} \subset [0,1]$ be the set of instruction membership degrees from Definition 2, let $G(M)$ be a subsemigroup of G generated by M. Syntactically, the elements of $G(M)$ are formed by 'products' of elements of M. Referring to the commutative and associative properties of the respective t-norm the elements of $G(M)$ take the form $\alpha_k^{s_k} * \alpha_{k-1}^{s_{k-1}} * \ldots * \alpha_1^{s_1}$, with $1 \geq \alpha_k > \alpha_{k-1} > \ldots > \alpha_1 > 0$ and $s_i \geq 0$ for $1 \leq i \leq k$. In the previous expression we wrote α^k to denote the product $\alpha * \alpha * \ldots * \alpha$ having k factors. Each element $\alpha \in G(M)$ of the previous form is uniquely determined by an integer k-tuple $\mathbf{s} = (s_k, s_{k-1}, \ldots, s_1)$. Such a k-tuple will be called a *tuple representation* (w.r.t. the subalgebra $G(M)$) of the respective element. The *length* of a tuple $\mathbf{s}$ is $len(\mathbf{s}) = s_k + s_{k-1} + \ldots s_1$. Let $\tau_k[\alpha_k, \alpha_{k-1}, \ldots, \alpha_1] : \mathbf{Z}^k \rightarrow [0,1]$ be the function that to each k-tuple representation of an element of $G(M)$ assigns the respective real number in [0,1]. Instead of $\tau_k[\alpha_k, \alpha_{k-1}, \ldots, \alpha_1]$ we will also write $\tau_k[M]$ or simply τ_k. Thus, under the previous notation, for element $\alpha \in G(M)$ represented by k-tuple $\mathbf{s}$ we have $\tau_k(s) = \alpha$.

Over the tuple representation of elements of $G(M)$ we will consider the partial order '$\sqsubseteq$' defined as follows: we will say that tuple $\mathbf{s} = (s_1, s_2, \ldots, s_k)$ *dominates* tuple $\mathbf{r} = (r_1, r_2, \ldots, r_k)$ (written as $\mathbf{r} \sqsubseteq \mathbf{s}$), or that $\mathbf{r}$ is *dominated by* $\mathbf{s}$ if and only if $r_i \leq s_i$ for $i = 1, 2, \ldots, k$. If an element dominates another one we say that the two elements are comparable. Considering the properties of the respective t-norm we can prove the following proposition.

Proposition 1 *If* $\mathbf{r} \sqsubseteq \mathbf{s}$ *then* $\tau_k(\mathbf{r}) \geq \tau_k(\mathbf{s})$.

A set D of k-tuples from $G(M)$ is called *independent* if and only if no two its elements are comparable.

Proposition 2 *For any* $k \geq 1$, *with* $|M| = k$, *any independent subset of* k*-tuples from* $G(M)$ *is finite.*

Proof: Assume that there would be an infinite independent subset D_k of k-tuples of $G(M)$. We will show that then there must exist infinite independent sets of i-tuples for any $1 \leq i < k$.

Choose any $\mathbf{s} = (s_1, s_2, \ldots, s_k)$ from D_k and consider a decomposition of D_k into all subsets consisting of elements whose i-th component is fixed to some value between 0 and s_i, for $i = 1, 2, \ldots, k$. Each k-tuple $\mathbf{r} \in D_k$ must fall into at least one of such subsets since, thanks to independence of D_k, at least one of the components of $\mathbf{r}$ must be less than the corresponding component of $\mathbf{s}$. There is a finite number of such subsets (which need not be mutually disjoint) and their union equals D_k. Because there is at most a finite number of such subsets, some of them must be infinite, since otherwise D_k would be finite. Choose any such infinite subset corresponding to some fixed component value of its i-th component and remove this component from all the corresponding k-tuples. As a result we get an infinite independent set D_{k-1} of $(k-1)$-tuples.

Now repeat the previous construction with set D_{k-1} until we get the decomposition of the given infinite independent set into a finite number of independent subsets of a finite size. But this would contradict the assumption on the infiniteness of the original set. Note that at the latest such a situation will occur when we reach $k = 1$ since there is no infinite independent set of elements which consist of a single component.

Thus, the assumption that D_k was an infinite independent set was wrong.

□

Next we show that that $G(M)$ is well-ordered w.r.t. the standard ordering '$\leq$', i.e. each subset of $G(M)$ has a maximal element.

Lemma 1 *Let* F *be a subset of* $G(M)$. *Then* F *contains a maximal element, i.e. there is an element* $\mathbf{a} \in F$ *such that* $\mathbf{x} \leq \mathbf{a}$ *for all* $\mathbf{x} \in F$.

Proof: Assume that there is no maximal element in F. In this case to each element $\mathbf{x_1} \in F$ there would be element $\mathbf{x_2} \in F$, with $\mathbf{x_1} < \mathbf{x_2}$, etc. Thus, there would be an infinite increasing chain $\mathbf{x_1} < \mathbf{x_2} < \ldots$ of elements of F. We show that in this chain there must exist i and j such that $i < j$ and $\mathbf{x_i} \geq \mathbf{x_j}$.

To see this consider the infinite sequence of k-tuples corresponding to the elements of the previous chain. Since there is but a finite number of tuples of each length, in this sequence there must be an infinite sub-sequence consisting of tuples of non-decreasing length. In this sub-sequence tuples are either incomparable or there exist pairs of comparable tuples. In the former case the respective tuples

will form an infinite independent set which, thanks to Proposition 2 cannot exist. In the latter case let there be two indices i, j with $i < j$ and two comparable tuples $\mathbf{r}$ and $\mathbf{s}$ in the subsequence such that $\tau_k(\mathbf{r}) = \mathbf{x_i}$ and $\tau_k(\mathbf{s}) = \mathbf{x_j}$. In this case since $len(\mathbf{r}) \leq len(\mathbf{s})$ the only possibility concerning the relation between $\mathbf{r}$ and $\mathbf{s}$ is $\mathbf{r} \sqsubseteq \mathbf{s}$ and consequently $\tau_k(\mathbf{r}) \geq \tau_k(\mathbf{s})$ (by Proposition 1). But this means that $\mathbf{x_i} \geq \mathbf{x_j}$ what is a contradiction with the assumption of the respective sub-chain ordering.

It follows that there are no infinite increasing chains in F and therefore it must contain the maximal element.

□

Corollary 1 *Let $\mathcal{F}$ be a Fuzzy-NTM, let w be any input, let $\mathcal{A} \neq \emptyset$ be the set of all accepting IDs of $\mathcal{F}$ on input w, let d be the evaluation function. Then the set $\{d(A)|A \in \mathcal{A}\}$ contains an element with a maximal truth degree.*

Proof: Consider the computational tree T of $\mathcal{F}$ on input w. To each path in T of form $Q_0 \vdash^{\alpha_{i_0}} Q_1 \vdash^{\alpha_{i_1}} Q_2 \ldots \vdash^{\alpha_{i_r}} Q_{r+1}$, with $r \geq 0$ and $\alpha_{i_j} \in M$, an element $\alpha_{i_0} * \alpha_{i_1} * \ldots \alpha_{i_r} \in G(M)$ is assigned. Hence, to each accepting ID $A \in \mathcal{A}$ there is a corresponding element $\alpha_A \in G(M)$ whose value equals to the truth degree of A. Let $F = \{\alpha_A|A \in \mathcal{A}\}$ be the set of elements corresponding to all accepting IDs in T. Clearly, $F \subseteq G(M)$ and hence, according to Lemma 1 there exists its maximal element α. Obviously, the corresponding ID A from $\mathcal{A}$ will get the maximal truth degree $d(A) \in [0, 1]$.

□

Now we proceed to the definition of fuzzy languages. The idea is to define a fuzzy language as a fuzzy set of words. That is, each word from the respective language belongs to it with a certain *membership grade* which is a real number between 0 and 1. This definition can be found already in [3] (but also in recent papers, such as [4]) which seems to be the first paper dealing with formal fuzzy languages. If such a language is recognized by a fuzzy Turing machines then the membership grade of each accepted word is equal to its acceptance truth degree.

Definition 4 *The* fuzzy language accepted *by $\mathcal{F}$ is the fuzzy set of ordered pairs*

$$L_{\mathcal{F}} = \{(w, e(w))|\ w \text{ is accepted by } \mathcal{F} \text{ with truth degree } e(w)\}$$

Function e in the context mentioned above is also called *membership function* of $L_{\mathcal{F}}$. Observe that in Definition 4 only words with the maximal acceptance degree are considered. It is important to realize that $\mathcal{F}$ does not 'print' the truth degree $e(w)$ corresponding to the accepted word w. In fact, this would also be principally impossible due to the fact that truth degrees are real numbers. Rather, this truth degree is *only defined* by Definition 3, but *not computed* by $\mathcal{F}$. In the next section we prove that in general it must be so since fuzzy Turing machines with the classical acceptance criterion can also recognize some non-recursively enumerable languages, i.e languages with a non-computable characteristic function.

Note that in case when $\mu(\delta) = 1$ for all $\delta \in \Delta$ the Fuzzy-NTM equals the classical acceptance criterion the standard NTM as defined e.g. in [2]. Such a machine is also called a *crisp Turing machine.*

4 The power of classical fuzzy Turing machines

We prove a result concerning the super-Turing computing power of Fuzzy-NTMs with the classical acceptance criterion.

Theorem 1 *There exist non r.e. languages accepted by Fuzzy-NTMs with the classical acceptance criterion.*

Proof: Let K be the 'standard' undecidable language that corresponds to the HALTING PROBLEM (cf. [2]). Consider a Fuzzy-NTM $\mathcal{F}$ that has a single nondeterministic branch which is the only fuzzy instruction of $\mathcal{F}$. This branch leads from the initial ID either

- via an instruction with truth degree 0 to an accepting state or
- via an instruction with truth degree 1 to an ID that is the starting point of a deterministic computation that enumerates K and eventually accepts w if and only if $w \in K$.

The language L accepted by $\mathcal{F}$ is the union of two sets $\{(w,0)|\ w \notin K\}$ and $\{(w,1)|\ w \in K\}$. If L were computationally enumerable (i.e. if there were a crisp machine simulating $\mathcal{F}$ in the sense as mentioned above) then we obtained as a contradiction that the complement of K, which is equal to the set of all w such that $(w,0) \in L$, would be computationally enumerable, too.

□

Corollary 2 *There is no crisp Turing machine that could simulate any given Fuzzy-NTM with the classical acceptance criterion.*

Note that the assumption that machine $\mathcal{F}$ was a nondeterministic Turing machine was a crucial one in the previous theorem. Namely, the deterministic Turing machine with fuzzy instruction set can be simulated by crisp machines, in the following sense:

Theorem 2 *If $\mathcal{D}$ is a Fuzzy-DTM with the classical acceptance criterion then there exists a (crisp) deterministic machine C that simulates $\mathcal{D}$ and outputs the acceptance truth degrees in the tuple representation (cf. Section 2 for the notion of tuple representation).*

Proof: Machine C has no problems in simulating $\mathcal{D}$ since if a word is accepted by $\mathcal{D}$ then there is exactly one computational path leading to acceptance that has to be followed by C. Following this path C also computes the tuple representation of $e(w)$. If on an input $\mathcal{D}$ runs forever then C will do the same.

□

Corollary 3 *Nondeterministic fuzzy Turing machines with the classical acceptance criterion are more powerful than the crisp ones.*

Thus, for machines with Zadeh et al.'s acceptance criterion fuzzy nondeterminism is more powerful than fuzzy determinism, and also fuzzy nondeterminism is more powerful than the 'pure', crisp nondeterminism.

Note that time–bounded fuzzy nondeterministic computations with the classical acceptance criterion can be simulated by deterministic machines, thanks to the fact that there is a known upper bound (which is equal to the time bound) limiting the depth of the underlying nondeterministic computational tree that is to be traversed when looking for the accepting state.

5 Partially computable acceptance criterion

In the context of computability theory it would be desirable that also fuzzy Turing machines obey the Church Turing Thesis. The reason why fuzzy Turing machine with the classical acceptance criterion accept also non r.e. languages is nicely seen on the example of language L from the proof of Theorem 1. Here, the 'non-recursive' part of L, i.e language $L_1 = \{(w,0)|\ w \notin K\} \subseteq L$ is accepted thanks to the fact that on input $w \in L_1$ the computations following the branch with truth degree 1 never reach the accepting state. Therefore they are 'not considered' in definition 4 and the accepting ID with truth degree 0 is chosen is the one satisfying the classical acceptance criterion. Yet, from purely computational point of view there is no computable evidence that this choice was correct. This motivates the following revision of the original acceptance definition. First, modify the fuzzy machine so that it will cycle in all halting states (i.e. in all accepting states and in all states from which there is no continuation of the computation on a given input). The membership degree of the respective 'cycling' instruction from Δ will be set to 1. This will cause that all computational paths will be infinite and the truth degree of the halting ID will be propagated downwards the tree infinitely. We say that in the computational tree an ID Q_k finds itself at depth $k \geq 0$ if there is a computational path from the initial ID Q_0 to Q_k of length k.

Definition 5 (Partially computable acceptance criterion) *Let $\mathcal{F} = (S, T, I, \Delta, b, q_0, q_f, M, \mu, *)$ be a Fuzzy-NTM. The input string w is* accepted with truth degree $e(w)$ *by $\mathcal{F}$ if and only if*

- *on input w, the computation of $\mathcal{F}$ reaches an accepting ID Q_k at depth k and $e(w) = d(Q_k)$;*
- *evaluation $d(Q_k)$ is not less than the evaluations of all IDs at the same depth*
- *k is the minimal depth at which such accepting ID exists.*

Now the proof that the above definition is sound is almost obvious since thanks to the properties (cf. Proposition 1) of the t–norm along any computational path the truth degree of IDs do not increase. Therefore the maximum of truth degrees of ID on any level is no less that the truth degrees of IDs at lover levels.

Proposition 3 *Deterministic fuzzy Turing machines and crisp machines with either classical or partially computable acceptance criteria are equivalent.*

Proof: In the case of deterministic fuzzy Turing machines to each input there is at most one accepting path and therefore both acceptance definitions coincide. In the case of crisp nondeterministic machines, it does not matter which accepting path is selected since all get the truth degree 1.

□

Note that unlike the classical criterion, the new criterion leads to a reasonable definition of time complexity of fuzzy computations based on the acceptance depth. Once a word is accepted at some depth, then a proof can be delivered that the respective truth degree of acceptance is indeed the maximal one as required by the Definition 5. The length of this proof depends on the acceptance depth. Nothing like that is possible for languages recognized by a fuzzy Turing machine equipped with the classical criterion (consider e.g. the case of language L_1 from the beginning of this section).

For partially computable acceptance criterion we can define the notion of fuzzy languages much in the same way as in Definition 4. However, in a contrast to Theorem 1, now the class of fuzzy languages recognized by fuzzy Turing machines with partially computable acceptance criterion will be 'computable', in the following sense.

Theorem 3 *Let $M \subseteq [0, 1] \subset \mathbb{Q}$ be a finite set of rational numbers, let $*_c$ be a computable t-norm. Let $G = \langle [0, 1], *_c, \leq \rangle$ be a commutative ordered semigroup and $G(M)$ its sub-semigroup generated by M. Let $\mathcal{F} = (S, T, I, \Delta, b, q_0, q_f, M, \mu, *_c)$ be a Fuzzy-NTM with a partially computable acceptance criterion.*

Then a fuzzy language L is recognized by $\mathcal{F}$ if and only if the membership function of L is a partially computable function over $G(M)$.

Proof: Consider an input w to $\mathcal{F}$ and simulate the actions of $\mathcal{F}$ by a deterministic Turing machine D in the following way. Machine D traverses the computational tree T of $\mathcal{F}$ in a breadth first manner. For each ID reached D computes the respective truth degree from $G(M)$. This is possible thanks to the fact that $*_c$ is a computable function and M is a set of rational numbers. Once completing the simulation of one level in T machine D checks whether there was an accepting ID at this level with a maximal evaluation $e(w)$ from among evaluations of all IDs at this level. If so then D accepts w and outputs $e(w)$. Otherwise D resumes the simulation. Therefore the membership function of L is partially computable.

To prove the reverse statement assume that the membership function of L is partially computable over $G(M)$. That is, there is deterministic Turing machine D that, given input w and set M and t-norm $*_c$ as above computes $e(w)$ by applying operator $*_c$ to elements of $G(M)$. We assume that there is a deterministic Turing machine N that, given $x, y \in G(M)$, with $x, y, \in \mathbb{Q}$ on its input tape, computes and prints $x *_c y$ on its output tape. Thus, in order to compute $x *_c y$ machine D calls N as a subroutine. Modify D into a machine D' that along with $e(w)$ also computes its tuple representation $\mathbf{s}$, with $\tau_k(\mathbf{s}) = e(w)$.

Let $\mathcal{F} = (S, T, I, \Delta, b, q_0, q_f, M, \mu, *_c)$ be a deterministic fuzzy Turing machine with a partially computable acceptance criterion. On input w machine $\mathcal{F}$ faithfully simulates D' by crisp instructions until tuple $\mathbf{s} = (s_1, s_2, \ldots, s_k)$ is computed. This means that $e(w) = \alpha_k^{s_k} * \alpha_{k-1}^{s_{k-1}} * \ldots * \alpha_1^{s_1}$, with

$1 \geq \alpha_k > \alpha_{k-1} > \ldots > \alpha_1 > 0$, $s_i \geq 0$ and $\alpha_i \in M$ for $1 \leq i \leq k$. Therefore, what remains to do for $\mathcal{F}$ is to perform a sequence of moves that transform the current ID (with truth degree 1) to an ID with truth degree $e(w)$. This is simply achieved by following the structure of $\mathbf{s}$ and subsequently performing s_i moves with truth degree α_i, for $i = 1, 2, \ldots, k$. Then $\mathcal{F}$ enters the accepting state whose truth degree clearly is $e(w)$ and this is the acceptance degree of w since $\mathcal{F}$ was a deterministic fuzzy machine.

□

This result shows that fuzzy Turing machines with a partially computable acceptance criterion satisfy the Church-Turing Thesis. It thus opens the way for simulating fuzzy computations by any other device obeying this thesis. In particular, fuzzy computations can be simulated by RAMs, (families of) neural nets, circuits, cellular automata, etc.

6 Conclusion

We have proposed a fuzzy variant of a nondeterministic Turing machine to serve as a formal model of fuzzy computations. To do so we have generalized the earlier approaches and also adjusted the originally considered acceptance criterion. The resulting model is fully compatible with the current state of fuzzy logic theory and fits well into the family of devices obeying the Church–Turing thesis. This model is intended to be used in computability and complexity-theoretic investigations aiming at the power, limits and efficiency of fuzzy computations, rather than in design of efficient fuzzy algorithms.

Acknowledgement. The author is grateful to P. Hájek for several useful comments and suggestions.

Bibliography

[1] Hájek, P.: Metamathematics of Fuzzy Logic. Kluwer, 1998

[2] Hopcroft, J. E. — Ullman, J. D.: Introduction to Automata Theory, Languages, and Computation. Addison–Wesley Publishing Company, Reading, Mass., 1979,417 p.

[3] Lee, E.T. — Zadeh, L.A.: Note on Fuzzy Languages. Information Science, Vol. 1, No. 4, pp. 421–434, 1969

[4] Matescu, A. — Salomaa, A. — Salomaa, K. — Yu, S.: Lexical Analysis with a Simple Finite–Fuzzy–Automaton Model. J. Universal Comp. Sci., Vol 1, No. 5, pp. 292–311, 1995

[5] Santos, E.: Fuzzy Algorithms. Information and Control, Vol. 17, pp. 326–339, 1970

[6] Zadeh, L.A.: Fuzzy Algorithms. Information and Control, Vol. 12, No. 2, pp. 94–102,1968