Fuzzy time and possible impacts of it on science [10]

TC^*

Farzad Didehvar

didehvar@aut.ac.ir

Amir Kabir University of Technology (Tehran Polytechnic)

Abstract. One of the possible hypotheses about time is to consider any instant of time as fuzzy number, so that two instants of time could be overlapped. Historically, some Mathematicians and Philosophers have had similar ideas like Brouwer and Husserl [5].

Throughout this article, the impact of this change on Theory of Computation and Complexity Theory are studied. In order to rebuild Theory of Computation in a more successful and productive approach to solve some major problems in Complexity Theory, the present research is done. This novel theory is called here, the fuzzy time theory of computation, TC^*.

Keywords. $P \neq NP$, $P = PBB$, $MA = AM$, Fuzzy Time, TC^*, Reducibility, Complexity Theory Problems

1. Introduction

Throughout this article, the author presents the Theory of Computation by appllying Fuzzy Time. More specifically, the author tries to rebuild the structure of the Theory of computation based on considering time as a fuzzy concept.

In fact, there are reasons to belive time as a fuzzy concept. In this article, the author does not try to assume these reasons and argue about, but just to note that Brouwer and Husserl views on the concept of time were similar [5].

More precisely, here, given the classical definition of Turing Machine, the concept of Time is changed to be Fuzzy. This new theory is called Theory TC^* and this type of computation "Fuzzy time Computation". We have relatively large number of fundamental unsolved problems in Complexity Theory. In the new theory, some of the major obstacles and unsolved problems have been solved. It should be noted that in this article, the author considers fuzzy number associated to instants of time as a symmetric one. The point is about applying the symmetricity of fuzzy time function in the proof of Lemma 3.

In particular, the new classes of complexity Theory, P^*, NP^*, BPP^* in the TC^* are defined similar to the definitions of P, NP and BPP as their natural alternative definition. Here, we will see, $P^* = BPP^*$, $MA^* = AM^*$.

2. Reducibility

In this section, firstly, we define a quasi-order relation in TC^* analogues with the m-reducibility in TC .

It should be reminded that a fuzzy time Turing Machine is a Turing Machine which works with fuzzy time.

In addition, here, the Turing Machine is considered as a two tuple (M, S). Whereas, M is a Turing machine in the usual sense and S is a polynomial function. Meanwhile, M runs in bounded time by S, equivalently, $M(x)$ in less than $S([x])$ steps is computed.

First, we remind the Classical definition of m-reducibility:

$Y >_m X$, if there is a polynomial time computable function f such that:

$$x \in X \leftrightarrow f(x) \in Y$$

The parallel definition in TC^* is introduced as following

Definition 1: For $\alpha > \frac{1}{2}$, $Y >_{\mathbf{m}}^{\boldsymbol{\alpha}} X$ if there is a polynomial time computable* function f such that:

1. $x \in X \ \& \ f(x) \downarrow \text{ in bounded time} \leftrightarrow (f(x) \in Y)$

2. Pr $(f(x) \downarrow \text{ in bounded time}) > \alpha$

A Computable* function f is a function that is computable by a fuzzy time Turing machine.

Here, by bounded time, we mean that for the function f there exists a Polynomial function h such that $f(x) \downarrow$ in less than $h\,(\text{length}(x))$ steps.

$Y >_{\mathbf{m}}^{\boldsymbol{\alpha}} X$ can be represented by a 5-tuple, (Y, X, f, S_f, α), $S_f(x)$ is the number of steps that $f(x)$ is computed. The definition is as follows

$$Y >_{\mathbf{m}}^{\boldsymbol{\alpha}} X \leftrightarrow (Y, X, f, S_f, \alpha) \text{ is an acceptable 5-tuple}$$

One of the major question here is about the independence of the definition from the value of α? $(\alpha > \frac{1}{2})$

In the first step, to answer the above question, we need the following simple lemma.

Lemma 1 Let for $1 > \alpha > \frac{1}{2}$, (Y, X, f, S_f, α) is an acceptable 5-tuple then for any $1 > \beta > \frac{1}{2}$ there is a computable function g in which (Y, X, g, S_g, β) is an acceptable 5-tuple.

Proof. Actually, there is a natural number k, so that the function g is equivalent to, k times repeating f, till we reach a solution with probability less than β. It is easy to understand that such k exists. □

Lemma 1 indicates for $1 > \alpha > \frac{1}{2}$, the relation $Y >_{\mathbf{m}}^{\boldsymbol{\alpha}} X$ would be independent of α. So, we define $Y >_{\mathbf{m}}^{*} X$ as follows

Definition 2. $Y >_{\mathbf{m}}^{*} X$ if for some α $(1 > \alpha > \frac{1}{2})$, $Y >_{\mathbf{m}}^{\boldsymbol{\alpha}} X$.

Lemma 2. $>_{\mathrm{m}}^{*}$ is a quasi-order relation.

Proof. $X >_{\mathbf{m}}^{\boldsymbol{\alpha}} Y$ implies $\forall \frac{1}{2} > \varepsilon > 0 \quad X >_{\mathbf{m}}^{\mathbf{1}-\varepsilon} Y$ (*)

$Y >_{\mathbf{m}}^{\boldsymbol{\alpha}} Z$ implies $\forall \frac{1}{2} > \varepsilon > 0 \quad Y >_{\mathbf{m}}^{\mathbf{1}-\varepsilon} Z$ (**)

From (*), (**), we have $\forall \frac{1}{2} > \varepsilon > 0 \quad X >_{\mathbf{m}}^{(\mathbf{1}-\varepsilon)^2} Y$ (***). □

Lemma 3. $Y >_{\mathrm{m}} X$ implies $Y >_{\mathbf{m}}^{*} X$.

Proof. …..

Remark 1. Using lemma 3, suppose we have a computation by Turing Machine $(\mathrm{M}, \mathrm{S_f})$ and the input x in classical time and $(\mathrm{M}, \mathrm{S_f})(\mathrm{x}) \downarrow$. If we change the classical time to the symmetric fuzzy time, the probability of reaching to the final state is more than $\frac{1}{2}$. As a conclusion, if we consider the computation $(\mathrm{M}, \mathrm{k\,S_f})(\mathrm{x}) \downarrow$, the probability of reaching to the final state is more than $1 - \frac{1}{2^k}$.

2.2 $\mathbf{P}^*, \mathbf{NP}^*, \mathbf{NP}^* - \mathbf{Hard}, \mathbf{NP}^* - \mathbf{Complete}$

One of the main questions throughout this article is, how to redefine the most important classes of Complexity Theory in the new theory? As a first attempt, let we try to define P^*as follows:

P^* is the class of all problems that can be determined by a Fuzzy Turing Machine (M, S).

But what exactly do we mean by determined? Since it is possible that we do not reach to the final state, we should consider the possibility associated with $\mathrm{x} \in \mathrm{p}$ for any $\mathrm{p} \in \mathrm{P}^*$ when x belongs to p, and the possibility associated with $\mathrm{x} \notin \mathrm{p}$ when x belongs to p^{c}. Hence, by the above consideration, we are able to modify the definition of P^*, as follows

Definition 3. P^* is a class of problems such that, for any $p \epsilon P^*$ and the probability α, we have a polynomial $Q_{\alpha,p}$ and an associated algorithm $A_{\alpha,p}$ to solve p by probability α such that $Q_{\alpha,p}$ is upper bound of the computation time.
Equivalently, for any $p \epsilon P^*$ (p as a language) and probability α we have an associated algorithm $B_{\alpha,p}$ and a polynomial $Q_{\alpha,p}$ as an upper bound of the computation time.
$x \epsilon p \rightarrow$ By probability α , $B_{\alpha,p} = 1$
$x \notin p \rightarrow$ By probability α, $B_{\alpha,p} = 0$

This is similar to the definition of the class BPP. Equivalently, by considering time as a Fuzzy concept we have BPP^*.

By the above considerations, it is easy to see:

Theorem 1. $P^* = BPP^*$.

The next natural question in TC^* is the situation of the problem P vs NP, more exactly P^* vs NP^*. Firstly, we are going to prove the following proposition about random generators.

Proposition1. By considering time as a fuzzy concept, random Generators exist.

Proof. …
□

Now, let we consider the following definition of NP problems.

Definition 4 The Complexity class **NP** is the set of decision problems like D such that there is a deterministic polynomial time Turing machine M_D and polynomils p_D, q_D in order that for every input x with length x' ($l(x)=x'$)

1. x belongs to D implies there exists string z with length $q_D(x')$ such that for all string y with length $p_D(x')$, $Pr(M_D(x,y,z) = 1) = 1)$
2. x does not belong to D implies for all string z with length $q_D(x')$ such that for all string y with length $p_D(x')$ $Pr(M_D(x,y,z) = 0) = 1$ (The definition is Quoted in [4])

By considering the above definition and by fuzzifying time we have the definition of NP^*.

We define NP^*--hard, NP^*-Complete likewise in below

Definition 5 X is NP^*-hard if for any $Y \in NP^*$, $X >_{\boldsymbol{m}}^* Y$.

Definition 6 X is NP^*-Complete if X is NP^*-hard and $X \in NP^*$.

Theorem 2 SAT is NP^*-Complete.

Proof. SAT belongs to NP, hence $SAT \in NP^*$, by definition. The analogues proof of Cook-Levin's theorem works here. More exactly, by employing the reduction associated with the reduction function f in Cook-Levin theorem with this difference that time is fuzzy, we have the analogous function f^*in the new proof, also here, we consider $>_m^*$ instead of m -reducibility. Lemma 3 guarantees the proof of the theorem. □

Theorem4. $P^* \neq NP^*$ implies $P \neq NP$.

Proof. To prove $P \neq NP$, we apply Theorem 2 and lemma 3.

Suppose $P = NP$ and we remind that SAT is a NP-Complete problem. Hence, there is an algorithm A which solves SAT in Polynomial time.

Considering Fuzzy time, A also solves SAT in polynomial time, hence SAT belongs to P^*. SAT is NP^*-Complete, so $P^* = NP^*$, A contradiction. Consequently, $P \neq NP$. □

Lemma. $SAT \notin P$ implies $SAT \notin P^*$, unless $P = NP$.

Proof. SAT is NP^*-Complete. Suppose $SAT \notin P$. If $SAT \in P^*$ then $P^* = NP^*$. In brief, $P \neq NP$ implies $P^* = NP^*$, which contradicts Theorem 4. □

Theorem 5. $P \neq NP$ implies $P^* \neq NP^*$.

Proof. **Suppose** $P \neq NP$**.** By above lemma, $P \neq NP$ implies $SAT \notin P^*$. But $SAT \in NP^*$, so $P^* \neq NP^*$. □

Chapter 2. MA^*, AM^*

In the previous chapter, by defining the concepts of P, BPP in the new framework, we define the new classes P^*, BPP^*. It is shown that the new classes P^*, BPP^* are both equal to each other. In contrast, what is the alternative definition for the NP class in this new framework? To illustrate NP problems in the Theory of Algorithm, it is required to define a new class for it. Possibly MA is the best choice in probabilistic classes [1], [4] (introduced by Laszlo Babai, Shafi Goldwasser, Micheal Sipser).

Indeed, the MA complexity class is known as an alternative for NP problems in probabilistic classes, we also have a theorem states [2], [3]

$$P = BPP \rightarrow MA = NP$$

The last point, besides $P^* = BPP^*$ confirms our choice. So, let we define the concept of NP problems in fuzzy time by applying and similar to the definition of MA. On the other hand in the previous chapter we defined NP^*, as the second way to define an alternative definition for NP

in TC^*. It is easy to see, these two ways of defining a parallel concept for NP in TC^* , leads us to the equivalent definitions.

Here, we mention the complexity class Merlin-Arthur MA, in Two-sided version definition[4].

Definition 7. The Complexity class **MA** is a set of decision problems like D such that there are deterministic polynomial time Turing machine M_D and polynomials p_D, q_D in order that for every input x with length x' $(l(x)=x')$

1. x belongs to D implies there exists string z with length $q_D(x')$ such that for all string y with length $p_D(x')$ $Pr(M_D(x,y,z)=1) \geq {}^{2}/_{3}$)
2. x does not belong to D implies for all string z with length $q_D(x')$ such that for all string y with length $p_D(x')$ $Pr(M_D(x,y,z)=0) \geq ⅔$ (The definition is Quoted in [4])

Likewise, we remind the complexity class Arthur-Merlin AM in Two-sided version definition [4].

Definition 8. The Complexity class **AM** is a set of decision problems like D such that there are deterministic polynomial time Turing machine M_D and polynomials p_D, q_D in order that for every input x with length x' $(l(x)=x')$

1. x belongs to D implies there exists string z with length $q_D(x')$ such that for all string y with length $p_D(x')$ $Pr(M_D(x,y,z)=1) \geq {}^{2}/_{3}$)
2. x dose not belong to D implies for all string z z with length $q_D(x')$ such that $Pr(\text{for all string } y \text{ with length } p_D(x'), M_D(x,y,z)=0) \geq ⅔$ (The definition is Quoted in [4])

By considering time as a fuzzy concept, we define MA^*. AM^* is defined similarly, by considering Two sided definition of AM in above.

The list of new possible classes which we study here, is

P^*, NP^*, BPP^*, MA^* $AM^*$and AM^*.

Instead of $P = NP$ problem and in parallel to it, we have the following problems

$$BPP^* = MA^*$$

$$BPP^* = AM^*$$

$$MA^* = AM^*$$

Theorems 3&4 shed a light on the above problems.

It is easy to see:

1. $\mathrm{P}^* = \mathrm{BPP}^*$ (Theorem 1)
2. $\mathrm{NP}^* = \mathrm{MA}^*$ (Considering certificate definition of NP)

It is notable to remind, by proposition 1, we have random generators in the new Theory. So, the pseudo-random generators exist too. In addition, we have $\mathrm{P}^* = \mathrm{BPP}^*$ (Theorem 1). In this theory the third major conclusion is about the classes $\mathrm{MA}^*, \mathrm{AM}^*$.

.

Theorem 3. $\mathrm{MA}^* = \mathrm{AM}^*$.

Proof. MA is the nondeterministic version of BPP, AM is the probabilistic version of NP.

So, clearly $\mathrm{AM}^* = \mathrm{NP}^*$and MA^* is the nondeterministic version of BPP^*.

By the way, $\mathrm{P}^* = \mathrm{BPP}^*$. Consequently, MA^* is the nondeterministic version of P^*. By definition, $\mathrm{MA}^* = \mathrm{NP}^*$. In sum, $\mathrm{AM}^* = \mathrm{MA}^* = \mathrm{NP}^*$. □

Moreover, by above we have

Theorem 6. The following statements are equivalent

1. $\mathrm{P} \neq \mathrm{NP}$
2. $\mathrm{P}^* \neq \mathrm{NP}^*$
3. $\mathrm{BPP}^* \neq \mathrm{MA}^* (= \mathrm{AM}^*)$

Proof. By Theorems 2, 3, 4, 5.

Conclusion. Throughout this article, it is shown that by considering time as a fuzzy concept, we have random generators. Under this condition, TC^* as a new theory in the field setting of computation is introduced. Hereafter, in the new theory, some problems in parallel to some of the famous problems in Complexity Theory are solved. In brief, $\mathrm{P}^* = \mathrm{BPP}^*$, $\mathrm{MA}^* = \mathrm{AM}^*$.

Refrences

1. L.Babai “TRADING Group Theory for Randomness”, STOC’85: Proceedings of the seventeenth annual ACM symposium on Theory of Computing, ACM, pp.421-429, 1985

2. O.Goldreich, In a world of P=BPP

3. O.Goldreich, Studies in Complexity and Cryptography: Miscellanea on the interplay between Randomness and Computation , Vol 6650 of Lecture Notes in Computer Science, Springer 2011, P 43.

4. S.Goldwasser; M.Sipser “Private coins versus public coins in interactive proof

systems", STOC'86: Proceedings of the Beighteenth annual ACM symposium on Theory of Computing, ACM, PP.59-68, 1986

5. Van Aten M, On Brouwer, Wadsworth Philosopher's Series, 2004