$P \neq NP$, By accepting to make a shift in the Theory (Time as a fuzzy concept), version 2

The Structure of a Theory (TC*, Theory of Computation based on Fuzzy time)

Farzad Didehvar

didehvar@aut.ac.ir

Amir Kabir University of Technology (Tehran Polytechnic)

Abstract. One of the possible hypothesis about time is to consider time as fuzzy concept, in a way that two instants of time could be overlapped. Historically, some Mathematicians and Philosophers had a similar idea like Brouwer and Husserl[14].

Throughout this article we show the positive impact of this change on Theory of Computation and Complexity Theory to rebuild it in a more successful and fruitful approach. We call this novel Theory TC*.

Keywords. $\mathrm{P} \neq \mathrm{NP}$, P=BPP, Fuzzy Time, Probabilistic Time, TC*, Reducibility, Complexity Theory Problems

1. Introduction

Here, we try to build the structure of a Theory of computation based on considering time as a fuzzy concept.

Actually, there are some reasons to consider time as a fuzzy concept. In this article, we don't go to this side but we remind that Brower and Husserl ideas about the concept of time were similar [14].

Throughout this article, we present the Theory of Computation with Fuzzy Time. Considering the classical definition of Turing Machine we change and modify the concept of Time to Fuzzy time. We call this new Theory TC* [5] 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 are solved [5]. It should be mentioned that in this article, we consider fuzzy number a symmetric one. The point about the symmetry is in the proof of Lemma 3, although we are able to generalize it.

More specifically, we define the new classes of complexity Theory, P*, NP*, BPP* in TC* analogues to the definitions P, NP, BPP as their natural substituted definition. We show P*$\neq$ NP*, P*= BPP*. Finally, we have Theorem 4.

2. Reducibility

In this section, first we define a quasi order relation in TC* analogues to m-reducibility in TC.

We should remind that a fuzzy time Turing Machine is a Turing Machine which works in fuzzy time.

In addition, our Turing Machine is a two tuple (M,S). M is a Turing machine in the usual sense and s is a polynomial function, here M runs in bounded time S equivalently in this machine we compute M(x) in less than S([x]) steps.

First we repeat the Classical definition of m-reducibility:

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

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

Associated definition in TC*

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

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

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

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

By bounded time, we mean for function f there exists a Polynomial function h such that $f(x) \downarrow$ in less than h(length(x)).

We represent $Y >_{\boldsymbol{m}}^{\boldsymbol{\alpha}} X$ by a 5-tuple, (Y, X, f, S_f, α), $S_f(x)$ is the number of steps that f(x) is computed. We define it as follows

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

Is this definition independent from the value of α? ($\alpha > \frac{1}{2}$)

In the first step in order to answer the above question, we need the following simple lemma from probability.

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 a 5-tuple.

Proof. Actually there is k, such that g= (k times repeating f till we reach a solution with probability β) . It is easy to see that, there is such a k.

Definition 2. Lemma 1, shows for $1 > \alpha > \frac{1}{2}$, $Y >_{\boldsymbol{m}}^{\boldsymbol{\alpha}} X$ is independent from α. So, we write $Y >'_{\boldsymbol{m}} X$.

Lemma 2. $\mathrm{Y} >'_{\mathbf{m}} \mathrm{X}$ is a quasi order.

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

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

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

Lemma 3. $Y >_m X$ implies $Y >'_{\boldsymbol{m}} X$

Proof. Here, we consider the fuzzy number is symmetric.

We have computable function f such that

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

f is supported by (M, S_f). The computation of f on x can be depicted by the following transition of configurations in time $S_f(x)$ to reach the final configuration.

Now, we change time to be fuzzy as it is mentioned in above. Now the probability of reaching or passing the final configuration is more than the probability of not to reach to this point.

By probability rules and above comment, if we consummate 2 $S_f(x)$ unit of time, the probability of reaching to the final configuration or passing it, is more than $\frac{3}{4}$ and the probability of not to reach to this final configuration is less than $\frac{1}{4}$. Likewise by consumption of p $S_f(x)$ unit of time, the probability of reaching to the final configuration or passing it, is more than $1 - \frac{1}{pn}$ and the probability of not to reach to this final configuration is less than $\frac{1}{pn}$. So we have, $Y >'_{\boldsymbol{m}} X$.

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

2.2 P*, NP*, NP*-hard, NP*-Compelete

One of the major question here is how we define the most important classes in Complexity Theory in the new theory? As a start we try to define P*. As the first attempt, we try to define it as following:

P* is the class of all problems that are decidable by a Fuzzy Turing Machine (M,S).

But what do we mean by decidable, exactly? Since it is possible we do not reach to final state, So we should speak about the possibility of xϵp for any pϵP* when x belongs to p , and the possibility of x/ϵp when x belongs to p^c. Hence by above consideration we define P* as following:

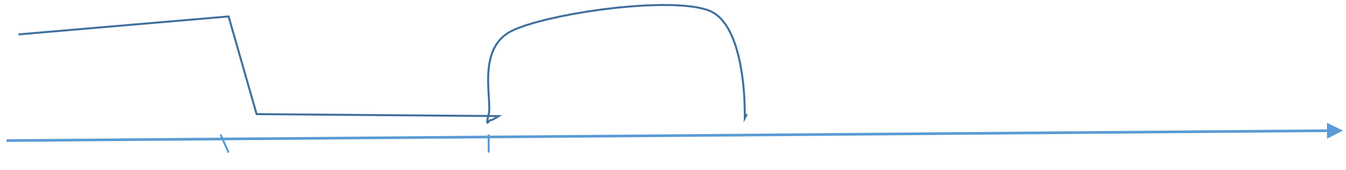
Definition 3: P* is the class of problems for any pϵ **P*** and probability α we have a polynomial $Q\alpha{,}p$ and an associated algorithm $A\alpha{,}p$ for solving p by probability α such that $Q\alpha{,}p$ is upper bound of time of computation.
Equivalently, for any pϵ **P*** (p as a language) and probability α we have an associated algorithm $B\alpha{,}p$ and a polynomial $Q\alpha{,}p$ as an upper bound of time of computation
xϵp → By probability α, $B\alpha$,(x)=1
x/ϵp → By probability α, $B\alpha{,}p(x)$=0
This is equivalent to the definition of the class BPP.
Additionally, by considering time as a Fuzzy concept we have BPP*. It is easy to see that it defines the same class as BPP. Consequently

Theorem 1 P*=BPP*(=BPP) [3], [5].

The next natural question in TC* is the situation of the problem P vs NP, more exactly P* vs NP*.

Proposition 1 Random Generator exists [3], [5].

Proof. By inventing an algorithm we show that random generator exists. It is sufficient to consider an algorithm that in interval times [2n,2n+1] it emits as an output 0 and in interval times [2n+1,2n+2] it emits 1, when time is considered as a classical concept. Now by considering time as a fuzzy concept it is seen easily that we have a random number generator. More exactly, by considering fuzzy time we have probability function p(x), $1 > p(x) > 0$. Such that for any X+t, $1 > t > 0$ and n is a natural number
If X is an odd number by probability p(t) in X+t is equal to 1 (p(t) is near to 1)
If X is an even number by probability p(t) in X+t is equal to 0 (p(t) is near to 1)



The diagram of p(t). It is periodic.

First 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 are deterministic polynomial time Turing machine $\mathrm{M_D}$ and $\mathrm{p_D}, \mathrm{q_D}$ such that for every input x with length $\mathrm{x'}$ ($\mathrm{l(x)=x'}$)

1. x belongs to D implies there exists string z with length $\mathrm{q_D(x')}$ such that for all string y with length $\mathrm{p_D(x')}$ $\mathrm{P\,r(M_D(x,y,z)} = 1) = 1)$
2. x belongs to D implies for all string z with length $\mathrm{q_D(x')}$ such that for all string y with length $\mathrm{p_D(x')}$ $\mathrm{P\,r(M_D(x,y,z)} = 0) = 1$ (The definition is Quoted [13])

By considering the above definition and by fuzifying 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 $Y \in \mathrm{NP}^*$, $X >'_{\boldsymbol{m}} Y$.

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

Theorem 2 SAT is NP*-Complete.

Proof. SAT belongs to NP, hence $SAT \in \mathrm{NP}^*$, by definition.

Analogues to the proof of Cook-Levin theorem by repeating it, and considering the associated reduction by function f when time is fuzzy we have the same function f and considering $>'_{\boldsymbol{m}}$ instead of m-reducibility. Lemma 3 guarantees the proof of theorem.

In [6], by defining the concepts P, BPP in the new framework we have $\mathrm{P}^*, \mathrm{BPP}^*$. It is shown that the new classes $\mathrm{P}^*, \mathrm{BPP}^*$ are both equivalent to BPP. In contrast, what about the substitution of class of NP in this new framework. To represent NP problems in the Theory of Algorithm, it is required to define a new class for that. Possibly the best choice in probabilistic classes in this purpose is MA [10], [13] (introduced by Laszlo Babai, Shafi Goldwasser, Micheal Sipser).

The complexity class MA is known as the candidate of NP problems in probabilistic classes, also we have a theorem states [12]

$$\mathrm{P} = \mathrm{BPP} \rightarrow \mathrm{MA} = \mathrm{NP}$$

This point besides $\mathrm{P}^* = \mathrm{BPP}^*$ strengthen our choice. So, we try to define the NP concept in fuzzy time by applying the definition of MA.

Here, we define MA in Two sided version definition [13].

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

3. x belongs to D implies there exists string z with length $q_D(x')$ such that for all string y with length $p_D(x')$ $\Pr(M_D(x,y,z) = 1) \geq {}^{2}/_{3})$
4. x belongs 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 [13])

As a conclusion, by changing and transforming the literature of Theory of Computation from Classical Time to Fuzzy time the classes of Complexity Theory changes to new classes. Likewise,

We have new problems.

The list of new possible classes are

P^*, BPP^* and MA^*, AM^*

Instead of $P = NP$ problem we have the following problems

$$BPP^* = MA^*$$

$$BPP^* = AM^*$$

$$MA^* = AM^*$$

The two last questions remained unproved.

It is easy to see:

1. $P^* = BPP^*$
2. $NP^* = MA^*$ (Considering certificate definition of NP)
3. $MA^* = MA$

Chapter 2.Pseudorandom generator & NP^+

Pseudo random generators play a major role in Theory of computation. The existence of pseudo random generator by applying classical time leads us to P≠NP. What about theory of computation when we consider time as a fuzzy concept (TC^*)?

By proposition 1, more strongly, we have random generator in our Theory,

To obtain our main result in Theorem*, we define NP+.

Definition 10 (NP+) Non deterministically guess the input for deterministic Turing machine M, we call this new machine $\mathrm{M}+$.

NP+ are the set of languages which accept by some M+.

When we consider time as a fuzzy concept in above, we have NP+*.

NP+ and NP and NP+* are subsets of NP*.

Theorem 3: P*= NP* & the existence of random generator leads us to a contradiction, moreover by proposition 1 we have P*≠ NP*.

(Hint of proof: P*= NP* implies NP+* is a subset of P*. First, we select all the seeds non deterministically, in a high probability we generate all random numbers. Since P*=NP* so the generator is not random. But by Proposition 1, we have a random generator.)

Corollary. PH* doesn't collapse.

Some Problems in New Theory:

1- Creativity and P vs NP

2-MA*=AM*

3-P*=NP* ∩ CO-NP*

Theorem 4 $\boldsymbol{P \neq NP}$.

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

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

By considering Fuzzy time, A solves SAT in polynomial time too and SAT belongs to P*. SAT is NP*-Complete, so P*=NP*. A contradiction.

Consequently, $\mathrm{P} \neq \mathrm{NP}$.

Conclusion. Here, we show considering time as a fuzzy concept, have some major results in solving some famous problems in Complexity Theory in a way that it adopts to the intuition and expectations of people in Theory of Algorithm. In brief, P*≠NP*, P*=BPP*. Finally we prove P≠NP.

Reference:

1. C.Witterich Probabilistic Time, Foundations of Physics, 2010

2. F.Didehvar, About Fuzzy time-Particle interpretation of Quantum Mechanics (It is not an innocent one!), version 1, Philpaper 2019

3. F.Didehvar, By considering Fuzzy time, P=BPP (P*=BPP*), Philpaper 2020

4. F.Didehvar, Double Slit Experiment About Fuzzy time-Particle interpretation of Quantum Mechanics (It is not an innocent one!) Version2, Philpaper 2019

5. F.Didehvar, Fuzzy Time & NP Hardness (P*=BPP*, P* $\neq$ NP*) , Philpaper 2020

6. F.Didehvar, Fuzzy time, A Solution of Unexpected Hanging Paradox, Philpapers 2019

7. F.Didehvar, Fuzzy Time from Paradox to Paradox (Does it solve the contradiction between Quantum Mechanics & General Relativity?), Philpaper

8. F.Didehvar, Is Classical Mathematics Appropriate for Theory of Computation? Philpaper 2018

9. F.Didehvar, SINGULARITIES, About Fuzzy time-Particle interpretation of Quantum Mechanics (It is not an innocent one!) Version 2, Philpaper 2019

10. 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

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

12. 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.

13. 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

14. Van Aten M, On Brouwer, Wadsworth Philosopher's Series, 2004 (Persian translation by Ardeshir.M)