Is this a contradiction?

Farzad Didehvar

didehvar@aut.ac.ir

Department of Mathematics &Computer Science,

Amir Kabir University, Tehran, Iran

Abstract: Here, by introducing a version of "Unexpected hanging paradox" first we try to open a new way and a new explanation for paradoxes, similar to liar paradox. Also, we will show that we have a semantic situation which no syntactical logical system could support it. Finally, we propose a claim in Theory of Computation about the consistency of this Theory.

Keywords: Unexpected hanging paradox, Liar paradox, Turing machine, Axiomatization.

Introduction: The surprise exam paradox (Unexpected hanging paradox) is a well-known paradox that too many works have been done based on that or about it.

Simple formulation of this paradox is in references.

This paradox is known as "Surprise test paradox" too. For some technical reasons we do not apply this name here and we use the more origin name. "The bottle Imp" and "Crocodile dilemma" has some similarity to this paradox.

Historically, O'Conner article [14] was the first article which expose this paradox academically, but this was exposed before.

Some people think that basically, it is not a paradox. Quine work [15] was in this line. Some think the paradox has been resolved by Quine and simply some aspects of it remain to be discussed [8].

In fact, there is no consensus on its nature so we have no final resolution yet, besides all we have different various reformulation of this paradox.

There are different approaches to solve this problem:

1. Logical approach: Some stress more on its logical aspects and they try to attack this problem by a logical approach. As an example some think to apply and to assume self contradictory and self referencing the nature of this problem.

Parallel to this approach, this paradox is applied to give a solution to the first Gödel Theorem and the Second Gödel Theorem. In [6] inspired by surprise examination argument and applying some other Theorems the authors give a new proof for second completeness theorem. Before this work chaitin by applying Berrys paradox has done a similar work [4].It is considered as an attempt to show the self referential features of this paradox. There are similar elder articles. [1],[3]

The explanation of [16] is in this line.

2. Epistemic approach: Some think about this paradox as a paradox of knowledge (Epistemic Paradox), in some approaches it goes to ability or inability to know some statements[9] .In recent years we have some attempts to solve this problem in dynamic epistemic logic.[12], [13]

There are attempts yet to solve this problem once and forever. While numerous resolutions have been proposed for this paradox during last six decades, it continuously surprises us. A number of articles make endeavors to put an end once forever. [11]

Here, throughout this article, our goal is totally different and the article is not in this line at all .We apply a version of this paradox to show that there is

a semantic situation that no formal system supports it, as one of our major claims.

In [2] by applying a version of "Unexpected hanging paradox", we tried to reveal that there is a proof that doesn't show the truth. Here, we try to modify and formalize this problem and by applying a version of this paradox to show the above claim.

In conclusion part, the above shows us a general claim about a contradiction in the "Theory of Computation" (In general Mathematical Modeling) or deficiency of this Theory in modeling some situations.

First, we explain this version of scenario of paradox.

We quote this restated "Unexpected hanging paradox" as follows:

'' A Computer Scientist invents a Computer which argues logically, and also any information and conclusion on this Computer is announced to public openly.

The People of the city do not like this Computer and they appoint a jury which decides about the destiny of this Computer.

The jury decision is declared to Computer Scientist and Computer as follows:

The Computer will be destroyed in the next week on a day that wasn't concluded by the computer itself before that day. Definitely we know everything that the computer knows or everything it will conclude. (The computer is invented in a way that it declares anything it concludes and any new information; we can imagine that it reveals all its conclusions in a printing form).

After announcing it, the computer starts to argue and it finds the following arguments (the argument of 'hanging paradox'). By the way, because of the computer's Intelligence, all people know that destroying the computer is equal to "executing the Computer".

Demonstration of the Computer:

I will not be executed on Saturday, since if I was executed on Saturday, I would be alive till Friday, and on Friday I conclude that I will be executed on Saturday (since based on what jury said I will be executed in the next week). So in case if I remain alive till Friday, I will conclude on that day, I will not be executed on Saturday (in view of the proposed claim of the jury).This argument exists in my data base right now (after concluding by me, right now).In a more exact form, the Computer argues as follows: I have a timer, by using my storage and database I conclude that I will conclude on Friday "I will be executed on the next day, so based on the claims of juries, I will not be executed on Saturday".

2. I will not be executed on Friday, since if I was executed on Friday I would remain alive till Thursday, and on Thursday and our claim in point 1 and based on what jury said("The Computer will be executed in the next week") I conclude that I will be executed on Friday. So by the proposed claim of the jury I will not be executed on Friday (This argument right now exists in my data base after concluding by me, so in case if I remain alive till Thursday, I will conclude on that day, that I will not be executed on Friday). In a more exact form, the Computer argues as follows: I have a timer, by using my storage and database I conclude that I will conclude "on Thursday that I will be executed on the next day, so by the proposed claim of juries, I will not be executed on Friday".

…

6. I will not be executed on Monday, since if I will be executed on Monday I will be alive till Sunday, and on Sunday and the claims made in point numbers 1&2&…&5, I conclude that I will be executed on Monday, since by what jury has said I will be executed in the next week. So, in case if I remain alive till Sunday, I will conclude on such a day that I will not be executed on Monday (Based on the proposed claims of jury), this argument exists right now in my data base after concluding by me. In a more exact form, the Computer argues as follows: I have a timer, based on my storage and database I conclude that I will conclude on Sunday "I will be

executed on the next day, so by the claim of jury, I will not be executed on Monday".

7. I will not be executed on Sunday, since based on points 1-6 I will not be executed on Saturday, Friday …Monday.

So I should be executed on Sunday my conclusion (to be executed on Sunday) concludes that I will not be executed on Sunday.

So, I will not be destroyed in the next week.

The computer didn't conclude more, except it's every day conclusion; it concluded that it will not be executed on the next day based on the similar above argument.

The computer was destroyed on Tuesday and the Computer Scientist complained about this injustice, and he transferred this argument of Computer to journals and the court. In this message the computer scientist mentions that the computer presented a logical argument, based on that he will not be executed in the next week, nevertheless he was executed.

The court said:

The Computer proved that he will not be executed. This is a true proof. We executed him on Tuesday, and as he claimed he didn't conclude that he will be executed on Tuesday. On the contrary, he announced that "he will not be executed on that day". In other words, he proved in his message that by accepting what the judge said as a true claim, he would not be executed in that week. More formally, here we state as follows:

P: we consider what the judge said as a true claim

Q: He would not be executed in this week

"His proof (p|---q) is a true proof, but it doesn't show the Truth". (*)

In this paper, we defend the above claim of the juries (*) and we try to show how we could develop this idea.

It is noteworthy to say that, we don't claim that the above result does not propose a solution for unexpected hanging paradox in all its versions. It is simply considered as the only way to explain this version of "Unexpected hanging paradox". Later on, we will know the above result as a possible explanation for the other versions of this paradox and some other paradoxes like liar paradox. In [10] we can find a more conclusive explanation of Liar paradox.

To formalize the above proof, we call our Computer "A" and whenever we say "A concludes", in our paradox, we replace it by A:[φ]. Also, A:$_i$[φ] stands for "A concludes or utters φ in the ith day".

In the following formalism, $\varphi(i)$ stands for A will be executed in the ith day. We change slightly the scenario. We suppose the ceremony of smashing the computer take places from 11/00-12/00, this gives us a better understanding for the second principal. By $\varphi(i)$, we mean the Computer is being smashed in the i- th day.

A: [ψ] means there exist $1 \le i \le 7$ in which A:$_i$[ψ].

Now in any formalization of the problem and proof we will have the following assertions:

1.$\bigvee_{i=1}^{i=7} \varphi(i)$

2. $\sim\varphi(n) \rightarrow$ [A: $[\sim\varphi(n)]$](In the evening of each day A understands he is not executed and he declares it).

3. $\varphi(k) \rightarrow \bigwedge_{i=1,i\neq k}^{7} \sim\varphi(i)$ (If he is executed in a day, we conclude in the other days he wasn't executed).

4. $(\bigwedge_{i=1}^{k-1}$ A: $[\sim\varphi(i)]) \wedge (\bigwedge_{i=k+1}^{7}$ A: $[\sim\varphi(i)]) \rightarrow$ A:$_{k-1}$ $[\varphi(k)]$

5.A:$_k$ $[\varphi(k+1)] \rightarrow \sim\varphi(k+1)$(If A utters in the i-th day that he will be executed in i+1 th day, he will not be executed in i+1 th day, i=1,…,6,7).

1. $\sim\varphi(7) \equiv \top$

Suppose $\varphi(7) \equiv \top$

$\Lambda_{i=1}^{6} \sim\varphi(i)$ (Principal 3)

$\Lambda_{i=1}^{6}$ A: $[\sim\varphi(i)]$ (Principal 2)

$A:_6 [\varphi(7)]$ (Principal 4)

$\sim\varphi(7) \equiv \top$ (Principal 5)

2. $\sim\varphi(6) \equiv \top$

Suppose $\varphi(6) \equiv \top$

$\Lambda_{i=1}^{5} \sim\varphi(i) \wedge \varphi(6) \wedge \sim(7)$ (Principal 3)

$\Lambda_{i=1}^{5}$ A: $[\sim\varphi(i)] \wedge$ A: $[\varphi(6)] \wedge$ A: $[\sim\varphi(7)]$ (Principal 2)

$A:_5 [\varphi(6)]$ (Principal 4)

$\sim\varphi(6) \equiv \top$ (Principal 5)

3. $\sim\varphi(5) \equiv \top$. Similar to above proof.

……

……

7. $\sim\varphi(1)$. In a similar way.

So we have a contradiction here, since $\varphi(3)$. ■

So, we have a model which its associated formal system is contradictory.

It is notable that any syntactical system that we attribute to the above paradox contains the above statements, whether directly or as a conclusion.

In other words, the proof shows that this formalism and any other formalism are essentially contradictory but we have a semantic situation for this contradictory formalism.

Remark: In any complete formalization of the above paradox we need to extend our language in a way the language includes some sentences in the form of:

$[A:_i \ [A:_j \ (\varphi)]]$, in order to write some statements like the following statement in the demonstration of Computer, I conclude that I will conclude on Friday "I will be executed on the next day, so by the claim of juries, I will not be executed on Saturday".

Actually we have the following axioms too:

$[A:_0 \ [A:_j \ (\varphi(j+1)]] \rightarrow [A:_j \ (\varphi(j+1)]$ j=0,…, 6.

However, for simply showing the contradiction existed in formalizing this semantic situation, it is not essential to face this extension of the language, and the above axioms.

An explanation and some conclusions:

In the above system, there is a contradiction. In brief, the judges claim that the Computer will be executed next week, but the Computer proves that he will not. So this system is a inconsistent system, but at the same time we have a semantic situation for this system. So we have a inconsistent system which has a semantic situation.

In other word, we have semantic situation which no consistent syntactical system supports them. (Here, we have such an example).

As a result, our proof in above does not show any truth. So, there are some incorrigible flaw in modeling and formalizing the proofs. In other word, formal systems are not able to support such semantic situations.

Clearly, this opens a new way to explain some paradoxes similar to liar paradox, as follows:

In such paradoxes the proofs don't show the truth, since there is no consistent syntactical system to support the related semantic situation.

This would be considered as the first central result and theme of this paper. As the last word we

conclude some results in the subject of Computability Theory (and theories in general).

Is this a contradiction in Mathematics and Theory

Of Computation? (The contradiction and introducing the Model)

In this chapter, we present some results about consistency of Theory of Computation.

Firstly we should define a special type of Turing machine. This Turing machine is able to smash itself (From now on instead of "executing" and "smashing" we use "cracking").

To do this, we add symbols of alphabet "s" and "*" to $\sum$ to have $\sum_1$.The machine is fed by a listing of a c.e set A, when it reaches the symbol "s'' firstly it emits "*" as the final output then it halts and it never works.

Definition: Let M be a Turing Machine and A$\subseteq \sum_1^*$is a c.e set. Suppose L_Aa list of A. (M,L_A) is a Turing machine that is fed by strings in $L_A$by the same order in $L_A$and when it reaches the symbol "s" first it emits a symbol"*", then it halts and does not work forever.

Theorem 1: For any Turing machine M and c.e. set A, (M,L_A) is equivalent to a Turing machine.

Proof: First we consider the following cases:

1. The strings of A contain no "s'' it is easy to see that the range and domain of this machine is c.e and the machine is equivalent to a Turing machine.

2. At least one of the strings in A contains s, so the range and domain of the machine would be finite. Hence it will be equivalent to a Turing machine.■

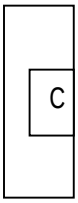
Now we design a (M,L_A) machine to demonstrate the situation in the paradox.

Here, our inputs are (0,0,0,0,0,0,0) represent no day in the week the computer would be cracked on it.

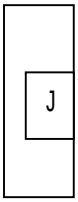
(1)represents it will be cracked on Sunday, (0,1) represents it will be cracked on Monday, (0,0,1) represents it will be cracked on Tuesday, (0,0,0,1) represents it will be cracked on Wednesday, ..., (0,0,0,0,0,0,1) represents it will be cracked on Saturday.

Q_0,....,Q_6,Q_F are usual states (No emission).They represent days of the week, Q_F is the final state, we call the other states "emission states".

Q_c When 1 enters this state as input, it emits "s". (The state of cracking)



This represents the computer in the paradox. It emits 0 when 1 enters as input. It emits "*" when "s" enters as input.



This represents the juries in our scenario. It is able to emit 1. (Here, it emits 1 when C (computer) emits 0 and associated Q_i emits 1 ($1 \leq i \leq 6$)).

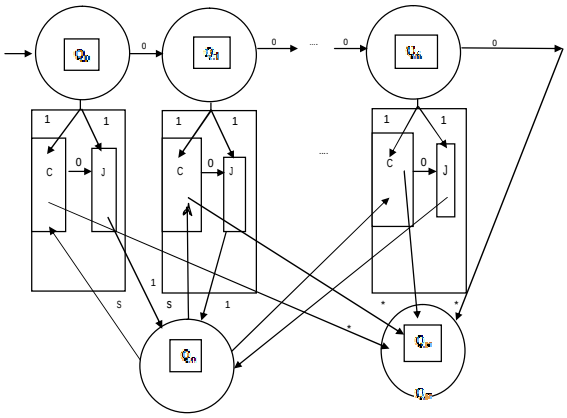


Fig 1

Here the computer has a proof that it will not be cracked and this proof is in its storage. When 1 enters in C and J, C emits 0, (it represents he knows, he will not be cracked). J emits 1 to Q_c and it emits "s" to C so it cracks C. (Since the computer does not know that it will be cracked. More exactly, the computer claims it will not be cracked and J emits 1, so the machine would be cracked). In the final step, C emits * to go the final state.

It is easy to see that, this machine is a (M,L_A) machine for a finite set A, and hence it is equivalent to a Turing machine by Theorem1.

So, any input from the above set of inputs is acceptable either as the day of cracking (executing day) or the computer is not cracked. This claim adapts to the jury claim in restated paradox.

But by demonstration and proof of computer simply (0,0,0,0,0,0,0) is considered as acceptable.

So 7=1, consequently $\forall n\ 2^n=1$.

We have two immediate results:

1. The Theory of Computation is self –contradictory,

2. P=PSPACE =NP.

The second claim seems interesting and exciting, but it is based on a contradiction in the system.

We should mention that, here the proof contradicts the computation.

So, we have a contradiction in the "Theory of computation" when we apply Turing model as our model of computation in order to model this situation.

Conclusion:

It is possible that some think all above is not a real picture of situation. It comes back to the "Theory of physics" that we accept as background.

Otherwise, we have two cases here:

In the first case, the Theory of Computation leads us to a contradiction (More exactly by above description, we are able to put a contradiction in it).It is a proof that we should concentrate more on some subjects and models like Quantum Computation and Randomized algorithm. In brief, it might be better to assume the models whose bits and states are not considered definitely separated. In such cases, the classes of complexity and problems are drastically changed, and the problems there, are not parallel to P=NP problem; so the problems are totally in a different type of form. In practice, any Algorithm and Computation is either a Randomized algorithm or Quantum Computation, based on an existing acceptable theory of Physics besides the errors possibility in Computation. The other possible solution is considering the non-classical logics, like Para consistent Logic.

Although the origin of contradiction seems more fundamental and it roots back to the whole body of Mathematics and Mathematical Modeling and not simply the "Theory of Computation" nevertheless the above arguments is true.

As a last word, most of the proofs and demonstrations in the Theory of Computation (specially in Complexity Theory) employs the three elements, first a situation similar to above (like discrete objects as graph and moving from one node to the other and passing edges,...), a model of computation (like Turing machines) as the second element, and Mathematical proofs and reasoning.

In this regard, there is no specific issue or exception in our example. Moreover, it is a simple example, since no infinite object is involved in this problem.

Consequently, it is not clear based on which property we are able to distinguish of proposed example in this paper from the others. In other words, it seems worryingly adhoc to consider this example as an exception.

Anyway, we face a contradiction, in Computational Modeling of this situation. If it is a true word, any formalizing "Theory of Computation" based on Classical Logic and Turing Model of Computation leads us to a contradiction. As a result, in this frame work the big questions of Classical Complexity Theory are answered.

Therefore as a last word, we have a central question. Is the above situation an inconsistency in Theory of Computation and Mathematics?

References:

1.Fredric B.Fitch, A Goedelized Formulation of The Prediction Paradox, , 1964

2.Farzad Didehvar, When the proof doesn't show the Truth, (Accepted in "The World Philosophy Day", 2010).

3.Fitch F A Goedelized formulation of the prediction paradox, Amer.Phil.Quart1,1964, 161-164

4.G.J.Chaitin, Computational complexity and Goedel Incompletness theorem, ACM SIGACT News 9(1971),11-12

5.Jose luis Ferreira The surprise exam paradox rationality and pragmatics : A simple Game-

6.shira krichmann & Ran Raz work"The surprise examination Paradox and The Second Incompleteness Theorem, Notices of the AMS, vol 57, Number 11, 1454-1458

7.Jose Luis Ferreira, The surprise exam paradox, rationality, and pragmatics: a simple game theoretic analysis, Journal of Economic Methodology 15(3):285-299, September 2008,).

8. Smullyan, Raymond What is the Name of this Book? (19)

9.Timothy Y.Chow, The surprise Examination or Unexpected Hanging Paradox,Amer.Math.Monthly 105(1998),41-51

10. John Barwise and John Etchmendy, The liar (1987), Oxford University Press

11. ken levy, The solution to the surprise exam paradox, The southern Journal of Philosophy, 2009

12.J Gerbrandy, The surprise examination in dynamic epistemic logic, synthese 155:21 33 2007

13. A Marcoci, The surprise examination paradox a review of two so called solutions in dynamic epestimic logic.

14.O'Conner, D(1948). Pragmatic paradoxes, Mind, 57, 358-359

15.Quine, On aso called paradox.Mind, 62, 65-66

16.M Ardeshir, Rasoul Ramezanian a solution to the surprise exam paradox in constructive Mathematics, The review of Symbolic Logic, 1-8)